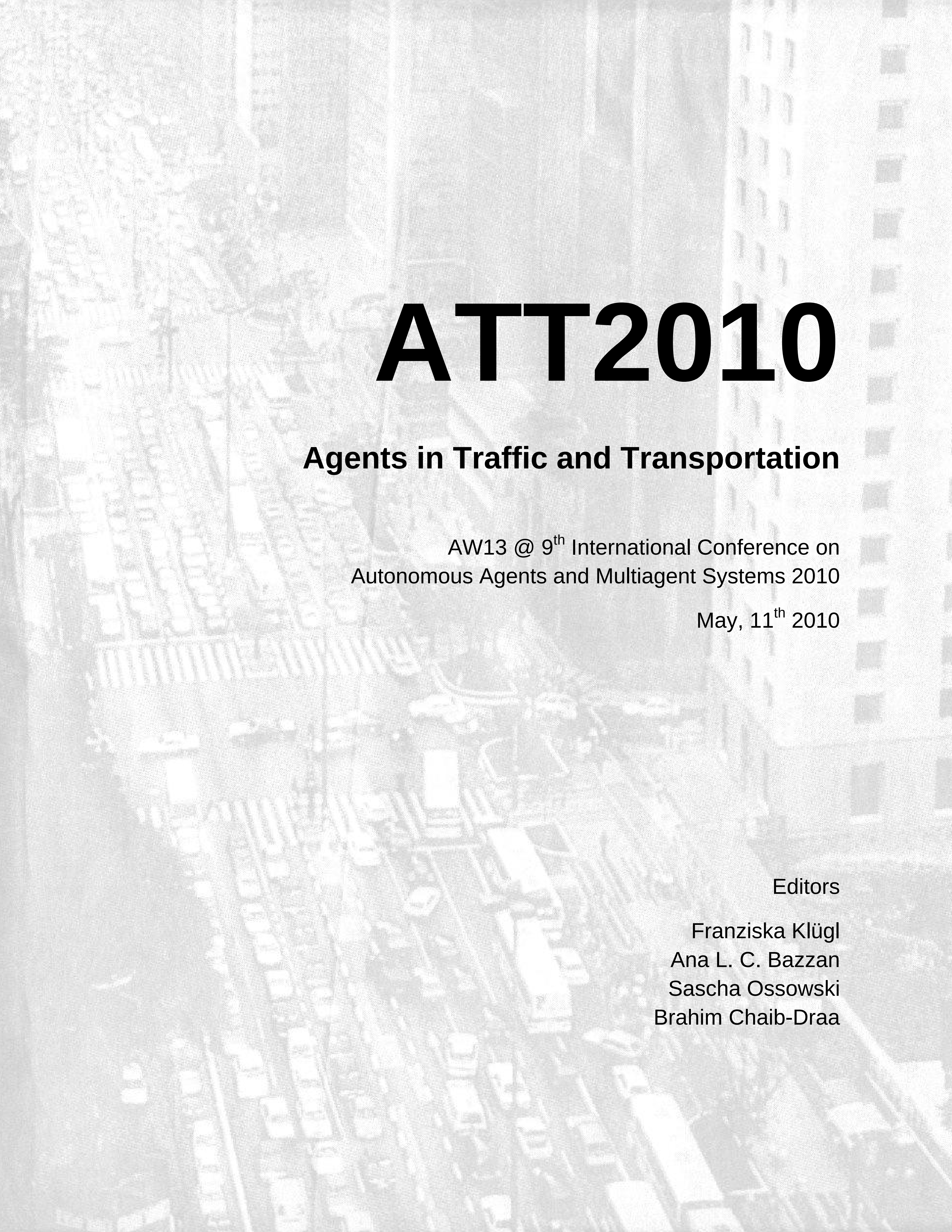

# ATT2010

## Agents in Traffic and Transportation

AW13 @ 9th International Conference on
Autonomous Agents and Multiagent Systems 2010

May, 11th 2010

Editors


Franziska Klügl
Ana L. C. Bazzan
Sascha Ossowski
Brahim Chaib-Draa

## Preface

Since its beginnings back in the year 2000, the international workshop series on "Agents in Traffic and Transportation" (ATT) provides a forum for discussion for researchers and practitioners from the fields of Artificial Intelligence – in particular from the area of Autonomous Agents and Multiagent Systems – and Transportation Engineering. The series aims at promoting cross-fertilization among these disciplines, focussing on how large-scale complex transportation systems can be modelled, simulated, and managed – both at micro and at macro level – employing techniques of agent-based simulation, decentralised coordination, and adaptive regulation.

This sixth edition of ATT was held together with the International Conference on Autonomous Agents and Multiagent Systems (AAMAS), in Toronto (Canada) on May 11, 2010. Previous editions were: Barcelona, together with Autonomous Agents in 2000; Sydney, together with ITS 2001; New York, together with AAMAS 2004; Hakodate, together with AAMAS 2006; Estoril, together with AAMAS 2008.

This edition of the workshop attracted the submission of 20 high-quality papers. All papers were thoroughly reviewed by renowned experts in the field. Based on the reviewers' reports, and the unavoidable space and time constraints associated with the workshop, it was possible to select only 10 of these submissions as full papers. In addition, 4 submissions were accepted as short papers. In the process, a number of good and interesting papers had to be rejected.

The present workshop proceedings cover a broad range of topics related to Agents in Traffic and Transportation, tackling the use of tools and techniques based on multiagent simulation, game-theoretical approaches, and learning to name just a few. We hope you will enjoy it! Finally, we owe a big "Thank you" to all people who dedicated their time and energy to make this edition of ATT a success: from authors and reviewers to hosts and chairs of the AAMAS conference.

Toronto, May 2010 — Franziska Klügl, Ana Bazzan, Sascha Ossowski and Brahim Chaib-Draa.

## Programme Committee

Theo Arentze (TU Eindhoven, Netherlands)
Itzhak Benenson (Tel-Aviv University, Israel)
Vicent Botti (Technical University of Valencia, Spain)
Eduardo Camponogara (Federal University of Santa Catarina, Brazil)
Paul Davidsson (Blekinge Institute of Technology, Sweden)
Kurt Dresner (University of Texas at Austin, USA)
Alberto Fernández Gil (University Rey Juan Carlos, Spain)
Tom Holvoet (KU Leuven, Belgium)
Ronald van Katwijk (TU Delft, Netherlands)
Tamas Mahr (TU Delft, Netherlands)
René Mandiau (University of Valenciennes, France)
Eugenio Oliveira (Universidade de Porto, Portugal)
Rosaldo Rosetti (Universidade de Porto, Portugal)
Andreas Schadschneider (University of Cologne, Germany)
Kardi Teknomo (Arsenal Research, Austria)
Harry Timmermans (TU Eindhoven, NL)
Kagan Tumer (Oregon State University, USA)
Matteo Vasirani (University Rey Juan Carlos, Spain)
Guiseppe Vizzari (University of Milan, Italy)
Mathijs de Weerdt (TU Delft, Netherlands)
Li Weigang (University of Brasilia, Brazil)
Marco A. Wiering (University of Groningen, Netherlands)

## Additional Reviewers

Stefan Axelsson (Blekinge Institute of Technology, Sweden)
Scott Proper (Oregon State University, USA)

## Organizers

Franziska Klügl (Örebro University, Sweden)
Sascha Ossowski (University Rey Juan Carlos, Spain)
Ana L. C. Bazzan (UFRGS, Brazil)
Brahim Chaib-Draa (Laval University, Canada)

## Schedule

**8:30 – 10:00 Session 1: Simulation**



**10:00 – 10:30 Coffee Break**

**10:30 – 12:00 Session 2: Freight Transportation**



**12:00 – 13:30 Lunch Break**

**13:30 – 15:00 Session 3: Modeling and Simulation**



**15:00 – 15:30 Coffee Break**

**15:30 – 17:00 Session 4: Traffic and Transportation Management**

# Modeling Individual Driving Behaviors for Multiagent Traffic Simulation

Hiromitsu Hattori Yuu Nakajima Toru Ishida
Graduate School of Informatics, Kyoto University
Yoshida-Honmachi, Sakyo-ku, Kyoto, 606-8501, Japan
{hatto,nkjm,ishida}@i.kyoto-u.ac.jp

## ABSTRACT

Multiagent-based Simulations (MABS) are increasingly seen as the most attractive approach to reproducing and analyzing diverse social systems. The essential for realizing practical MABS is human behavior modeling technology. In order to make behavior models realistic, it seems natural to learn from human behavior in the real world. The challenge presented in this paper is to obtain an individual behavior model by using participatory modeling technology in the traffic domain. We show a methodology that can elicit prior knowledge for explaining human driving behavior in specific environments, and then construct a driving behavior model based on a set of prior knowledge. In the real world, human drivers often perform unintentional actions, and occasionally they have no logical reason for their actions. In these cases, we cannot elicit prior knowledge to explain them. We are forced to construct a behavior model with an insufficient amount of knowledge to reproduce driving behavior. To construct an individual driving behavior model with insufficient knowledge, we take the approach of using knowledge from others to complement the lack of knowledge from oneself. To clarify that the behavior model, which is filled out by knowledge from others, offers driving behavior individuality, we experimentally confirm that the driving behaviors reproduced by the hybrid model correlate reasonably well with human behavior.

## Categories and Subject Descriptors

I.2.11 [**Distributed Artificial Intelligence**]: Multiagent systems

## General Terms

Design

## Keywords

multiagent simulation, traffic simulation, participatory modeling

## 1. INTRODUCTION

Multiagent-based Simulations (MABS) are increasingly seen as the most attractive approach to reproducing and analyzing diverse social systems including autonomous and heterogeneous decision-making entities, i.e., humans [2, 10]. The key technology to implement MABS is agent modeling. This is because collective phenomena emerge from the local behaviors of many agents; that is, the simulation result depends on each agent's micro-level behavior. Most existing studies, however, use simple or abstract agent models [5, 7]. Our research focus is to develop a methodology for generating behavior models from human behavior for achieving practical simulations.

Participatory modeling is a promising technology with which to obtain individual behavior models based on actual human behavior [9]. Participatory modeling allows us to elicit a human's behavior as well as the reason for the behavior in particular application domains. Such information can be used as prior knowledge to explain a human's individual behavior. For a sequence of human behaviors, we can construct an individual behavior model composed by a set of prior knowledge, each piece of which can explain one of the local behaviors in the sequence.

In this paper, we try to use participatory modeling technology to obtain a human-like behavior model in the traffic domain. A human driver controls his/her car based on his/her driving style. We want to construct a drivering behavior model that can reproduce diverse driving styles. Trying to achieve that with participatory modeling technology raises difficulties when trying to explain a sequence of driving behaviors. In the real world, a human driver occasionally performs unintentional actions (*i.e.*, actions with no logical reason). Additionally, there are cases where the driver cannot remember the reason for his/her actions. As a result, we cannot obtain sufficient prior knowledge to explain his/her driving behavior.

To permit a driver agent model to be created even though the knowledge is insufficient, we take the approach of using complimentary prior knowledge from other drivers. That is to say, if it is impossible to explain a driver's behavior using only the knowledge elicited from the driver, the knowledge acquired from other drivers is used to provide the explanation. This approach allows us to acquire a driving behavior model that is fleshed out (patched) by knowledge from others. In order to know whether the individuality of a driver's behavior is effectively preserved by the patched behavior model or not, we conduct an experiment on a driving behavior model to confirm that it well reproduces the individuality of driving behavior.

## 2. RESEARCH BACKGROUND

In this section, we show the background of our research on multiagent-based simulltion. In MABS research field, it seems that there are two research directions; that is, creating large-scale simulations and elaborating human behav-

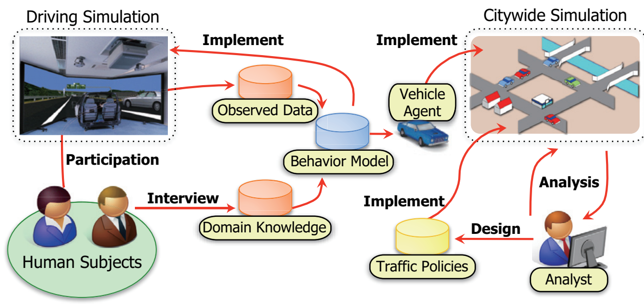


**Figure 1: Overview of Simulation Processes**

ior models. Although many studies have been conducted, the vast majority of them seems to be biased to one viewpoint. Yamamoto and Mizuta proposed a Java-based platform for massive agent-based simulations called ZASE [11]. While they showed that ZASE can perform well with a large number of agents, they did not show great concern about the details of the behavior models. Guyot et al. also constructed models from the log data of participatory simulations, but they ran only small-scale simulations with around 10 agents [3]. This is because there is a trade-off between the scale of multiagent-based simulations (MABS) and the granularity of behavior models in terms of the computation time. However, thanks to recent advances in computing power and large-scale software development technologies, it is becoming possible to conduct massive multiagent simulations with fine-grained behavior models. Therefore, we are trying to achieve massive urban traffic simulations with fine-grained levels of driving behaviors.

Figure 1 shows an overview of simulation processes. There are three major steps: 1) constructing driving behavior models based on participatory driving simulations, 2) conducting massive traffic simulations using constructed behavior models, and 3) analyzing simulation results and sophisticating the simulation environment. We reproduce realistic traffic situations on the virtual space and human subjects participate in simulations on there. Using obtained data from simulations, we can construct driving behavior models (the details of this step is shown in 3). Traffic situations on driving simulator can be improved by applying obtained models for behavior control of neighboring vehicles. After implementing vehicle agents based on obtained models, we conduct citywide traffic simulations. For realizing citywide traffic simulations, we introduced MATSim (Multiagent Transport Simulation Toolkit), which is an open source toolkit for conducting large-scale agent-based traffic simulations [1], developed by Swiss Federal Institute of Technology Zurich and Technische Universität Berlin. We have constructed a traffic simulator by the extension of MATSim to achieve traffic simulations with the fine-grained level of driving behavior models which can reproduce individual driving operations. On the constructed simulator, we can analyze various traffic policies or systems through numbers of simulations.

In this paper, we focus on the modeling part involved in the above processes.

# 3. DRIVER AGENT MODELING

**Figure 2: 3D Virtual Driving Simulator used for Collecting Driving Log Data**

## 3.1 Outline

Using the participatory modeling technique allows us to construct behavior models from not only our (modeler's) knowledge, but the actual behavior of the human subjects. The modeling process consists of the following five steps.

1. Collect human driving log data from trials performed on a 3D virtual driving simulator.
2. Together with domain experts, identify individual driving behaviors by the investigation of collected log data.
3. Collect prior knowledge constituting a driving behavior model by interviewing the subjects of the driving simulation
4. Select meaningful prior knowledge and represent it in formal expression
5. Construct a driving behavior model that can explain human subject's actions based on hypothetical reasoning [8]

We detail each step in the remainder of this section.

## 3.2 Collecting Driving Log on 3D Virtual Driving Simulator

In order to construct a driving behavior model, we need realistic driving data from humans. In the real world, however, it is hard to collect sufficient driving data in actual traffic environments due to the difficulties of setting up an experimental environment. Thus, we use a 3D virtual driving simulator that has a lifelike cockpit and a wide screen that can display a virtual environment (see Figure 2)[1]. Such simulations are often used to train drivers, and so our simulator is expected to yield realistic driving data. Figure 3 is one example of a chart made from driving log data. As shown, we can get information on transitions in running speed (the graph at the top), acceleration (graph second from the top), and the usage of accelerator/brake (graphs at the bottom).

[1]This virtual driving simulator is located at Graduate School of Engineering Division of Global Architecture, Osaka Univ., JAPAN

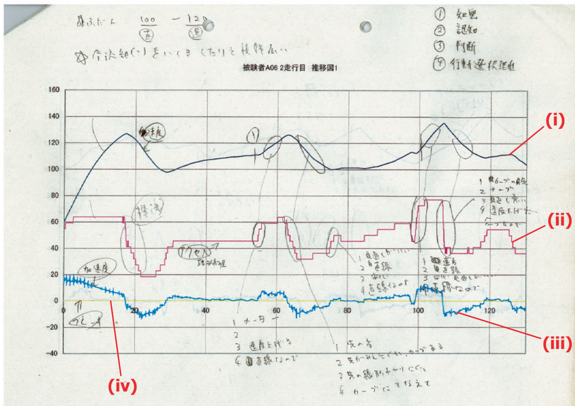


**Figure 3: An Example of a Chart made from Driving Log Data. Each graph (i), (ii), (iii), and (iv) denotes Speed, Acceleration, the Usage of Accelerator, and the Usage of Brake, respectively. Circles on the graph represent the subject's specific behaviors identified by traffic engineers.**

### 3.3 Identifying individual behaviors with domain expert

We investigated the collected driving log data to identify each subject's individual driving behavior. For the investigation, we use the following data collected for each subject.

**1) Mileage(km)** The mileage from the origin

**2) Speed(km/h)** The speed of subject's car

**3) Acceleration(m/s)** The acceleration of subject's car

**4) Usage of Accel.(%)** The usage of accelerator, *i.e.*, accelerator pedal position[2]

We try to capture an individual's behavior by investigating his/her driving log data. In particular, the speed/acceleration transitions provide a lot of useful data. The experiment shown in Section 4 confirms that different drivers have different driving styles, even in identical conditions. Therefore, the sequence of each local driving behavior can be taken as an expression of driver individuality. Figure 3 shows some transitions on graph (ii) in the figure (marked by circles); they represent the results of specific operations. Since, it was difficult for us to accurately identify key transitions from the log data, we elicited the help of domain experts (*i.e.*, traffic engineers).

### 3.4 Interview of Subjects

We interviewed the subjects after they participated in the driving simulation. The purpose of the interview was to gather information on their specific operations, identified in the previous step, for generating prior knowledge. We use screen shots of the simulation and charts like Figure 3 in order to make it easy for the subjects to remember the reasons for his/her actions in the simulation.

In the interview, we asked each subject about the following four points for each specific operation.

[2]In this paper, when the pedal is not depressed, the rate is 0%, and the rate is 100% when the pedal is fully depressed.

**1) Reason/motivation for the operation**
Confirmation of the reason or motivation for the operation

**2) Target of subject's gaze**
Confirming what the subject really gazed at

**3) Recognized target**
Confirming what the subject recognized

**4) Evaluation of the recognition**
Confirming how the subject evaluated the result of the recognition

Figure 3 shows some notes on several of the transitions. For example, the notes at the center of the figure show the following responses:

1) Getting ready for a curve

2) The road in front of me

3) The curve is close and I cannot see into the curve

4) The road forward is unclear

Our analyses of the interview log and charts yielded information on the subjects' operations under a range of conditions, *i.e.*, "sense-act" information. We use such information as prior knowledge and represent it as driving rules, each of which denotes a driving operation made under a certain condition.

### 3.5 Formal representation of collected knowledge

We first cleaned up the collected prior knowledge (*i.e.*, driving rules). For example, in the real example shown in Section 4, we obtained knowledge such as "If I feel fine, I'll step on the accelerator." This kind of knowledge, which is related to feeling, is not suitable for use for modeling because we cannot observe the internal states of humans. Thus, we first eliminated such knowledge. The knowledge remaining is represented using formal expressions based on predicate logic. After a discussion with traffic engineers, we fixed some predicates to represent prior knowledge, see Table 1.

These predicates are also used to formally describe the observations extracted from the driving log data. An observation describes what the subject noticed, and how he/she operated his/her car in the situation presented.

This formal description of prior knowledge and observations allows us to use them in the next step of model construction.

### 3.6 Construction of Driving Behavior Models

*Formalizing the Problem.*

In this paper, we assume that a subject decides his/her next operation based on the surrounding environment as observed from his/her viewpoint. We denote the environment observed by the subject as $E$; it consists of conjunctions of literals about the environment; the environment at time $t$ is tagged $E_t$. The driving model $\mathcal{M}$ is a set of prioritized driving rules $\langle P, \preceq \rangle$, which is a set of driving rules where $\preceq$ represents the priorities of each rule in $P$. $P$ is a subset of *Rules* which is the set of rules obtained from all subjects.

| Predicate | Description |
|---|---|
| `Straight(X)` | `X` is a straight road. |
| `Curve(X)` | `X` is a curve. |
| `Uphill(X)` | `X` is an uphill. |
| `Downhill(X)` | `X` is a downhill. |
| `On(X,Y)` | `Y` is driving on `X`. |
| `InSight(X,Y)` | `Y` can see `X`. |
| `OverDesiredSpeed(X)` | The speed of a car `X` exceeds the desired speed. |
| `UnderDesiredSpeed(X)` | The speed of a car `X` is under the desired speed. |
| `OverCurveSpeed(X,Y)` | The speed of a car `Y` is too high in a curve `X`. |
| `SpeedUp(X)` | A car `X` is speeding up. |
| `SlowDown(X)` | A car `X` is slowing down. |
| `Accelerate(X)` | A car `X` is accelerating. |
| `Decelerate(X)` | A car `X` is decelerating. |

**Table 1: Predicates to Represent Actions**

Therefore, each driving model may be consist of prior knowledge obtained from several human subjects. $\preceq$ is a subset of the Cartesian product, *i.e.* $Rules \times Rules$. Each driving rule in $Rules$ is denoted as $rule_i (0 \leq i \leq j \leq |Rules|)$, so that $\langle rule_i, rule_j \rangle \in \preceq$ is described as $rule_i \preceq rule_j$.

In order to apply hypothetical reasoning [8] to the modeling of driving behaviors, we define driving rules and an operation selection mechanism as domain knowledge $\Sigma$. An element of domain knowledge is indicated by $\sigma_k (0 \leq k \leq |\Sigma|)$. We hypothesize which driving rules are employed by the target subject ($rule_i \in P$), and which rules take priority ($rule_i \preceq rule_j$). A set of these hypotheses is indicated by $H$. Additionally, we describe the subject's behavior from the beginning of the simulation on a 3D simulator, 0, to the end of the simulation, *end*, as observation $G$ and the observation at time $t$ is denoted as $G_t$.

The operation selection mechanism is defined as follows:

**Definition 1** (Driving operation selection: $\sigma_1$) $(\exists rule_i (rule_i \in P \wedge rule_i = \max_{\preceq}\{rule | \texttt{Applicable}(rule, E_t)\}))$
$\Rightarrow \texttt{Do}(operation(rule_i))$

Here, `Applicable` and `Do` are pseudo-predicates meaning that the condition part of a rule is satisfied, and that the subject initiates an operation, respectively. Function *operation* returns the operation initiated by the subject when he/she executes $rule_i$. $\sigma_1$ means a subject employs $rule_i$, the rule that has the highest priority among all applicable operations at $E_t$.

**Definition 2** (Continuation of operation: $\sigma_2$)
A subject can continue his/her current operation.

**Definition 3** (Constraint: $\sigma_3$)
$\forall rule_i, rule_j (rule_i, rule_j \in P \wedge$
$(condition(rule_i) = condition(rule_j))$
$\Rightarrow (operation(rule_i) = operation(rule_j))$

$\sigma_3$ means that $P$ does not include driving rules that have identical condition parts but different operations. Here, the function *condition* returns the precondition of its argument.

We define $G$ and $G_t$ below:

**Definition 4** (Observation $G$)
$G \equiv (G_0 \wedge ... \wedge G_t \wedge ... \wedge G_{end})$

**Definition 5** (Observation $G_t$)
$G_t \equiv (E_t \Rightarrow A_t)$

$A_t$ is the literal represented by predicate `Do`.

The observations, present in driving log data, are described using the predicates shown in Table 1. We use road structure, driving speed, and acceleration pedal operation as observations. A typical description is as follows:

**Example 1** (Description of observation)
$\texttt{Curve}(Curve_1) \wedge \texttt{InSight}(Curve_1, \text{self})$
$\wedge \texttt{Uphill}(Uphill_1) \wedge \texttt{On}(Uphill_1, \text{self})$
$\wedge \texttt{OverDesiredSpeed}(\text{self})$
$\Rightarrow \texttt{Do}(\texttt{ReleaseAccel}(\text{self}))$

This observation means that the subject released the accelerator when he/she sees $Curve_1$ (`InSight`), his/her car is driving $Uphill_1$(`On`), the speed of car exceeds the desired speed (`OverDesiredSpeed`), and he/she is decelerating (`ReleaseAccel`).

*Model Acquisition Process.*
We applied a modeling method based on hypothetical reasoning [6] to acquire a driving behavior model of each human subject. The method should yield models that can explain $G$ in association with $\Sigma$ and $H$. As mentioned above, $\Sigma$ is the operation selection mechanism and operation rules, and $H$ indicates which driving rule is employed by the subject, *i.e.* which rule has priority.

The major steps of the model acquisition algorithm are as follows.

1. The driving model at time $t-1$, $\mathcal{M} = \langle P, \preceq \rangle$, is input.

2. If the target subject continues the same driving operation as at time $t-1$, the algorithm just returns $\mathcal{M}$.

3. If the subject initiates a new operation at time $t$, a driving rule $p$, which is applicable to $E_t$ and can explain $A_t$, is chosen from $P$. $p$ is assigned higher priority than all other rules applicable to $E_t$ in $P$ ($\preceq$ is updated to $\preceq'$); finally, $\mathcal{M} = \langle P, \preceq' \rangle$ is returned. The goal of the algorithm is to obtain a minimal explanation. Therefore, the algorithm first tries to find an applicable rule in the current $P$ to avoid adding another rule.

4. If there is no applicable driving rule in $P$, a driving rule $p$, which is applicable to $E_t$, is chosen from $Rules$. $p$ is assigned higher priority than all other rules applicable to $E_t$ in $Rules$ ($\preceq$ is updated to $\preceq'$); finally, $\mathcal{M} = (P \cup \{p\}, \preceq')$ is returned.
   If $P \cup \{p\}$ is inconsistent, the algorithm returns "$fail$".

# 4. A REAL EXAMPLE OF DRIVER AGENT MODELING

We conducted an experiment to construct driver agent models based on the modeling methodology we mentioned above. In this section, we show how the proposed methodology works, and what models were constructed in the experiment.

## 4.1 Setting and Modeling Process

First, we describe the setting of the driving simulation used to collect driving log data. In this experiment, we used an 11km virtual highway whose layout is shown in Figure 4. For simplicity, in this experiment, each human subject drove

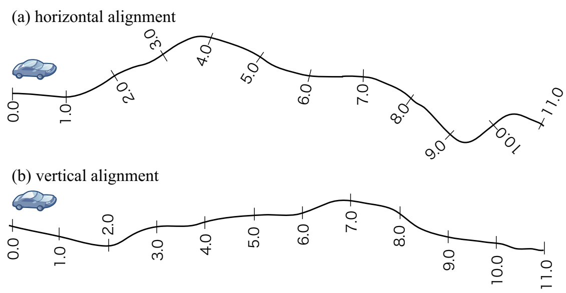


**Figure 4: Road Structure in 3D Driving Simulator**

| RuleID | Description of a rule |
|---|---|
| **rule$_{01}$** | If a subject is driving a curve, he/she releases the accelerator. |
| **rule$_{02}$** | If a subject is driving a straight, he/she steps the accelerator. |
| **rule$_{03}$** | If a subject is driving uphill, he/she steps on the accelerator. |
| **rule$_{04}$** | If a subject is driving downhill, he/she releases the accelerator. |
| **rule$_{05}$** | If a subject sees a curve ahead, he/she releases the accelerator. |
| **rule$_{06}$** | If a subject sees a straight ahead, he/she steps the accelerator. |
| **rule$_{07}$** | If a subject sees an uphill ahead, he/she steps on the accelerator. |
| **rule$_{08}$** | If a subject sees a downhill ahead, he/she releases the accelerator. |
| **rule$_{09}$** | If the speed exceeds the desired speed, a subject releases the accelerator. |
| **rule$_{10}$** | If a subject is driving under the desired speed, he/she steps on the accelerator. |
| **rule$_{11}$** | If a vehicle slow down, he/she steps on the accelerator. |
| **rule$_{12}$** | If a vehicle speed up, he/she release the accelerator. |

**Table 2: Obtained Knowledge from Human Subjects**

alone, so that we could elicit prior knowledge representing just the driving operations. There were 36 subjects, each of them had experience in using the 3D simulator. We could successfully obtain prior knowledge (*i.e.*, driving rules) from all subjects through a collaboration with traffic engineers, but some subjects provided only one or two rules. The set of obtained prior knowledge is shown in Table 2. Because the experiment was held on a virtual highway with no other cars, all subjects used just the accelerator. In a few cases, the subject used the brake, but had no logical reason for doing so. Prior knowledge indicated how the human subject might decide to use the accelerator considering surrounding road structure, current velocity, and own desired speed.

We then formally expressed the obtained prior knowledge by using the predicates we defined to describe observations. Example 2 shows a description of prior knowledge.

**Example 2** (Description of prior knowledge)

**rule$_5$**:
  **if** $\texttt{Curve}(x) \wedge \texttt{InSight}(x,\text{self})$ **then** $\texttt{ReleaseAccel}(\text{self})$

**rule$_7$**:
  **if** $\texttt{Uphill}(x) \wedge \texttt{InSight}(x,\text{self})$ **then** $\texttt{Accelerate}(\text{self})$

For instance, $rule_5$ means that if there is an upcoming curve $x$ ($\texttt{Curve}(x)$) and if the subject ("self") sees the curve $x$ ($\texttt{InSight}(x,\text{self})$), he/she releases the accelerator ($\texttt{ReleaseAccel}(\text{self})$). $rule_7$ means that if hill is to be climbed $x$ ($\texttt{Uphill}(x)$) and the subject sees that, he/she steps on the accelerator ($\texttt{Accelerate}(\text{self})$).

Finally, we used the obtained knowledge and observations to construct driving behavior models using the algorithm shown in 3.6. We show here an example of the modeling process using the rules and observation in Example 1 and 2. This example shows how $\texttt{Do}(\texttt{ReleaseAccel}(\text{self}))$ is derived. Here, we assume $rule_{12} \in P$.

1. In order to derive $\texttt{Do}(\texttt{ReleaseAccel}(\text{self}))$, due to $\sigma_1$, it is required to prove that $action(rule_i) = \texttt{Release Accel}(\text{self})$, $rule_i \in P$, and that $rule_i = \max_{\preceq}\{rule | \texttt{Applicable}(rule, E_{t-1})\}$ are true.
2. Because the consequences of $rule_5$ is $\texttt{Do}(\texttt{ReleaseAccel}(\text{self}))$, they validate $action(rule_i) = \texttt{ReleaseAccel}(\text{self})$.
3. Substitute $rule_5$ for $rule_i$
   (a) Choose an assumption, $rule_5 \in P$, from $H$ to prove $rule_5 \in P$ is true.
   (b) Choose an assumption, $rule_7 \preceq rule_5$ from $H$ to prove $rule_5 = \max_{\preceq}\{rule | \texttt{Applicable}(rule, E_{t-1})\}$ is true.
   (c) $h_{t-1} = \{\{rule_{12}, rule_5\}, \{\{rule_7 \preceq rule_5\}\}\}$ is acquired.

This process is iterated until $G_{end}$ can be explained; the result is a driving model.

## 4.2 Acquired Driving Behavior Models

In the experiment, we could construct driving behavior models for all subjects. In this section, we show some examples of the driving behavior models so acquired. Table 3 shows a set of driving rules and their priorities. Figure 5 shows transitions in running speed and acceleration of the subjects and their corresponding driver agents. In Figure 5, the vertical axis and horizontal axis represent speed (km/h) and mileage (km), respectively. The bold blue line and bold green line plot subject's running speed and acceleration, respectively. The thin red line and thin orange line represent driver agent's running speed and acceleration, respectively.

**Case 1 for $S_1$:** The driving behavior model of subject $S_1$ consists of 6 driving rules and the relationships defining their priorities. The road section of 1km - 7km is a gentle ascending slope with some curves, as shown in Figure 4. $S_1$ drove under his/her desired speed (120km/h) in this zone (see Figure 5(A-1)). $S_1$'s behavior model can reproduce his/her driving log by the application of three rules, $rule_{03}$, $rule_{07}$, and $rule_{10}$. The running speed is increased by these rules. After the 7km point, the road curves downhill. Because $S_1$'s model does not include a rule to release the accelerator, at first, the running speed is continuously increased. However, once the speed exceeds the desired speed, $rule_{09}$ is fired, and the accelerator pedal is released. If the speed becomes too slow, this model can recover because $rule_{11}$, which is used to speed-up when car speed becomes too slow, is prioritized over $rule_{01}$ and $rule_{05}$ which are used to release the accelerator in a curve.

**Case 2 for $S_2$:** The driving behavior model of subject $S_2$ includes 8 driving rules. In Figure 5 (A-1) and (A-2),

| ID | Driving behavior model |
|---|---|
| $S_1$ | $P = \{rule_{01}, rule_{03}, rule_{05}, rule_{09}, rule_{10}, rule_{11}\}$ <br> $\preceq = \{rule_{10} \preceq rule_{01}, rule_{01} \preceq rule_{05}, rule_{10} \preceq rule_{05},$ <br> $rule_{05} \preceq rule_{10}, rule_{01} \preceq rule_{10}, rule_{01} \preceq rule_{03},$ <br> $rule_{05} \preceq rule_{03}, rule_{03} \preceq rule_{01}, rule_{03} \preceq rule_{09},$ <br> $rule_{09} \preceq rule_{03}, rule_{09} \preceq rule_{05}, rule_{01} \preceq rule_{11},$ <br> $rule_{05} \preceq rule_{11}, rule_{09} \preceq rule_{11}\}$ |
| $S_2$ | $P = \{rule_{01}, rule_{02}, rule_{04}, rule_{05}, rule_{06}, rule_{09}, rule_{10},$ <br> $rule_{11}\}$ <br> $\preceq = \{rule_{01} \preceq rule_{04}, rule_{09} \preceq rule_{01}, rule_{01} \preceq rule_{11},$ <br> $rule_{09} \preceq rule_{11}, rule_{11} \preceq rule_{09}, rule_{09} \preceq rule_{02},$ <br> $rule_{02} \preceq rule_{06}, rule_{09} \preceq rule_{06}, rule_{02} \preceq rule_{09},$ <br> $rule_{11} \preceq rule_{05}, rule_{05} \preceq rule_{11}\}$ |
| $S_3$ | $P = \{rule_{01}, rule_{02}, rule_{03}, rule_{04}, rule_{05}, rule_{06}, rule_{11}\}$ <br> $\preceq = \{rule_{04} \preceq rule_{02}, rule_{11} \preceq rule_{04}, rule_{04} \preceq rule_{11},$ <br> $rule_{04} \preceq rule_{01}, rule_{11} \preceq rule_{01}, rule_{01} \preceq rule_{11},$ <br> $rule_{06} \preceq rule_{05}, rule_{01} \preceq rule_{06}, rule_{11} \preceq rule_{06},$ <br> $rule_{02} \preceq rule_{01}, rule_{02} \preceq rule_{06}, rule_{01} \preceq rule_{03},$ <br> $rule_{05} \preceq rule_{03}, rule_{03} \preceq rule_{01}, rule_{03} \preceq rule_{11},$ <br> $rule_{05} \preceq rule_{11}, rule_{11} \preceq rule_{03}, rule_{03} \preceq rule_{06},$ <br> $rule_{02} \preceq rule_{03}, rule_{06} \preceq rule_{03}, rule_{03} \preceq rule_{05}\}$ |

**Table 3: Examples of Acquired Driving Behavior Models**

$S_2$'s behavior looks similar to $S_1$. The difference is apparent around 7km - 9km region. $S_2$ drove at around 100km/h while $S_1$ exceeded 100km/h. $S_2$'s model can reproduce this difference in driving behavior. It includes $rule_4$, representing "if the subject sees a downhill ahead, he/she releases the accelerator." Therefore, $S_2$'s model lowers the speed. This is one example of realizing individuality in driving style.

**Case 3 for $S_3$:** $S_3$ was a driver whose driving style was hard to explain and reproduce. The frequency of acceleration is relatively high. This is because he/she seems keen to maintain his/her desired speed exactly (100km/h). As shown in Figure 5 (A-3), $S_3$ speeds up little by little to just over 100km/h. The model of $S_3$ can reproduce this driving style by including both $rule_{10}$ ("If the car speed up, he/she release the accelerator") and $rule_{12}$ ("If the subject is driving under the desired speed, he/she steps on the accelerator"). A comparison of the transitions in acceleration makes it clear that $S_3$'s model yields behavior different from those of the other two models.

## 5. EVALUATION AND DISCUSSION

The previous section claimed that our behavior models can reasonably reproduce individual behaviors. In this section, we investigate the quality of the acquired behavior models through quantitative metrics. First, we evaluate whether the acquired models can well reproduce the transitions in running speed. To do that, we calculated the correlation value between the running speed of the human subject and that of his/her behavior model. Such correlation value is a time-tested and an academically accepted index to quantitatively measure the performance of simulations, especially traffic simulations [4]. Table 4(a) shows correlation values for the running speed of human subjects $S_1$, $S_2$, and $S_3$ and their agents. Bold values in the table shows the correlation value between human subjects' log data and the corresponding agents' log data. This data confirms that the first two models for $S_1$ and $S_2$ reasonably reproduce the transitions in running speed. Although the correlation value of the model for $S_3$ is not as high, it still exceeds 0.60. The average correlation value for all human subjects was 0.72. While this is not an outstanding value, we think the quality

| | | $S_1$ | | $S_2$ | | $S_3$ | |
|---|---|---|---|---|---|---|---|
| | | H | A | H | A | H | A |
| $S_1$ | Human | 1 | * | * | * | * | * |
| | Agent | **0.95** | 1 | * | * | * | * |
| $S_2$ | Human | 0.66 | 0.62 | 1 | * | * | * |
| | Agent | 0.90 | 0.87 | **0.72** | 1 | * | * |
| $S_3$ | Human | 0.30 | 0.21 | 0.59 | 0.34 | 1 | * |
| | Agent | 0.05 | -0.03 | 0.1 | -0.03 | **0.61** | 1 |

(a) Correlation value for the running speed of humans and agents

| ID | Entity | Average | Standard Dev. |
|---|---|---|---|
| $S_1$ | Human | 100.9 | 10.6 |
| | Agent | 95.8 | 9.1 |
| $S_2$ | Human | 91.6 | 7.18 |
| | Agent | 86.8 | 9.5 |
| $S_3$ | Human | 102.9 | 5.32 |
| | Agent | 100.2 | 8.22 |

(b) Average and Standard deviation of the running speed

**Table 4: Comparison between Human Subjects' Log Data and Agents' Log Data**

of the acquired behavior models is acceptable given that the behavior models were created using intermingled knowledge. Additionally, from the data shown in this table, we can acquire models that can reproduce individual driving styles. For example, the model for $S_1$ is best at reproducing subject $S_1$'s driving style, it does not well reproduce those of others. The correlation values between $S_1$'s model and $S_2$ ($S_3$) are 0.62 and 0.21. In particular, as we can sense from Figure 5(A), the model for $S_3$ is highly uncorrelated. The correlation values for $S_1$ and $S_2$ are 0.05 and 0.1, respectively. Accordingly, we have succeeded in acquiring individual driving behavior models, each of which can reproduce the characteristic driving style of a different human subject.

The above evaluation assessed the agreement of transitions in running speed, but the actual speeds are equally important. Thus, we assessed whether the speeds were similar or not. Figure 5 (B) shows the distribution of running speeds. This figure plots the number of opportunities to drive at each speed. In this figure, the blue bar is for the human subjects and the red bar is the result of the behavior models. In Table 4(b), we also plot the average and the standard deviation of the running speed of three examples. We can confirm that there is no crucial misfit in the standard deviation for all cases, so that the acquired models can well reproduce driving at the approximate speed with human subjects. In particular, for $S_1$, both of transitions in running speed and value of the speed are approximate. Also, for $S_3$, both human subject and his/her behavior model can drive at the approximate running speed and the characteristic driving style using the accelerator at highly frequent rates. As a result, we can acquire driving behavior models which can reasonably well reproduce individual driving styles of human subjects.

## 6. CONCLUSION

The modeling methodology proposed in this paper represents another direction in behavior modeling for realizing human-like individual agent behavior. Our method does not rely on the modeler's knowledge or ability, but learns from actual human responses by applying the participatory mod-

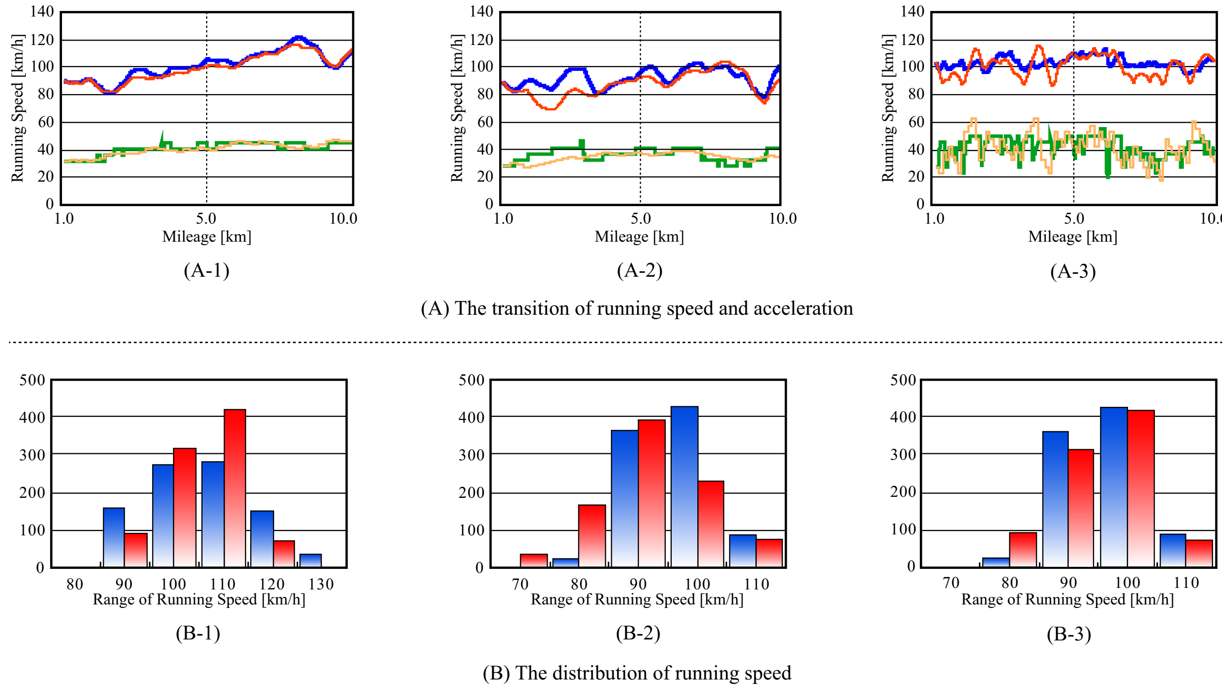


**Figure 5: Transitions in Running Speed and Acceleration of Human Subjects and Corresponding Driver Agent**

eling technique. We can explicitly obtain information on humans' characteristic behavior, *i.e.*, prior knowledge, through the modeling process, and then construct diverse and individual agent behavior models from the obtained knowledge. As shown in the evaluation, we can obtain reasonably well correlated driving behavior from agents.

Our next step is to incorporate interaction models into agents for realizing reasonable interaction among neighboring vehicle agents. Then, we try to invesigate the effect during large-scale traffic simulations.

## 7. REFERENCES

[1] M. Balmer, K. Meister, M. Rieser, K. Nagel, and K. W. Axhausen. Agent-based simulation of travel demand: Structure and computational performance of matsim-t. In *Proc. of the 2nd TRB Conference on Innovations in Travel Modeling*, 2008.

[2] J. M. Epstein and R. L. Axtell. *Growing Artificial Societies: Social Science from the Bottom Up*. MIT Press, 1996.

[3] P. Guyot, A. Drogoul, and C. Lemaitre. Using emergence in participatory simulations to design multi-agent systems. In *Proc. of the 4th International Confrence on Autonomous Agents and Multiagent Systems (AAMAS-2005)*, pages 199–203, 2005.

[4] J. Hourdakis, P. G. Michalopoulos, and J. Kottommannil. Practical procedure for calibrating microscopic traffic simulation models. In *Proc. of the TRB 82nd Annual Meeting (TRB-03)*, pages 130–139, 2003.

[5] T. Moyaux, B. Chaib-draa, and S. D'Amours. Multi-agent simulation of collaborative strategies in a supply chain. In *Proc. of the 3rd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-04)*, pages 52–59, 2004.

[6] Y. Murakami, Y. Sugimoto, and T. Ishida. Modeling human behavior for virtual training systems. In *Proc. of the 4th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-05)*, pages 127–132, 2005.

[7] L. Panait. A pheromone-based utility model for collaborative foraging. In *Proc. of the 3rd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-04)*, pages 36–43, 2004.

[8] D. Poole. *The Knowledge Frontier*, chapter Theorist: A logical reasoning system for defaults and diagnosis. Springer-Verlag, 1987.

[9] D. Torii, T. Ishida, and F. Bousquet. Modeling agents and interactions in agricultural economics. In *Proc. of the 5th International Joint Conference on AUtonomous Agents and Multiagent Systems (AAMAS-2006)*, pages 81–88, 2006.

[10] M. Vasirani and S. Ossowski. A market-inspired approach to reservation-based urban road traffic management. In *Proc. of the 8th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-09)*, pages 617–624, 2009.

[11] G. Yamamoto, H. Tai, and H. Mizuta. A platform for massive agent-based simulation and its evaluation. In N. Jamali, P. Scerri, and T. Sugawara, editors, *Lecture Notes in Computer Science*, volume 5043, pages 1–12. Springer, 2008.

# Towards a more accurate agent-based multi-lane highway simulation

Yi Luo and Ladislau Bölöni
School of Electrical Engineering and Computer Science
University of Central Florida
Orlando, Florida
yiluo@mail.ucf.edu, lboloni@eecs.ucf.edu

## ABSTRACT

The next several years will see an acceleration of the adoption of intelligent driving aid devices. Studying the impact of such devices on the overall traffic performance and safety requires highly realistic microscopic traffic simulation models, which account not only for the overall traffic flow, but also for the details and variability of the individual driver's behavior. In this paper, we first describe a series of improvements on current state of the art multilane highway driving models. The general theme of these improvements is to make the models more agent-like, by considering the specific goals and limitations of the individual drivers and vehicles. We are also performing a more detailed simulation of some of the critical steps in multi-lane highway driving, such as the process of merging at highway entrances and changing lanes. We apply our model to a real world example of the busy commuter highway 408, which crosses Orlando. Through a series of experiments, we investigate the effects of individual driver behavior on the traffic flow.

## Categories and Subject Descriptors

I.2.11 [**Computing Methodologies**]: Distributed Artificial Intelligence: Intelligent Agents

## General Terms

Algorithms, Human Factors

## Keywords

Agent-based simulation,Traffic models

## 1. INTRODUCTION

The upcoming years will see an unprecedented adoption of computing technologies in the traditionally conservative world of car manufacturing. We already see sensing systems such as rear and side view cameras and blind spot warning systems, as well as actuating systems such as "follow the preceeding car" cruise control systems, lane maintenance and lane departure warning systems, and automated parking aids. The upcoming standardization of vehicle-to-vehicle and vehicle-to-infrastructure communication around the 5.9GHz DSRC model will likely usher in a collection of new technologies, such as long range signaling of intentions, advance warning of traffic conditions, real time accident warning as well as negotiated long-lifetime highway convoys. Despite many technologies performing actuation as well as sensing, due to legal reasons, many of these technologies will be positioned as *intelligent driver aids* as opposed to *driving automation*. At least for the next decade, the human driver will still be (at least nominally) in control.

Will these devices allow for faster, safer, less congested traffic? This question is surprisingly hard, because the answer depends not only on the technological achievement of the device but also on the social mechanics of the driving. Even a safety device with a proven efficiency such as Anti-Lock Brakes (ABS) generated controversy about its overall effect. If drivers learn to drive on an ABS equipped car, they do not acquire certain skills such as "pumping the brakes", and can become overly confident of the ability of the car to stop on slippery roads. An even more subtle problem is that a driver of an ABS equipped car might overestimate the driving abilities of the other participants in the traffic.

The upcoming proliferation of intelligent driving aid devices will create a traffic with a heterogeneous mix of driving abilities. While some technologies such as ABS and stability control will be probably almost universal for new vehicles, other capabilities will be a source of differentiation for the individual manufacturers. And of course, the overall traffic mix will include older vehicles which possess none of these technologies.

We can conclude that the overall impact of the intelligent driver aids on the traffic will be a mix of clear improvement and unintended negative consequences. The improved driver control and extended amount of information available to the driver can be used for a more efficient driving and early detection of dangerous situations. On the other hand, a thus-equipped driver might be able to push the limits, drive faster, change lanes with only minimal space available, reduce car following distances - all this relying on the superior technology.

An even more problematic situation appears when the majority of drivers rely on intelligent driving aids. This can create traffic flow patterns which are safe and efficient for driver-aid-equipped vehicles (for instance, long and tight convoys relying of highly accurate speed synchronization), but which are dangerous for traffic participants without the necessary technology.

Except for post-facto studies of already deployed technologies, microscopic traffic simulation is virtually the only available technology which can study that impact of intelligent driver aids to the overall flow of the traffic. To achieve this, however, we need to simulate certain factors which currently are only marginally represented in the simulation literature. We need to consider the decision making process of the individual drivers, as well as the way in which the individual driver aids influence it. We need to also represent and model the psychological model of the human driver, its intentions, sensing and actuating limitations and strategies to cope with them (such as defensive driving), and the specifics of its awareness of the situation.

Finally, we need to model in more detail some of the specific actions taken by the drivers, such as lane changing, merging into the traffic, and exiting the highway, as these actions are the source of most traffic accidents.

Many of these requirements can be expressed by simply saying that the simulation of the traffic should be based on the simulation of independent agents. Significant previous work exists in traffic simulation based on statistical physics, particle systems, flow models and cellular automata [**?**, 1, 3]. For instance, dangerous situations arising during lane-changes have been investigated in [4]. We do believe that with increasing power of current computers, the performance increase due to the particle and cellular automata model becomes less important. We can perform real time simulations of long highway stretches representing the position of the cars without discretization, and even considering the gradual transition from one lane to another. Similarly, we can afford to model the individual decision making process of each driver in a relatively detailed way.

In this paper, we describe our work towards developing an agent based model for multilane highway driving, which reaches the sufficient level of realism for modeling the potential impact of intelligent driving aids.

We start with one of the most realistic microscopic traffic models. The simulation technologies which represent the starting point for our model are summarized in Section 2. Then we augment this model with a series of agent oriented extensions in Section 3. Section 4 describes an experimental study based on an accurately modeled stretch of highway (Highway 408, a busy commuter highway crossing Orlando in the East-West direction). We conclude in Section 5.

# 2. A BASELINE FRAMEWORK FOR MULTI-LANE HIGHWAY SIMULATION

We started our work by implementing in our simulation framework a collection of technologies which together represent a good snapshot of the state of the art in microscopic multi-lane highway simulation. We have chosen as starting point models which are relatively simple and can be augmented with behavioral considerations without a major disruption to the model.

Our baseline is a collection of three technologies: a time-continuous car following model, a lane change model and a human driver model. The first two form the base on which the augmentations described in the next section are applied. The human driver model has been superseeded by our model, but it has been highly influential on its design.

## 2.1 The car following model

Car following models describe the behavior of a car on a single lane highway. Most such models calculate the acceleration or deceleration of the car though a formula of the following general pattern:

$$\frac{dv_i(t)}{dt} = f(\Delta x_i, v_i, \Delta v_i) \tag{1}$$

where $\Delta x_i = x_{i+1}(t) - x_i(t)$ is the distance between the vehicle and its immediate leader, and $\Delta v_i = v_i(t) - v_{i+1}(t)$ is the approaching speed. The specific formula we choose to use is the one introduced by Treiber et al. [5]:

$$\frac{dv_i(t)}{dt} = a\left[1 - \left(\frac{v_i}{v_0}\right)^4 - \left(\frac{\delta(v_i, \Delta v_i)}{\Delta x_i}\right)^2\right] \tag{2}$$

where $a$ is the maximum acceleration of the vehicle, $v_0$ is the desired speed, and $\delta(.)$ is the desired distance from the leading vehicle. This distance depends on a number of parameters through the following formula:

$$\delta(v_i, \Delta v_i) = \Delta x_{min} + v_i T + \frac{v_i \Delta v_i}{2\sqrt{ab}} \tag{3}$$

where $\Delta x_{min}$ is the minimum distance in case of congestion ($v_i = 0$), $T$ is the safe time headway which models the buffering time of the driver, and $b$ is the comfortable deceleration, which couldn't be less than -9 $m/s^2$ on a dry road.

Let us now discuss the intuitions behind this formula. On a free road, the instant acceleration changes from the maximum acceleration $a$ (when the vehicle is still $v_i = 0$) to 0 (when the vehicle reaches its desired speed $v_i = v_0$). If a vehicle follows a leader vehicle with a negligable approaching speed ($\Delta v_i \approx 0$), the term $v_i T$ in Equation 3 dominates such that the vehicle maintains a distance $v_i T$ from the leader.

In the situation that the vehicle approaches the leader with a high speed, the last term $v_i \Delta v_i / 2\sqrt{ab}$ dominates and the formula dictates a decceleration. The most extreme case is when the vehicle moves with its desired speed $v_0$ and observes a still obstacle at the distance of $x_i$. To avoid a collision, the vehicle must brake with deceleration $-b$ when it reaches a distance of $\Delta x_i = v_i^2/2b$. Indeed, this is exactly what the model predicts:

$$\frac{dv_i(t)}{dt} = -a\left(\frac{\delta}{\Delta x_i}\right)^2 = -a\frac{\left(\frac{v_i \Delta v_i}{2\sqrt{ab}}\right)^2}{\Delta x_i^2} = -\frac{v_i^4}{4b\Delta x_i^2} = -b \tag{4}$$

The car following model, defined in this way is considered *collision free*. Of course, this assumes that the drivers have perfect information about their environments. Collisions can still happen if, for instance, a static obstacle or a slow moving vehicle appears on the road at a distance where the vehicle can not come to a stop even if braking with maximum force. We also note that the model assumes that there is no delay between the moment when the acceleration is computed and actual application to the vehicle.

## 2.2 The lane changing model

Our baseline model extends the car following model with the lane change model described by Kesting et al. [2]. This model assumes that lane changes happen instantaneously: for a shift to the left lane, a vehicle which has been previously in the middle lane, at time $t$ disappears from the middle lane and appears in the left lane. This opens the possibility that a car, coming from behind in the new lane with a higher speed can not break sufficiently quickly and collides with the lane changing car. The model assumes that it is the responsibility of the lane changing car to ensure that the rear left vehicle $j-1$ has sufficient buffer distance such that it can decelerate before hitting the lane changing vehicle

$$\hat{a}_{j-1}(t) \geq -b_{max} \tag{5}$$

If this condition is not satisfied, the vehicle concludes that it is not safe to change lanes.

The second feature of the lane changing model is the analysis of the motivations to change lanes, and the "politeness of the drivers". We assume that the goal of the vehicles is to achieve their desired speed, which implies a certain desired acceleration $\hat{a}_i$. The motivation of the driver to change lanes is such that it can achieve this acceleration (which, we assume, is not achievable in the current lane). However, the changing of lanes might also trigger accelerations in the other vehicles: for instance, it allows the current follower to accelerate, and it might force the new follower to brake.

The notion of *politeness* models the fact that the driver might consider the accelerations of the other vehicles as well when taking a decision to change the lane. The politeness parameter $p$ specifies how much does the vehicle discount the other vehicles' desired acceleration compared to its own. A value $p = 0$ indicates an impolite, fully selfish driver which does not care about other drivers (however, it still considers the safety criteria). The vehicle $i$ will decide to change the lane if the following inequality is verified:

$$(\hat{a}_i + p \cdot (\hat{a}_{j-1} + \hat{a}_{i-1}) - (a_i + p \cdot (a_{i-1} + a_{j-1})) \geq \Delta p_{th} \tag{6}$$

$\Delta p_{th}$ is the politeness threshold. The left hand side is the difference between the new accelerations $\hat{a}_i$, $\hat{a}_{j-1}$ and $\hat{a}_{i-1}$ if the vehicle $i$ successfully changes into the target lane and the old accelerations $a_{i-1}$ and $a_{j-1}$ if it doesn't change lane. The intuition is that the vehicle favors to change lane only when the advantage of the action is greater than the disadvantage it exerts to its neighboring vehicles. However, because the vehicle $i$ can not obtain the parameters $(T, v_0, a, b)$ for its successor $i-1$ and $j-1$, the utility of lane change can only be calculated by vehicle $i$'s own parameters.

## 2.3 Human driver model

The models described until now still maintain some elements of the physical models used for traffic simulation starting from the 1950's. Once we move beyond the global picture, many details of the traffic are due to the specific behavior, psychology, cognitive skills and limitations of the *human driver*. There is a general tendency to extend traffic models towards the more accurate modeling of the human driver. Many aspects of our work are in this direction.

A human driver is in some aspects "less capable", but in other aspects "more capable" than the abstract driver envisioned in the models considered up to this point.

It is less capable, because the human driver will inevitably spend some time reasoning about the traffic situation, which delays action. There is also an effect of the cognitive load - humans can make only a limited number of independent decisions per unit of time. Finally, humans occasionally make mistakes, either by taking the wrong decision, not investigating the environment (for instance, by missing a car in the blind spot), or by actuating incorrectly (pushing the wrong pedal or not keeping the car in the lane).

On the other hand, humans usually form a more complete picture of the specific traffic, involving cars around the vehicle, instead of just the vehicle they are currently following. The driver scans ahead several vehicles, and also, occasionally scans the position of vehicles behind it on its own and neighboring lanes. Drivers can also make predictions about the movement patterns of the vehicles. These predictions are facilitated by the similarity of the human drivers' minds. Drivers can recognize the intention of another vehicle to change lanes in the early phases of the action, or even during preparation. In addition, there are standard inter-vehicle signaling methods, such as brake lights and turn signals.

Current microscopic traffic models implement a subset of these human behaviors. Our baseline model is based on Treiber et al. [6] and implements the following aspects.

First, we consider the fact that humans can not perform an indefinite number of decisions per unit of time. This is modeled by considering a time step $\Delta t$. At every time step $\Delta t$ the drivers observe the traffic and make a decision about acceleration. This acceleration value will remain constant for the next interval $\Delta t$:

$$\begin{aligned} v_i(t + \triangle t) &= v_i(t) + \dot{v_i}(t)\triangle t \\ x_i(t + \triangle t) &= x_i(t) + v_i(t)\triangle t + \frac{1}{2}\dot{v_i}(t)\triangle t^2 \end{aligned} \tag{7}$$

Another aspect of the human behavior modeled is the reaction time $T'$ necessary to reason about the traffic situation and make decisions accordingly. This can be achieved by substituting in Equation 1 the current state ($\Delta x_i$, $v_i$, $\Delta v_i$) at time $t - T'$. If $t - T'$ falls between two simulation steps, then it will be adjusted as:

$$x(t - T') = \beta x_{t-n-1} + (1 - \beta)x_{t-n} \tag{8}$$

# 3. AGENT-ORIENTED EXTENSIONS

The model presented in the previous section creates realistic multi-lane highway behavior in terms of global numbers and overall look of the traffic. If we are considering it, however, from the point of view of the individual driver, many of the individually important details are "simplified away". The system is calibrated for assuming no accidents (and also no near misses). It assumes that cars are driving on an indefinitely long highway with no concept of source and destination. The cars do not have a preference for a particular lane. There is no modeling of the maneuvers necessary to merge into the traffic from an entrance, neither the lane changes necessary to leave the highway at a particular exit. By assuming that lane changes occur in a single simulation step, it ignores the complex preparation and perspective change needed to change lanes.

The main goal of our model is to extend the realism of the model for a number of significant events in the course of driving. In the following we describe our contributions which extend or replace components of the baseline model.

## 3.1 Sensing and actuation delays

One of the important real world factors in driving is that there are some psychologically, technologically and physically motivated delays between the information based on which a decision is made, the time when the decision is actually taken and the moment when the consequences of a decision (such as applying a certain acceleration or decceleration or changing lanes) is actually enacted.

Specifically, we will introduce two delay factors:

- The **sensing delay** $t_{sense}$ is the delay between the time when a certain situation happens and the time when this situation becomes available to the driver for reflex action or conscious decision making. The sensing delay will exist even for non-human driver assistant systems: for instance a radar-based car following model

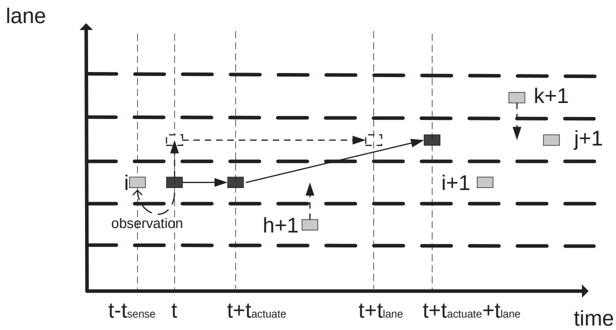


**Figure 1: The sensing and actuation delays for a driver agent making a decision at time $t$**

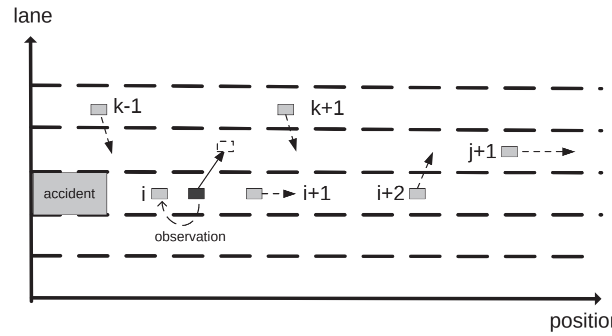


**Figure 2: At time $t$, agent $i$ tries to evaluate the decision of left change.**

will have its own delay establishing the distance and speed of the preceding car.

- The **actuation delay** $t_{actuation}$ is the time between the decision making and the actual enactment of the acceleration in the motor vehicle. This includes several factors: first there is the actual decision making time and the motor enactment of the human driver. This is the component which is extended if the driver is under the influence of alcohol or certain medication types. Second, there is the time which is required to enact the control on the car's user interface: for instance, moving the feet from the acceleration to brake pedal and depressing it. Finally, the vehicle's actuation time is the time it takes between receiving a certain command, such as braking, to the moment when the vehicle actually decelerates with the specified value.

In summary, if a decision is made at time $t$, it is made based on the state of the world at time $t - t_{sense}$ and the decision is enacted at time $t + t_{actuation}$ (see Figure 1).

## 3.2 Increasing the realism of lane changes

Our baseline model assumes that the lane changes are instantaneous: a vehicle appears in the neighboring lane and disappears from the current lane without any warning. The following car will need to hit the brakes at the very instant when this movement happens. In practice, cars change lanes along a diagonal lane, over a period of time $t_{lane}$ (see Figure 1). At the moment when the lane change starts, the cars in the back on all the lanes will know the intention of the driver, and they will react accordingly. To drive safely, the followers on the destination lane need to act as if the lane change has been completed as soon as it starts. On the other hand, the followers on the source lane act as if the car is still on the target lane until the completion of the maneuver. This caution is justified by the fact that the cars can, indeed, abandon a line change in the middle of the maneuver. We can model this pessimistic reaction by making the assumption that during the lane change maneuver the vehicle occupies both lanes.

This model has implications for the car following and lane-changing model. In the Figure 2, at time $t$, agent $i$ tries to evaluate the decision of the left change based on the observation at $t - t_{sense}$. Behind itself, the agent observes an accident so it considers no follower in the original lane. On the left, the agent finds no vehicle, but a vehicle two lanes left is changing to the right. So the new follower in the target lane should be vehicle $k - 1$. There are several possible new leaders in the target lane: vehicle $k+1$, $j+1$ or $i+2$. In general, the agent should consider all the vehicles in the target lane, as well as all the vehicles which are moving towards the target lane. In this example, the new leader should be vehicle $k + 1$ as it is the nearest vehicle in the target lane.

To model the cognitive limitations of human drivers, we do not allow vehicles to decide on a second lane change during the time it is engaged in the first. During the lane change, the agent can only control the acceleration of the vehicle. The new accelerations are calculated based on the predicted leader on the destination lane.

## 3.3 Abandoning lane changes

Let us consider a situation when two vehicles, driving in parallel on the left and right lanes of a three-lane highway simultaneously make the decision to move to the middle lane. This leads to an accident under all the models discussed previously[1]. In real life, however, such situations rarely lead to accidents, because the drivers will become aware of the other driver's intentions either through the index signals or by the movement of the car. Seeing the dangerous situation developing, one or both drivers abandon the lane change and remain in (or return to) their previous lanes.

We implement this behavior in the model. When the two cars initiate the lane change, they will be perceived by other cars as occupying two lanes. Their accelerations will be, however, based on the target lane. If in their predicted position in the common target lane either the accident checker predicts and accident or the agent calculates an acceleration which exceeds the safety limit, the agent will abandon the lane change. If the vehicles are not aligned the vehicle which will abandon the lane will be the one behind. For aligned vehicles both vehicles will abandon the lane change.

## 3.4 Visibility constraints and defensive driving

Drivers on the highway can only see and consider a subset of the vehicles ahead of them. This is normally expressed through a distance value indicating the farthest distance ahead where a driver can locate a vehicle. Our model

[1]The accident can be avoided under the highly artificial assumptions that the lane change is instantaneous, there is no sensing and actuation delay and the vehicles decision and actions are performed in a strict, non-overlapping sequence.

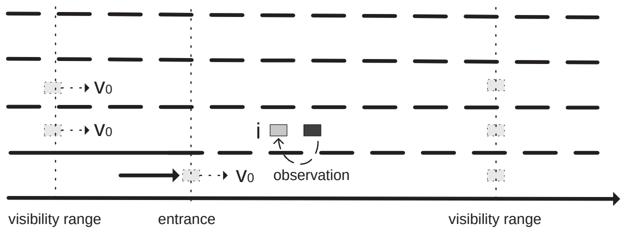


**Figure 3: Modeling visibility constraints and defensive driver.**

represents the visibility constraints by limiting the vehicles considered in the equations to those which are inside the visibility range.

Environmental conditions such as fog, rain or night affect the visibility range of all the vehicles. However, the visibility can change from vehicle to vehicle. Visibility can be reduced by the driver's vision problems or vehicle dependent factors such as broken windshield wipers. On the other hand, technologies such as fog lights, infrared nighttime vision, or, in the future, inter-vehicle notification can extend the visibility range. To allow such modeling, our simulator allows us to individually set the visibility on a vehicle-by-vehicle basis.

Low visibility situations can trigger accidents because if a vehicle suddenly appears at the edge of the visibility range, the driver might not be able to brake in time. Human drivers respond to this through *defensive driving*, by setting a lower speed under low visibility.

We model defensive driving by adding virtual, non-moving vehicles (obstacles) at the end of the visibility range (see Figure 3). This obstacle appears only in the calculations of acceleration, and will move with the vehicle. The overall result on all lanes is that the vehicles will set their speed in such a way as to be able to stop at the visibility edge. Defensive driving vehicles will not reach their nominal desired speed even on an empty highway even if the highway is empty.

## 3.5 Modeling the source and destination

Many previous simulations only modeled linear stretches of highways. Our goal is to create a more realistic system which also models the lane changes, the entrances, exits, and lane shifts on the highway. In a typical highway crossing an urban area, there is a high density of entrances, which are usually, but not always, paired with exits. Most exits are right exits, but occasional left exits exist. The number of lanes in a highway changes. A typical pattern for a highway crossing a city is that the number of lanes increases from 2 outside the city, to 3, 4 or 5 lanes in downtown, followed by a gradual decrease as the highway moves out from the city. In addition, a frequent feature of the highways is the gradual shift of the lanes to the right. For many exits the rightmost lane becomes an exit lane and a new lane is added to the left.

For many highways, relatively good statistics are available for the inflow rate $In(i)$ and outflow rate $Out(i)$, at a specific entrance or exit $i$. This value is easily measurable by simply counting the cars at these exits and entrances. There is, however, no information about the respective source and destinations of the cars, because this would require a full trace of every vehicle throughout the length of the highway.

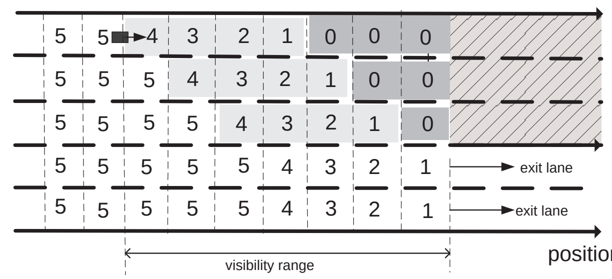


**Figure 4: The evaluation of lane preferences of a vehicle approaching the destination exit. Note that the preferences are comparable only within a single column.**

As a note, such a tracing *is* possible for vehicles which are using automatic payment transponders at toll roads but this information is not publicly available, and it does not include cars paying cash.

We are using a probabilistic model for the entrance and exit of the vehicles. We assume that the vehicles arrive at entrances with an average rate $In(i)$ following a Poisson distribution. Vehicles have a merging lane of limited length, if this is full, they are queued at the entrance. For each vehicle, we assume that its destination is one of the highway exits downstream. The actual destination is chosen stochastically, with the probability that the vehicle entering at entrance $i$ will have a destination at exit $j$ being:

$$Pr(j) = \frac{Out(j)}{Out(j) + \sum_{k>j} Out(k) - \sum_{l>j} In(l)} \tag{9}$$

where $In(l)$ is the inflow rate of entrance with label $l$, and $Out(k)$ is the outflow rate of exit with label $k$. The denominator in the equation 9 is the total number of vehicles which will pass or exit the location. However, the selection probability is calculated in condition that the vehicle doesn't exit before $j$, so we need to normalize them as

$$Pr(i,j) = \prod_{i<m<j} (1 - Pr(m))Pr(j) \tag{10}$$

## 3.6 Modeling the preference of vehicles for specific lanes on the highway

Our baseline model assumes that the only reason for a vehicle to change lane is to be able to achieve an acceleration which will bring it closer to the desired speed. In real life, however, drivers also consider other factors. Most drivers will prefer not to drive on the leftmost lane because of the distractions and slowdowns created by entering and exiting cars. On the other hand, when vehicles are approaching their destination exit, they need to gradually get closer to the exit lane. If drivers are notified that a lane ends or it is blocked by an accident, they will try to move away from the lane as soon as possible. We model all these aspects by introducing a preference value for each lane, by each driver, for every point of the road. The introduction of preferences introduces two problems: how are the preference values chosen and how are the preference values affecting the driving.

The preference values depend on the source and destination of the vehicle, the events on the road (such as obstacles and accidents) as well as the preferences of the human driver.

They do not depend on the other vehicles on the road, which are already modeled by the baseline model.

Figure 4 shows the evolution of the preferences of a vehicle driving on a five lane highway which is planning to exit at an exit where the two rightmost lanes are exit lanes. While initially the preferences are identical across all the lanes, the preferences gradually shift towards the right. Right before the exit only the two exit lanes have non-zero preferences.

The preference values are used as a weight in the line change decision process. To allow this we reinterpret the Equation 6 in the following way: $U_{current} = \Delta p_{th}$ is the utility of the current lane, while $U_i = (\hat{a}_i + p \cdot (\hat{a}_{j-1} + \hat{a}_{i-1}) - (a_i + p \cdot (a_{i-1} + a_{j-1}))$ is the utility of the neighboring lane $i$. The vehicle will move from the current lane $i$ to the neighboring lane $j$ only if $Pr(i) \cdot U_{current} < Pr(j) \cdot U_j$. Notice that the lane change decision is still subject to the safety conditions.

Let us now consider some of the implications of this model. A vehicle travels on the fastest lane. When its destination exit approaches, the preference weights will start to gently favor the right lanes. However, if the other lanes are slower and the desired speed is high, the vehicle will still remain in the leftmost lane, until its preference drops to zero. At this moment, the vehicle will definitely *want* to move to a lane on the right, but it might not be able to do it safely for a while, due to the lane busy with cars. It is possible that the vehicle will not be able to make it to the exit lane in time and it will miss the exit. Such an occurrence is more likely if the vehicle has a fast decrease of the priorities over a short distance span, in contrast with vehicles which adapt their priority long distance ahead of the exit.

### 3.7 Modeling accidents

Despite the best efforts of the participants in traffic, dangerous situations and accidents do occur in real world highway driving. It should not be the goal of a simulation to model a hypothetical, collision free model. Also, we do not want to introduce accidents for reasons related to the internal working of the simulation.

We want to model the real causes of accidents - whether that be human error coupled with risky driving behavior, increased reaction time due to alcohol or medication, slippery roads or low visibility. Although we strive to model the dangerous and accident situations from first principles, inevitably, some calibration based on real world data will be necessary.

In addition to accidents, which are rare events, we also model and detect situations which are *dangerous*. We also model the aftermath of the accidents, such as lane closures and the congestions triggered by them.

We are considering three specific situations:

- **Dangerous situation:** are cases when an accident did not occur, but only because of a "lucky" set of choices made by other cars. One such example are accidents avoided through the abandoning of the lane changes.
- **Minor accident:** is a collision at relative speeds of < 10 km/h, or a lateral collision when changing lanes. For a light collision, we assume that the vehicles are damaged, they are not continuing to travel, but they will not constitute a road block (they are able to move to the side of the road).
- **Major accident:** the vehicles collide with a relative speed of > 10 km/h. We assume that the vehicles will occupy a highway lane for a period $t_{accident}$. Even if the collision is a result of two vehicles colliding, we assume that only one highway lane will be blocked.

| Parameter | Symbol | Value |
|---|---|---|
| simulation step | $\Delta t$ | $0.1s$ |
| maximum deceleration | $b_{max}$ | $5.0m/s^2$ |
| car length | $x_{length}$ | $4m$ |
| minimum distance | $\Delta x_{min}$ | $2m$ |
| acceleration | $a$ | $1.5m/s^2$ |
| desired deceleration | $b$ | $2.0m/s^2$ |
| headway time | $T$ | $1.5s$ |
| desired speed | $v_0$ | $105 \pm r\%$ |
| politeness | $p$ | $0.5 \pm r\%$ |
| politeness threshold | $\Delta p_{th}$ | $0.2 \pm r\%$ |
| visibility range | $x_{visibility}$ | $200m \pm r\%$ |
| sensing time | $t_{sense}$ | $0.1s$ |
| actuating time | $t_{actuation}$ | $0.4s$ |
| lane change time | $t_{lane}$ | $2.0s$ |
| heterogeneity range | $r$ | $0\% \ldots 50\%$ |

**Table 1: Simulation parameters**

## 4. EXPERIMENTAL RESULTS

### 4.1 Experimental settings

In this section we will describe a series of initial experiments performed with out simulator, which implements the baseline model in Section 2 and the agent oriented enhancements in Section 3.

We create a heterogeneous population of vehicles by choosing the vehicles' individual parameters from a heterogeneity range $r$. The simulation parameters are summarized in Table 1.

### 4.2 High traffic simulation

For the first set of simulations, we use a hypothetical model of a 4km-long stretch of highway depicted in Figure 5. The originally 3-lane highway has an entrance at 1000m, and it is temporarily extended with a merging lane. At 2500m an exit lane starts which ends at an exit at 3000m. The simulation assumes a very heavy flow of vehicles (1000 vehicle/hour/lane). Figure 6 shows the number of dangerous situations versus the average politeness factor. As expected, more polite driving reduces the number of dangerous situations. Figure 7 shows the number of lane changes function of the politeness. As most of the lane changes done by a car impact negatively the neighboring car's utility, the number of lane changes decreases with the increase in politeness.

### 4.3 Modeling a real highway

In the second part of our simulation we are modeling a stretch of Highway 408 between UCF and Goldenrod Road (see Figure 8). The inflow and outflow rates for each entrance and exit were taken from official statistics [7].

Figure 9 plots the accident rate function of the actuation time. The main source of the accidents are situations where the vehicles could not abort a lane change in time. As expected, the accident rate increases with the actuation time, with a remarkable take-off after the actuation rate is

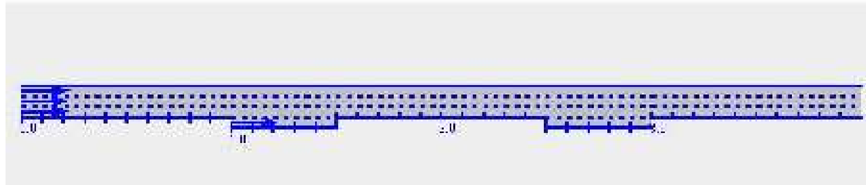

**Figure 5: The sample scenario used to calibrate our model**

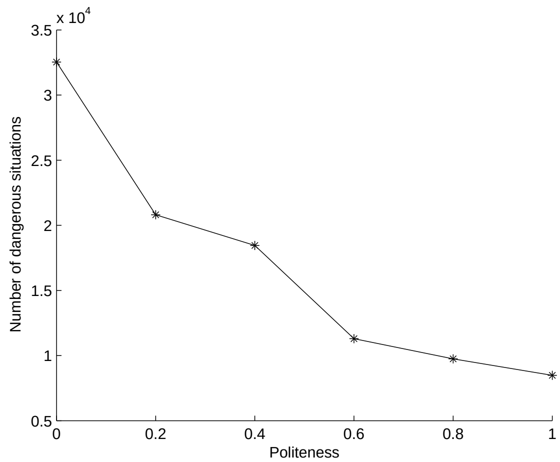


**Figure 6: Dangerous situations function of politeness**

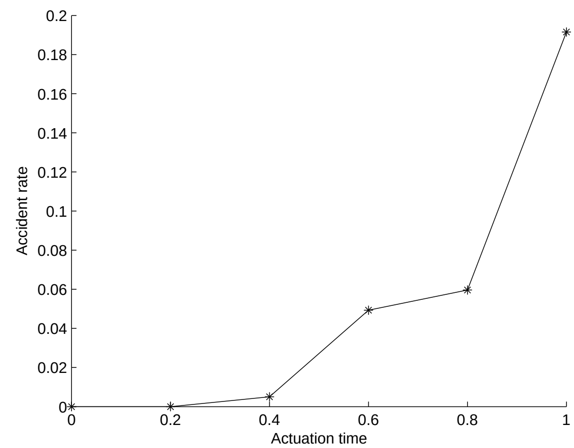


**Figure 9: Accident rate function of the actuation time on highway 408**

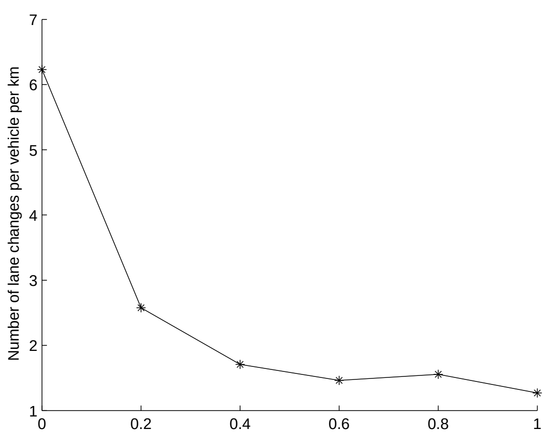


**Figure 7: Lane changes function of politeness**

higher than 0.4. (Such a universally high actuation time corresponds to the unlikely situation that all the drivers are DUI).

In the last two experiments we study the impact of the heterogeneity of the drivers on the driving performance. Figure 10 plots the number of lane changes versus the heterogeneity range $r$. As expected, the increase in the heterogeneity of the drivers triggers higher number of lane changes as the drivers with faster desired speed overtake the slower ones.

Figure 11 plots the average relative speed of the drivers over their complete trip from source to destination. For instance, if a driver's desired speed was 100 km/h but it averaged only 70 km/h from source to destination, the relative speed is 0.7. Although this appears low, it also includes the time spent merging in traffic, as well as the time spent sitting in congested traffic. We note that the heterogeneity of the drivers decreases the average relative speed, by creating more traffic congestion and forcing more lane changes.

## 5. CONCLUSIONS AND FUTURE WORK

In this paper we described a simulation framework which allows a more accurate modeling of individual drivers behavior in multi-lane highway driving, as well as the more specific simulation of specific events such as merging into traffic, preparation for an exit, avoiding accident cars and so on. This simulator opens wide avenues for future studies concerning the implications of particular human driving styles, vehicle heterogeneity and intelligent driving aids.

## 6. REFERENCES


[1] D. Helbing. Traffic and related self-driven many-particle systems. *Reviews of modern physics*, 73(4):1067–1141, 2001.

[2] A. Kesting, M. Treiber, and D. Helbing. General lane-changing model MOBIL for car-following models. *Transportation Research Record: Journal of the Transportation Research Board*, 1999(-1):86–94, 2007.

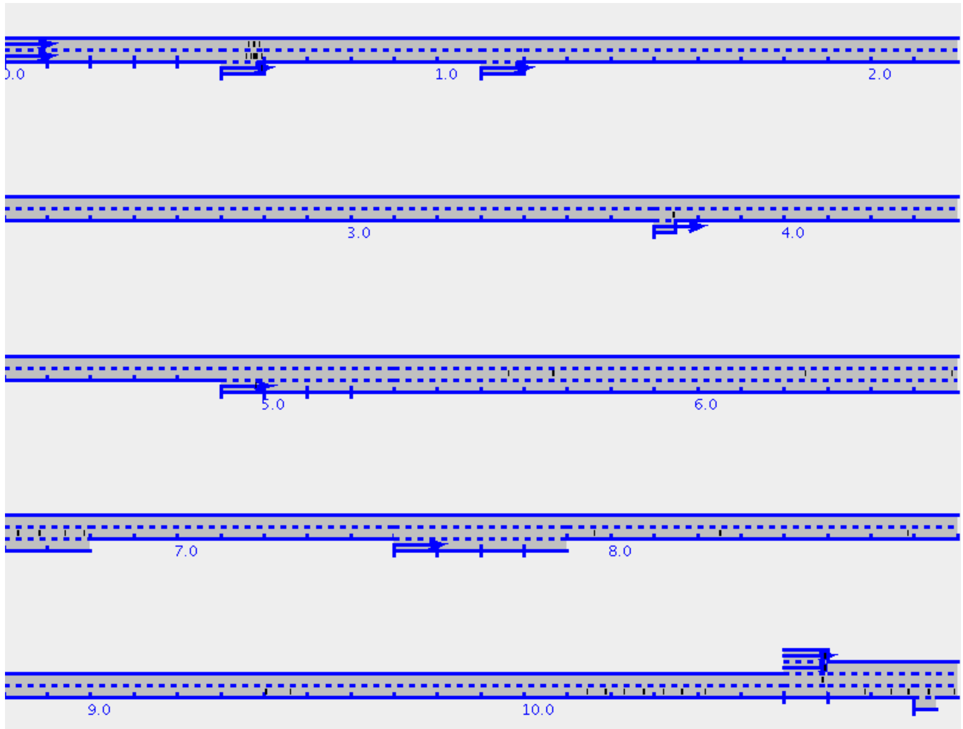


**Figure 8: The stretch of Florida Highway 408 from UCF to Goldenrod Rd, with the entrances, exits and evolution of the lanes accurately represented. The distances are in miles.**

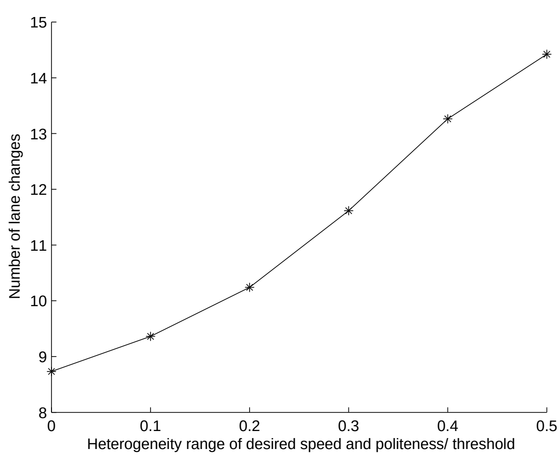


**Figure 10: Lane changes function of the heterogeneity range on highway 408**

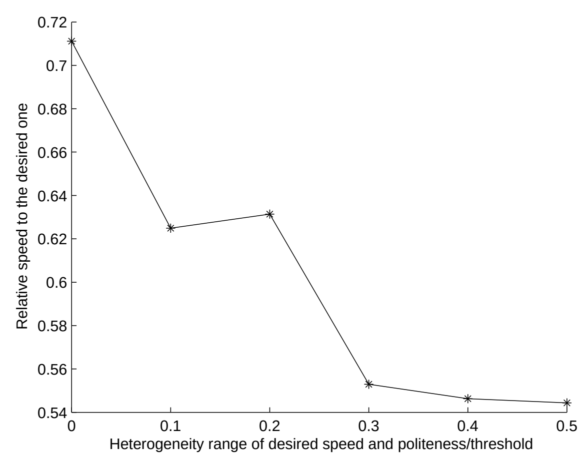


**Figure 11: Relative speed function of the heterogeneity range on highway 408**

[3] S. Maerivoet and B. De Moor. Cellular automata models of road traffic. *Phys.Rep.*, (419):1–64, 2005.

[4] N. Moussa. Dangerous Situations in Two-Lane Traffic Flow Models. *International Journal of Modern Physics C*, 16:1133–1148, 2005.

[5] M. Treiber, A. Hennecke, and D. Helbing. Congested traffic states in empirical observations and microscopic simulations. *Physical Review E*, 62(2):1805–1824, 2000.

[6] M. Treiber, A. Kesting, and D. Helbing. Delays, inaccuracies and anticipation in microscopic traffic models. *Physica A: Statistical Mechanics and its Applications*, 360(1):71–88, 2006.

[7] Expressway history. URL `http://www.expresswayauthority.com/Corporate/aboutStatistics/HistoricalTraffic.aspx`.

# From activity and context toward reaction time: Application to traffic simulation*


Feirouz Ksontini
INRETS
LEPSIS/INRETS - LAMIH FRE CNRS 3304
58 Boulevard Lefebvre, 75732,
75732, Paris 15, France
ksontini@inrets.fr

René Mandiau
Université de Valenciennes et
du Hainaut Cambrésis
LAMIH FRE CNRS 3304
59313 Valenciennes, France
rene.mandiau@univ-valenciennes.fr

Stéphane Espié
INRETS
LEPSIS
58 Boulevard Lefebvre, 75732,
75732, Paris 15, France
espie@inrets.fr

Zahia Guessoum
Université de Paris-VI,
LIP6, DESIR Team
4 place Jussieu, 75252, France
zahia.guessoum@lip6.fr



## ABSTRACT

Multi-agent systems allow the simulation of complex phenomena that cannot be easily described analytically. This approach is often based on coordination of agents whose actions and interactions cause the emergence of the phenomenon to simulate. In this article, we focus on emergent traffic phenomena and in particular we seek to generate the perception-response time of the driver in response to external signals, this under the behavioral traffic simulation model ARCHISIM based on multi-agent modelling. In the literature, and concerning traffic simulation, the reaction time of the drivers does not depend on the context and is calculated from average values observed on the field. We support that the reaction time of the driver is a result of interactions between individuals and the context rather than an input of the model based on average values. We propose an agent model which takes into account the reaction time of the agents and generates individual reaction times relying on the driving task and on the context.


## Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*multiagent systems*

## General Terms

Algorithms, Experimentation.


## Keywords

Traffic simulation, multi-agent simulation, driver behavior, reaction time.


## 1. INTRODUCTION

In recent years, the concepts of Distributed Artificial Intelligence, and more particularly, Multi-Agent Systems (MAS), have brought a new vision to the study and the simulation of distributed systems with focus on interactions of entities that compose them. MAS allow the simulation of complex phenomena that cannot easily be described analytically. They are often based on the coordination of agents and interactions which lead to the emergence of the simulated phenomenon [11]. MAS provides thus perfect solution to the traffic simulation problems, the traffic management, the traffic signal control, etc. ([2], [3], [16]).

Commonly used traffic simulation tools are based on mathematical models involving different statistical laws identified by measurements on the field. However, the mathematical approach and the resulting tools seem limited. Indeed, the obtained laws are generally related to the physical characteristics of the section on which were made the measurements (length, number of lanes, type of markings).

The behavioral approach offers a solution to the weaknesses of mathematical approaches. In the models of the behavioral approaches, phenomena of traffic (congestion, occupancy lanes, etc.) emerge. They are the results defined by individual practices (e.g. heterogeneous behaviours of drivers), interactions and the offer of travel (geometry and structure of the road, regulation, control system, etc.). More precisely, the traffic variables (e.g. the capacity of the road, average speed, reaction time of drivers) are rather a result of the emergence and rely on the context of the driving task. Unlike mathematical models which tend to consider them as inputs of the model identified from real traffic data measured on the field by means of sensors.

In this paper, we use MAS-based behavioral approach to present the road traffic simulation. This approach was developed over the past fifteen years by the French National Institute of Transport and Safety Research (INRETS) in the traffic simulation tool ARCHISIM [10].

We focus on emergent traffic phenomena, and in particular on the human driver perception-response time in response to external signals. In the traffic simulation literature, the reaction time of the human drivers does not rely on the context. It is calculated from values obtained by real observed measures and their standard deviations (variability of behaviours). We consider that the reaction time of the driver is a result of interactions between individuals and the context instead, it is not an input of the model based on aver-

*For use with aamas2010.cls

age values. This idea has been introduced in the behavioral traffic simulation model ARCHISIM based on multi-agent modelling.

This paper is organized as follows. The following section gives an overview of the problem of reaction time in the literature. Section 3 describes a driver agent model taking into account the reaction time. The evaluation of the model and the results are presented in Section 4. Finally, we conclude and give some perspectives to our work.

## 2. DRIVER REACTION TIME IN THE MODELS OF TRAFFIC SIMULATION

As we mentioned earlier, some traffic variables are considered as inputs of mathematical simulation models; while the behavioral models consider that those variables emerge from the simulation and are therefore an output of the model and not an input. After describing generalities of emerging traffic phenomena, the drivers reaction time will be explained in this section.

### 2.1 Generalities

In the psychological studies, the driver reaction time depends on several parameters: perceived data, its propagation mode and conditions, the type of concerned response, the individual, task, workload, fatigue, attention level, etc. ([1], [13], [14]). Due to this large number of variation sources, this problem has often been simplified, limited to certain standardized experimental situations. Generalizations become therefore difficult [15].

One of the mathematical models that took into account the concept of stimulus-response has been initially defined by [4]. The driver receives a stimulus at time $t$ and responds with a lag time corresponding to reaction time. The limitation of this model is that all vehicle-driver pairs are assumed to be identical. They have the same reaction time. This reaction time does not rely on the context of the driving task. Subsequently, several mathematical models of traffic simulation took into account the concept of reaction time of drivers in their modelling. Some studies allocate to reaction time a fixed value which is around 1 second [12], value determined by experimental studies on the human reaction time. Other researchers use a probability law in order to describe the reaction time in a population (for example, [18] uses a truncated log-normal function to describe the reaction time distribution in a population). More generally, the reaction time in all these works has not been considered as an emerging result of the model.

### 2.2 Concepts for modelling reaction time

Based on psychological findings, two concepts seem crucial for modelling reaction time: the workload and the focus of attention.

#### *2.2.1 The workload*

The activity of driving requires activities potentially consuming cognitive resources such as analysis of the situation and decision making. These tasks make thus reference to the relationship between solicitations and the capacity of information processing. They are thus related to the theories of mental workload (in connection with the overloading of the processing capacities).

When driving, the measurement of mental workload can give an indication of cognitive demands for the driver. The interaction between the driving performance, the difficulty level of the driving situation and the resources level needed to address this situation, has also been modeling fine-grained ([5], [6]). According to [5], in region D (D for deactivation) the driver's state is affected (fatigue, medication, stress). So the performance level is mediocre. In region A2 performance is optimal, the driver can easily cope with the task requirements and reach an adequate level of performance. In the regions A1 and A3 performance remains unaffected but the driver has to exert effort to preserve an undisturbed performance level. In region B this is no longer possible and performance declines, while in region C performance is at a minimum level: the driver is overloaded (Figure 1).

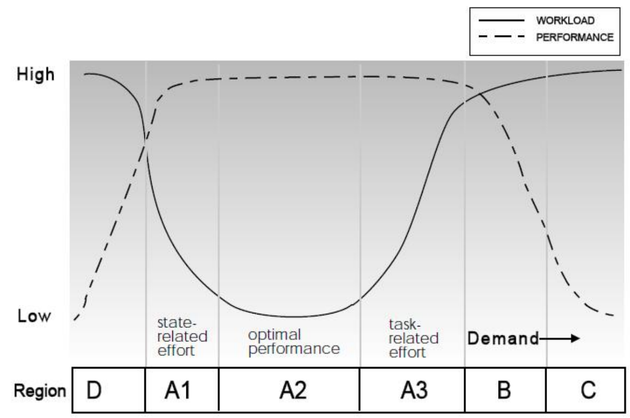


**Figure 1: Relation between workload and driving performance ([5])**

In general, the results show that the reaction time and error rates increase with the complexity of the driving task. These results are important because they highlight that the workload generally increases with the increasing of complexity of driving context (driving in a straight line, intersections, overtaking manoeuvres, etc.).

To summarize, the driver attentional resources are limited, thus under specific driving situations he can be over or under loaded.

#### *2.2.2 Focus of attention*

In the reality, to reason and to make decisions about anticipative future states in which the human driver finds himself, he builds a representation of the situation, based on his goals and his context. Two different situations can be characterized:

- When he is in dynamic situation, he is limited by his capacities of information processing. Time constraints make him focusing his attention on the elements of context that he considers most critical. Following a sequence involving all its capabilities, he has often a "relaxation" step.

- In a situation where the task of driving consumes little cognitive resources, he loses his attention and he falls asleep or develops auxiliary routines.

The amount of resources depends on the requirement level of the task: the execution of a simple and monotonous activity reduces the attentional level while a complex activity leads the driver to maintain a certain level by using various mechanisms (management of resources, focusing, inhibition, etc).

For example, let us note two typical situations:

- *Restarting at a traffic light:* When all drivers behave the same way in a platoon, the time to restart at light is less optimal. Situations obtained in terms of simulation are therefore biased: the road occupation is not realistic and behaviors are homogeneous. In actual situation, behaviors are very heterogeneous, the time latency before restarting at traffic light is very different. The situations observed in reality lie between these extremes and depend largely on societal factors (country, working/non-working days, and so on). The "mathematical" traffic simulations use a reaction time to reproduce the phenomenon. The reaction time can actually be defined by a time related to the resumption of attention and a time associated with a response time of the vehicle "engine".

- *Intersection crossing:* The intersections are situations where interactions among users may be particularly complex. The driver must sometimes manage conflicts with several other drivers. Due to their limitations, they manage priorities and ignore information according to priority criteria. An overload of attention is often followed by a relaxation of the attention. Two concepts are identified in the preceding discourse: the attention and the ability to anticipate and act. At the level of attention, we identify an overload of attention (we have also identified an "under-load" of attention due to a "sub-activity").

As part of this work, we argue that for "plausible" situations at the collective level (road traffic), we must take into account (as much as possible) these human factors (such as limited ability to reason, focus of attention, the attention level) in modelling the behavior of an entity, at individual level (driver). This consideration will affect the own entity representation, of the situation in which it finds itself and its decisions making. The aim is to generate the driver reaction time in response to external signals.

We propose to use the concept of attention to generate the reaction time. The next section describes a preliminary model for calculating a contextual level of attention.

# 3. THE AGENT MODEL

We consider the average of reaction times not using an average of observed response times but using individual reaction times relying on the driving task and on its context.

## 3.1 Attention - Activity levels

By hypothesis, more the driving activity is sustained (mental workload level fairly high), more driver attention level is high (this reduces the reaction time). This remains true in so far as the mental workload level does not exceed the driver treatment capabilities, that is to say, does not exceed the maximum mental workload level. In the latter case, the driver will face a situation of over-load, he will have to focus his attention on the elements of the context that he considers most critical. Due to his limitations (the mental workload level of the individual is limited), he manages priorities and ignores information according to priority criteria (it is supposed that we dispose of a filter which determines the highest priority informations). This case will not be considered in our first modelling approach. If, instead the level of activity decreases significantly, the driver will face a situation of under-load. He will therefore be less and less attentive to the driving task. In this case, reaction time of driver increases.

As we have explained in Section 2.2.2, we propose to use the concept of attention to define the reaction time. We postulate that the attention level varies relying on this cognitive charge of processing. In the driving context, we consider that the level of cognitive charge of processing is simply the activity level of the driving task that depends on the context of the driver. The attention level then varies according to activity level. At the best of our knowledge, the relationship between the attention level and the activity level has not been explicitly defined in the state of art.

$$Attention\ Level = f(Activity\ Level)$$

The driving activity depends not only on the road infrastructure but also on interactions among agents. We make the hypothesis that the activity level of the agent is the aggregation of two parameters:

- $dT$ describes the interactions of an agent in the global flow (MAS)

- $dG$ defines the difficulty to guide the vehicle (lower on motorway, higher in rural roads)

The activity may be defined by the following equation:

$$Activity\ Level = dT + dG$$

For reasons of simplicity, we have chosen this expression form. This is a first approach that can be enriched by taking into account more parameters which can have an impact on the activity level such as the driver experience, his age, driving by night or by day, etc.
The following subsections define $dT$ and $dG$ parameters.

## 3.2 Quantification of $dT$ parameter

To quantify the difficulty of traffic, we analyse the various constraints that a driver must manage. We thus calculate the number of vehicles that make difficult the driver task (Figure 2).

So, we consider that the restrictive vehicles are:

- the vehicle standing in front of the relevant vehicle (A) is still regarded as potentially restrictive (vehicle B)

- the more restrictive vehicle (one who makes slow down the more) in the queue without consider the vehicle ahead (which was already considered)(vehicle C)

- the vehicles on the left and the right lanes who want to join the relevant vehicule lane but only those at greatest risk (those who just want to cut up and forcing him to brake) (vehicles D and E).

- In an intersection, we also consider the traffic coming from other roads.

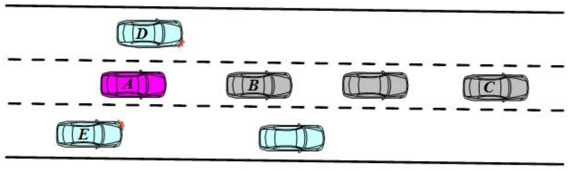


**Figure 2: Determination of binding vehicles**

In our model, the number of these restrictive targets defines the degree of difficulty related to traffic.

$$dT = Number\ of\ binding\ vehicules$$

## 3.3 Quantification of $dG$ parameter

We now study the complexity of guiding the vehicle which has an influence on the activity level of the driving task. The difficulty of guiding the vehicle relies on several factors. In our analysis we consider only difficulties caused by the sinuosity of the road. Indeed, the difficulty of the driving task is not the same in the case of a straight road or in the case of a winding road. The guiding difficulty is then defined by the radii curvature variation of the road at a distance of vision (which depends on the vehicle speed). This is a first approach that can be enriched by taking into account, for example, the quality of the road (lane width, grades of "sides").

$$dG = Number\ of\ radii\ curvature\ variations$$

The activity level is therefore given by the following expression:

$$\begin{aligned} Activity\,Level &= Number\,of\,binding\,vehicules \\ &+ Number\,of\,radii\,curvature\,variations \end{aligned}$$

## 3.4 Global formulations

### 3.4.1 Attention level

As we mentioned earlier, the attention level is a function of activity level. However, the attention level also depends on other parameters. We postulate that the attention level is composed of:

- the initial attention level at instant $t = 0$
- earlier attention level (at the previous step, instant $t-1$) multiplied by a ratio which depends on the agent speed and the desired speed (constant during the simulation for each agent ) (The desired speed is the speed at which the agent would drive)
- the activity level of the driving task at $t$
- The pressure variable which is a random variable (constant during the simulation for each agent) that indicates whether the driver is under pressure or not. If he is under pressure, he would be more attentive to the driving task. For example, at a stop light he would start as soon as possible. Therefore his reaction to the traffic light will be faster than someone who is not in a hurry.

We propose the following definition:

$$\begin{aligned} (Attention\ Level)_t &= (Attention\ Level)_{t_0} \\ &+\alpha * (Attention\ Level)_{t-1} * \frac{Speed_t}{Desired\ Speed} \\ &+\beta * (Activity\ Level)_t \\ &+Pressure \end{aligned}$$

The parameters $\alpha$ and $\beta$ enable to calibrate the attention expression since all parameters are not of the same order of magnitude. Let us note also that the attention level depends firstly on the driver speed (more the speed is high more the driver is attentive to his driving task) and secondly depends on the driving task (defined by the activity level).

Furthermore, we find that the driver has a minimum activity level that allows him to stay awake and a maximum activity level before being overloaded. The minimum and maximum limits are different from one individual to another (we do not all have the same attention capacity and resistance to sleep).

### 3.4.2 Reaction time

We now study the impact of this attention level on the behavior of the driver through the reaction time expression. We find that the reaction time evolves inversely with the attention level (the more we are attentive, more reaction time decreases and vice versa). We consider a reaction time expression by stage according to attention level. When the level of attention is relatively low (less than or equal to the minimum level), the driver is increasingly inattentive until actually fall asleep. In this case, the reaction time is very high. When the attention level is above the minimum limit, the reaction time is an inverse function of attention. Let us note that this definition is empirical and may express "a quantification" of the reaction time.

$$Reaction\,Time = \begin{cases} t + \frac{min\,att\,level - attention}{100}, & \text{if attention} \leq \text{min att level} \\ \frac{a}{attention} + b, & \text{else} \end{cases}$$

*with $t$ and $a$ constants that are used to calibrate the response time expression and $b$ is the reaction time of vehicle "engine".*

We aim to demonstrate that by using these concepts and calibrating the parameters we will obtain on average the same results as the observed reaction times. This model is taking into account the context.

# 4. TOOL AND EXPERIMENTS

## 4.1 Simulation traffic tool: Archisim

In France, INRETS (acronymous for French Institute for Transport and Safety Research) has been researching into road traffic simulation based on the driving behaviour of human drivers since the end of the 1980s. The simulation tool, ARCHISIM, has its origin in research into driving psychology. It can be considered as a virtual reality tool in which a human driver can interact within an environment of autonomous vehicles. In ARCHISIM, the traffic is the result of the individual actions and the developing interactions between the different actors present in a road situation([7], [8], [9]). The objective aims at making ARCHISIM an open tool for the study of the "traffic system". Moreover, ARCHISIM has been developed such that the traffic model can host a driving simulator. In this case, the person in the driving simulator interacts with the traffic within the simulation model.

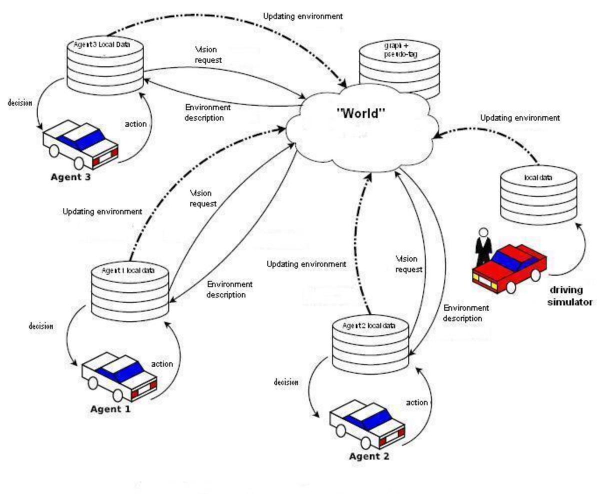


**Figure 3: Global architecture of ARCHISIM**

The principal advantage of this behavioural model is that simulation conditions can be dynamically modified (the degree of visibility which results from the weather, the driving preferences of the human driver, the characteristics of the autonomous vehicle - cars, lorries, buses, pedestrians etc.) as can the road equipment (traffic signals, traffic signs, etc.). ARCHISIM is a simulation model and its implementation is based on the principles of multi-agent systems. Each autonomous vehicle (AV) is considered as an agent. It therefore possesses a model of its environment and interacts with the other agents, possibly including vehicles with a human drivers (driving simulators). A given road traffic situation is both a heterogeneous system and an open system (the number of AVs can vary), in which drivers or autonomous vehicles interact, each having their own objectives. The environment is non-deterministic and the system may have an infinite number of states. The information which is perceived by agents is geographically limited and incomplete.

## 4.2 Evaluation and Discussion

In ARCHISIM, a traffic situation results from the interaction of this behavior with the road environment. Such a model must be evaluated at two levels: (i) microscopic level, (ii) macroscopic or global level. The experiments at the individual level verify that the individual driver behavior seems similar to the expected behavior. This experiment will also consider the visual coherence of simulated vehicle behavior. A visual realism is needed to conduct such studies. Furthermore, it is useful to study the macroscopic level only if the result is visually convincing.

The situation of the highway allows us to verify our assumptions about the attention level variation. This variation depends on the agent context (the impact of the driving task difficulty). Restarting at the traffic light allows to underline the impact of the behavior heterogeneity (due to different attention levels and minimum/maximum attention limits that differ from one individual to another) on the latency time. We consider a set of vehicles on two lanes road and a continuous traffic demand generated by a Gaussian distribution, we place on this road a stop light and a sensor after this light.

Figure 4 gives the results of the attention level and the reaction time variations depending on the activity level (horizontal axis describes the activity level variation during the simulation). We show that when the activity level of the agent increases (i.e. the complexity of the driving task is more important), the attention level also increases and the reaction time decreases. The driver agent is increasingly attentive to the driving task because its activity is quite sustained, therefore, it reacts quickly to stimuli in his environment.

If the attention level decreases significantly (among others because his activity level decreases), the agent is less and less attentive to the driving task and he has not a good perception of the environment. Therefore, his reaction time will raise and sometimes, he may even cause an accident or he may exit the road.

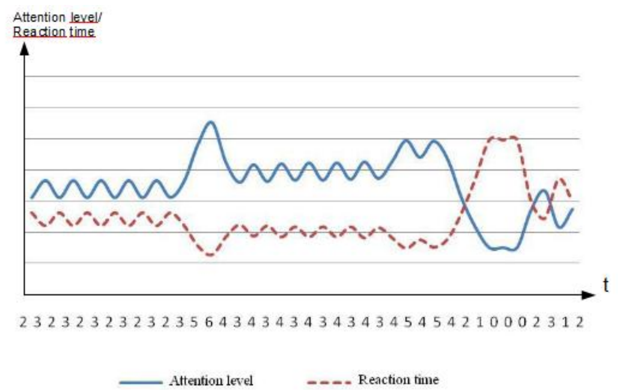


**Figure 4: Attention level and reaction time variations depending on activity level**

In general, the introduction of the attention concept in the agent behavior modeling, may allow to consider new situations in the simulation (where a driver is completely inattentive to the light and does not react when it switches to green). Such a situation can have a global impact on all traffic. To assess the overall level, we need a certain number of traffic informations collected from the simulation. To acquire these data, a number of sensors must be placed on the network. Each sensor fills in a specific file during the simulation. We obtain, as result of this implementation, a file per sensor. ARCHISIM has a number of tools to exploit data from "sensors" files generated during the simulation. This operation is done through aggregating "sensor" data according to temporal and/or space perspectives.

We notice at global level, that taking into account the concept of attention in the agent modelling increases the total duration of the concerned circuit portion trip and reduces the average speed of the trip on this same portion (see figure 5, for a two minutes simulation).

This means that when some agents are inattentive, their reaction time will be longer than if they were completely attentive to the driving task. Therefore, we expect that it

| | Attentive Agents | Agents can be inattentive |
|---|---|---|
| Circuit transit time | 00:01:08 | 00:01:17 |
| Transit average speed (km/h) | 101,7 | 90,5 |

**Figure 5: Attentive vs inattentive agent**

also has an effect on traffic flow.

For evaluating the traffic fluidity, we use an index which is obtained by dividing the reference time[1] by the transit time. Indeed, comparing the fluidity indexes during a six minutes simulation, we notice that traffic is less fluid when we take into account the driver reaction time in the simulation (Figure 6).

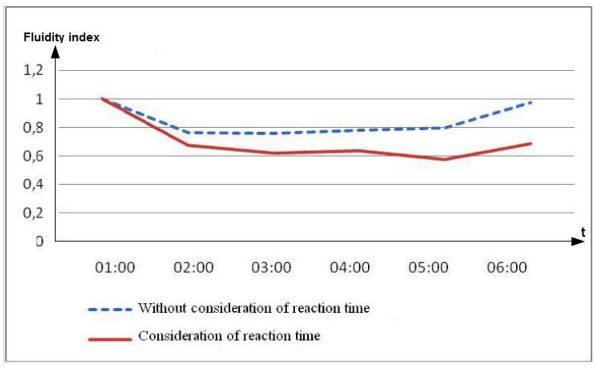


**Figure 6: Comparison of fluidity indexes**

Let's notice that an assessment of the intersection case would be interesting. It would verify that the model is relevant in this case which can be complex. Moreover, in order to perform a calibration and validation of the model, we must look for real situations for which accurate and complete data concerning the characteristics of traffic have already been measured.

## 5. CONCLUSION AND PERSPECTIVES

This work is part of the traffic simulation and especially in the traffic model ARCHISIM. This model is based on the results of psychological research on driver behavior and uses a multi-agent architecture. It uses a new traffic simulation approach, called behavioral approach. In ARCHISIM, a traffic situation is considered as the result of interactions of driver behavior with other road users, and road infrastructure and regulation. Each actor has an autonomy and his own knowledge and his own goals and his own strategy to perform his various tasks and to resolve any conflict that he might face. The decision making of each agent is individual and relies on a perception of his environment.

We think that the preliminary works presented in this paper will give a better "realism" of driver behavior. The simulation of behavior has attempted to take into account the information present in their environment (limited capacity of perception, attention level, etc.) and the processing of this information (mental workload, multiple and competing tasks, etc.). Specifically, we used the concept of attention to produce the reaction time. This concept is more contextual and can yield reaction times related to the context of the driving situation. This consideration affects the agent representation, of the situation in which he finds himself and his decision making. We have introduced a model that can actually have a response time not using an average of observed response times but using the individual reaction times relying on the context. The objective is to validate the implemented mechanisms and to observe if they can simulate the behavior observed in reality (it consists only in trends).

The implementation of our contribution has enabled us to observe situations that are close to reality. We saw that the consideration of the attention concept in agent behavior modeling has an impact on traffic flow. This is explained by higher reaction times for drivers who are less attentive to the driving task. The studied situations were limited to a highway with traffic situation and to restart at light signalization. This latter example helped to highlight the observed phenomena as it is in the stop light that drivers are more likely to become less attentive to the road.

Our model shows interesting trends. However the case of crossing intersection still has to be tested. The intersections are situations where interactions among users may be particularly complex and therefore their management requires a lot of attentional resources.

Moreover, for the calibration and validation of the model, it will be useful to consider real-life situations where complete data concerning the characteristics of traffic have already been measured. The simulation data can later be compared to data of these real-life situations.

[1]The reference time is the time it would take a vehicle to travel the same distance with an average speed equal to the desired speed.

## 6. REFERENCES

[1] H. Alm and L. Nilsson. The effects of a mobile telephone task on driver behavior in a car following situation. *Accident Analysis and Prevention*, 27:707–715, 1995.

[2] A. L. C. Bazzan. A distributed approach for coordination of traffic signal agents. *Auton Agent Multi-Ag*, 10:131–164, 2005.

[3] B. Burmeister, J. Doormann, and G. Matylis. Agent-oriented traffic simulation. *Trans Soc Comput Simul*, 14:79–86, 1997.

[4] R. E. Chandler, R. Herman, and E. W. Montroll. Traffic dynamics : studies in car following. *Operations research*, 6:165–184, 1958.

[5] D. DeWaard. *The measurement of drivers' mental workload.* PhD thesis, University of Groningen, Groningen, 1996.

[6] D. DeWaard and K. Brookhuis. On the measurement of driver workload. *In: Rothengatter, T., Carbonell Vaya, E. (Eds.), Traffic and Transport Psychology: Theory and Application. Pergamon, Amsterdam*, pages 161–171, 1997.

[7] A. Doniec, S. Espié, R. Mandiau, and S. Piechowiak. Multi-agent coordination and anticipation model to design a road traffic simulation tool. In *EUMAS'06, Proceedings of the fourth European Workshop on Multi-Agent Systems, Lisbon*, 2006.

[8] A. Doniec, R. Mandiau, S. Espié, and S. Piechowiak. Dealing with multi-agent coordination by anticipation: Application to the traffic simulation at junctions. In

*EUMAS'05, Proceedings of the Third European Workshop on Multi-Agent Systems*, 2005.

[9] A. Doniec, R. Mandiau, S. Piechowiak, and S. Espié. Anticipation based on constraint processing in a multi-agent context. *Journal of Autonomous Agents and Multi-Agent Systems (JAAMAS)*, 17:339–361, 2008.

[10] S. Espié. Archisim, multi-actor parallel architecture for traffic simulation. In *proceeding of the Second World Congress on Intelligent Transport Systems, volume IV, Yokohama, Japan*, 1995.

[11] Z. Guessoum and R. Mandiau. Modèles multi-agents pour des environnements complexes. *Numéro spécial de la Revue Française d'Intelligence Artificielle (RIA)*, novembre 2008.

[12] A. Kesting, M. Treiber, and D. Helbing. *Multi-Agent Systems: Simulation and Applications*, chapter Agents for Traffic Simulation, pages 325–356. 2009.

[13] D. Lamble, T. Kauranen, M. Laakso, and H. Summala. Cognitive load and detection thresholds in car following situations: safety implications for using mobile (cellular) telephones while driving. *Accid Anal Prev*, 31(6):617–623, 1999.

[14] J. D. Lee and D. L. Strayer. Preface to the special section on driver distraction. *Human Factors*, 46:583–586, 2004.

[15] G. Malaterre. Temps de réponse et manoeuvre d'urgence. In *INRETS-LPC Laboratoire de Psychologie de la conduite*. Décembre 1986.

[16] R. Mandiau, A. Champion, J.-M. Auberlet, S. Espié, and C. Kolski. Behaviour based on decision matrices for a coordination between agents in urban traffic simulation. *Applied Intelligence*, 28:121–138, 2008.

[17] H. D. Parunak, R. Savit, and al. Agent based modelling vs equation based modelling: a case study and user's guide. *In J. Sichman, R. Conte and N. Gilbert, editors, Multi agent systems and agent based simulation, Lecture Notes in Computer Science 1534. Springer*, 1998.

[18] T. Toldeo, H. Koutsopoulos, and M. Ben-Akiva. Integrated driving behavior modelling. *Transportation Research Part C: Emerging Technologies*, 15, Issue 2:96–112, April 2007.

# Employing Agents to Improve the Security of International Maritime Transport

Michal Jakob, Ondřej Vaněk, Štěpán Urban, Petr Benda and Michal Pěchouček
Agent Technology Center, Dept. of Cybernetics, FEE Czech Technical University
Technická 2, 16627 Praha 6, Czech Republic
{jakob, vanek, urban, benda, pechoucek}@agents.felk.cvut.cz

## ABSTRACT

We explore how agent-based techniques can be employed to reduce the threat of contemporary maritime piracy to international transport. At the center of our approach is a data-driven agent-based simulation platform incorporating a range of real-world data sources in order to provide a solid computational model of maritime activity. The platform is integrated with extension modules providing advanced analysis, reasoning and planning capabilities. Two such modules are presented. The first module applies statistical machine learning techniques to extract models of vessel movement from trajectory data; the models are subsequently used for categorizing vessels and detecting suspicious activity. The second module employs game theory-based strategic reasoning to plan risk-minimizing routes for vessels transiting known pirate waters. Empirical evaluation performed on the data-driven simulation shows promising potential of agent-based methods for reducing the security risks and economic costs of illegal maritime activities.

## Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*multiagent systems*

## General Terms

Algorithms, Security, Economics

## Keywords

transport security, maritime crime, agent-based simulation, trajectory classification, route planning, game theory

## 1. INTRODUCTION

The recent surge in maritime piracy presents a serious threat to the international maritime transport system. Over the past years, insurance rates have increased more than 10-fold for vessels transiting known pirate waters and the overall costs of piracy in the Pacific and Indian ocean alone were estimated at US$ 15 billion in 2006 and continue to rise [14]. Various methods are explored for putting piracy back under control and for mitigating the risks it entails.

In this paper, we explore how the agent-based approach can contribute to addressing this pressing issue. Specifically, we describe a testbed developed for prototyping and evaluating (multi-)agent-based techniques for understanding, detecting, anticipating and eventually supressing piracy and possibly other categories of maritime crime. At the center of the testbed (see Section 2) is an agent-based simulation (see Section 3) which integrates a wide range of real-world data in order to provide a solid computational model of maritime activity. The simulation provides an interface through which it can be integrated with advanced agent-based techniques.

At the moment, two categories of such techniques are investigated. The first category comprises analytical techniques for gaining insight into the structure and dynamics of maritime activities with the view of utilizing this insight in further decision making. A prime representative of this category is a method for probabilistic modeling and classification of vessel trajectories (see Section 4). The second category comprises planning and coordination techniques through which maritime traffic can be (re-)organized in order to eliminate or reduce the negative impact of illegal activities. A representative of this category is a game-theoretic method for planning risk-minimizing routes for vessels transiting known pirate waters (see Section 5).

Although agent-based techniques have been successfully applied in other traffic and transportation domains and problems (see e.g. [7], [15]), this is – to our best knowledge – the first integrated attempt at employing agent-based concepts and techniques in the domain of maritime transport security. Further discussion of the existing work and our progress beyond it can be found in the sections dedicated to individual aspects of the problem.

## 2. MARITIME TRAFFIC SOFTWARE TESTBED

The agent-based maritime traffic testbed [10] has been designed to allow incorporation of various data sources and different counter-piracy methods and to support systematic experimentation with these methods under varied conditions, both on synthetic and real-world data. More information about the testbed can be found in [9] and at the project website[1], which also features a brief video overview of testbed's capabilities.

### 2.1 Testbed Architecture

The main objective in designing the testbed was modularity and extensibility; this objective has been met by employing a loosely coupled architecture with clearly defined data and control flows. The components of the testbed can be arranged into the following layers:

- **Data-driven maritime traffic simulation layer**

[1] `http://agents.felk.cvut.cz/projects/agentc/`

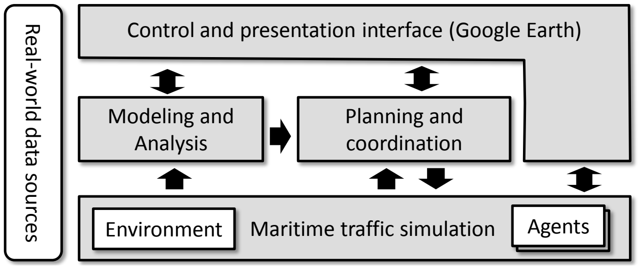


Figure 1: Layered architecture of the AgentC testbed. Information flows between individual components are depicted.

stands at the core of the testbed. It contains parts responsible for the representation and operation of the simulation model, both the simulated vessels and the simulated environment in which they operate.

- **Analysis layer** contains algorithms performing analysis (such as vessel classification and future route prediction) on the data coming from the simulation.
- **Planning and coordination layer** contains algorithms responsible for more complex coordination and planning beyond the basic vessel behaviors implemented as part of the simulation.
- **Control and presentation layer** is responsible for presenting the output of the simulation and the analytical modules; these data are synthesized for unified visualization using the KML format. The presentation frontend is described in Section 3.5.

## 2.2 Data Sources

As an essential feature, the testbed incorporates several categories of real-world data and allows for their integrated analysis, specifically:

- **Geographical Data (general)** comprise general information about the geography of the environment, in particular shore lines, ports and shallow waters. These data are supplied directly by Google (Earth); they are used primarily for general vessel navigation (see Sec. 3.4) and for providing background geographical context in the user frontend.
- **Geographical Data (operational)** comprise geographical information specific to the operation of simulated vessel types, in particular the location of main piracy hubs[2], piracy zones, fishing zones and transit corridors. The operational geographical data govern the operation of individual vessel categories (see Section 3.3).
- **Vessel Attributes** describe vessel operational attributes such as vessel type, length, tonnage, max speed etc. These data are extracted from vessel tracking servers, e.g. AISLive[3] and are used to provide realistic parameters for simulated vessels.
- **Vessel Motion Data** comprise records of vessel trajectories. Vessel tracking servers provide recorded AIS[4] traces that are sampled with different spatio-temporal resolution (typically every 20 minutes on average with 0.1 degree precision). These traces are filtered and used for the modeling of the transport vessel movement and route planning.
- **Behavioral/Activity Data** comprise higher-level information about maritime activity. These data are typically provided by organizations observing the situation in relevant regions. The Maritime Security Centre, Horn of Africa (MSCHOA)[5] provides up-to-date information on piracy incidents and alerts in the Gulf of Aden and off the Somali coast, including guidelines on how to proceed when traversing these areas. These data are used for pirate behavior modeling as well as for the route planning of transport vessels.

  UNOSAT[6] publishes technical posters analyzing the situation around Somalia and providing summary analytical information, such as spatial distribution of pirate attacks and its relation to established transit corridors. These data are used primarily for validating the simulation model by comparing the global, macroscopic properties of the generated traffic with those published by UNOSAT.

# 3. MARITIME TRAFFIC SIMULATION

Agent-based simulation is a fundamental component of our approach. It provides a controlled environment for experimenting with the developed techniques and also helps to overcome the lack of hard real-world data in certain areas (in particular good-quality traces of illegal vessels).

Although naval simulation have been long used for military purposes, simulations of civilian maritime traffic are hard to find. The MATRICS [3] projects, modeling the behavior of transport ships near the shore of Canada, seems to be the only case; however, the model used in MATRICS is based on fluid mechanics and has therefore difficulties capturing the inter-vessel relations and non-linear features of the traffic model. We are not aware of any civilian maritime traffic simulation employing the agent-based approach.

## 3.1 Simulation Process

The simulation is synchronous and proceeds in discrete time steps. The simulation step size can be modified on-the-fly through the user interface. The amount of wall-clock time[7] spent in each step can be also controlled in order to leave enough computational time to analysis and planning algorithms running on top of the simulation. The synchronous nature of the simulation together with controlled pseudo-random number generators guarantees determinism and thus repeatability of the simulation. It also simplifies debugging and consequently increases the speed with which new techniques can be prototyped and tested.

## 3.2 Environment

The Environment module represents the simulated model of the real-world environment. It holds and provides infor-

[2] `http://bbs.keyhole.com/ubb/ubbthreads.php?ubb=showflat&Number=1242871&site_id=1`
[3] `http://www.aislive.com/`
[4] AIS (*Automatic Identification System*) is a short range tracking system used for identifying and location vessels.
[5] `http://www.mschoa.eu/`
[6] `http://unosat.web.cern.ch/unosat`
[7] *Wall clock time* is the human perception of the passage of time from the start to the completion of a task, i.e. the elapsed real-world time as determined by a chronometer such as a wristwatch or wall clock.

mation about the state of the simulated environment and executes actions performed by agents situated therein.

The state data in the environment model include both geographical and vessel-related data. The geographical data, general as well as operational, are made available to all modules. Moreover, the vessel-related data (vessel attributes and motion data) are made available to the Analyzer module for analysis and the Presentation Interface for visualization. The update of vessel position data proceeds as follows: the vessel agents modify the environment state by controlling the vessels and modifying their state and position. As the simulation runs and agents execute their plans, the vessel position is periodically updated. The information about the vessel positions is recorded and provided for analysis and visualization.

The current simulation time (i.e. time within the simulated world – as opposed to the wall-clock time) is stored in the Environment module and can be accessed by any of the modules or agents. This way e.g. day/night cycles can affect the operation of a pirate ship and the planning of its movement (shorter vision range at night, move off the port in the morning etc.)

## 3.3 Vessel Agents

Every agent controls one or more vessels. The plans for each vessel are either created prior to the simulation (e.g. for transport vessels) or, typically, generated dynamically during the simulation run (e.g. for pirate vessels).

### 3.3.1 Vessel Types

The platform can simulate the simultaneous activity of a large number (thousands) of the following categories of vessels:

- **Long-range transport vessels** are large- to very large-size vessels transporting cargo over long distances (typically intercontinental); these are the vessels that are most often targeted by pirates.
- **Short-range transport vessels** are small- to medium-size vessels which transport passengers or cargo close to the shore or across the Gulf of Aden.
- **Fishing vessels** are small- to medium-size vessels which perform fishing within designated fishing zones; fishing vessels launch from their home harbors and return back after the fishing is completed.
- **Pirate vessels** are small- to medium-size vessels operating within designated *piracy zones* and seeking to attack a long-range transport vessel. The pirate control module supports several strategies some of which can employ multiple vessels.

Table 1 summarizes the main parameters of each type of vessel agents. Note that in general, a vessel agent can control more than one vessel (e.g. a pirate vessel agent controlling a hijacked transport ship).

The behavioral models for individual categories of vessels have been manually synthesized based on the information about real strategies obtained from several sources including IMB Piracy Reporting Centre[8] and Maritime Terrorism Research Center[9].

[8] `http://www.icc-ccs.org/index.php?option=com_content&view=article&id=30`

[9] `http://www.maritimeterrorism.com/`

| Vessel type | Parameters |
|---|---|
| Long-Range Transport Vessel | Start destination, Goal destination |
| Short-Range Transport Vessel | Start destination, Goal destination |
| Fishing Vessel | Home port, Fishing Zone |
| Pirate Vessel | Home port, Area of control, (targeted ship) |

Table 1: Main parameters of different types of vessel agents.

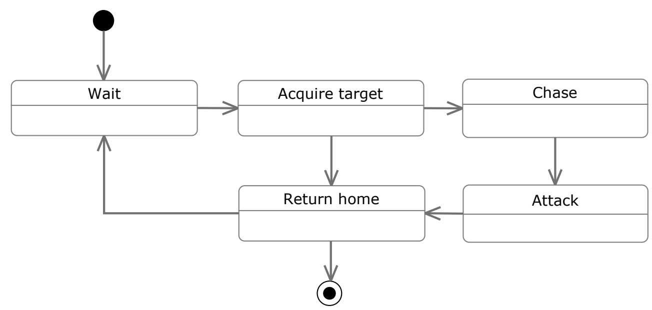


Figure 2: Finite state machine of the pirate vessel agent.

### 3.3.2 Executable Behavior Representation

Vessel agent behavior is implemented using finite state machines (FSM). Agent FSMs consist of states that represent agent's principal mental states. Transitions between the states are defined by unconditional or by conditional transitions conditioned by external events. Implementation-wise, the simulator allocates a time slice to the agent and the agent delegates the quantum to the FSM. The current state may either use the whole time slice and stay in the current state, or it can utilize only part of the time slice and delegate the rest of the time to a following state or states. An example of a pirate FSM is depicted on Figure 2.

## 3.4 Vessel Route Planning

A modular route planning architecture has been developed allowing to combine general shortest-route point-to-point planners with specialized planners for specific areas.

### 3.4.1 General Shortest-Route Navigation

The basic route planner finds a shortest route between two locations on Earth's surface considering vessel operational characteristics and environmental constraints, including minimum allowed distance from shore, which can differ between regions. The planner is based on the A* algorithm adapted for spherical environment and polygonal obstacles. To make the route search tractable, a navigation graph is first constructed (see Figure 3 for a simple example). More details about the operation of the route planner can be found in [9].

### 3.4.2 Gulf of Aden Transit Planner

Factors other than route length need to be considered when planning the route through the Gulf of Aden. Two specialized planners have been therefore developed for this purpose.

The simple corridor planner navigates the ship through the International Recommended Transport Corridor and mim-

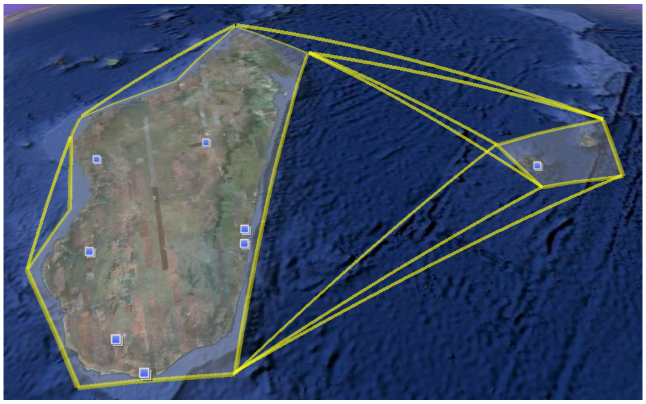

Figure 3: An example of a simplified navigation graph.

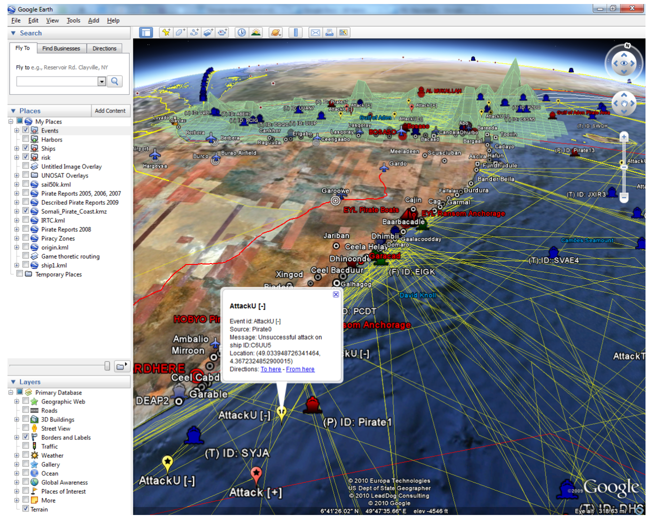


Figure 4: Google Earth-based frontend showing vessels, their past trajectories and a dynamic incident risk map over the Gulf of Aden.

ics the current practice in the area. As a possible future alternative, we have also implemented a risk-minimizing route planner based on game theory (see Section 5 for the description of the approach).

## 3.5 User Front-end

The user frontend provides both the geo-based visualization of the various outputs provided by the testbed and the interactive user control of the simulation. For the former, Google Earth, a KML[10] capable viewer as well as data provider is used; for the latter, standard Java-based graphical user interface are created.

Google Earth-based frontend allows to interactively visualize the various outputs of the testbed, both static data described in Section 2.2 and dynamically generated output of the advanced analysis and planning modules. A screenshot of the frontend is given in Figure 4.

In addition to ergonomic navigation and 3d camera control, the main advantage of the frontend is the ability to present structured data on varying level of detail. The layer-based interface allows to select different layers of information and compose an information picture with the aspects and the level of detail fit for the specific user's need. To leverage the layer-based concept, we organize the testbed output into multiple information layers. The simulation itself provides a number of layers; each of the analysis and planning tools then also adds its own layer.

The integration with Google Earth is provided via dynamically constituted KML files served by an HTTP server running inside the platform. The KML files are read into the Google Earth application using its HTTP data link feature and automatically refreshed. This way, dynamic data can be displayed (such as a moving vessel), though for performance reasons, the refresh rate is limited to about once a second. Because of this and other limitations (e.g. in interaction with the user), we may in the future migrate the frontend to the Java-based NASA WorldWind platform[11].

# 4. PROBABILISTIC TRAJECTORY MODELING

Good understanding of the behavior of different types of vessels is instrumental in deploying security measures efficiently. To this aim, we have developed a method for learning typical motion patterns for individual categories of vessels based on sample trajectory data. The method combines probabilistic spatial and temporal modeling in order to construct classification models that can be used for categorizing unknown and for detecting anomalous, and thus possibly illegally behaving vessels. Here we provide only a brief overview of the method; more detailed discussion can be found in [20].

## 4.1 Two-level Trajectory Model

The proposed method works on two levels. On the first level, spatial properties of the traffic are represented using a Gaussian mixture model (GMM). On the second level, temporal/sequential aspects are captured using a hidden Markov model (HMM).

The input to the algorithm is a set of trajectories $\mathcal{T} = \{T_1, \ldots, T_m\}$. Each trajectory $T = (\boldsymbol{x_1}, \ldots, \boldsymbol{x_n})$ is a tuple representing a sequence of points $\boldsymbol{x} = (x_{lat}, x_{long}) \in \mathbb{R}^2$ on the Earth's surface expressed in GPS coordinates. The output of the algorithm is a Gaussian mixture model (Level 1) and a hidden Markov model (Level 2) best approximating the trajectories.

### *4.1.1 Level 1: Modeling Spatial Distribution*

On the first level, the method uses the expectation minimization algorithm [4] to build a Gaussian mixture model of the spatial density of the maritime traffic. The algorithm disregards the sequential aspect of the trajectories and treats them as unordered sets of vessel positions. It then approximates the empirical distribution of vessels using a mixture of 2d Gaussian kernels[12]

[10]Keyhole Markup Language (KML) is an XML-based language schema for expressing geographic annotation and visualization

[11]`http://worldwind.arc.nasa.gov/`

[12]This process can be viewed as clustering the very large number of vessel positions into much lower number of spatial clusters which are the used as the basis for temporal modeling

More specifically, each Gaussian kernel

$$\phi(\boldsymbol{x}|\boldsymbol{\mu},\boldsymbol{\Sigma}) = \frac{1}{2\pi\sqrt{|\boldsymbol{\Sigma}|}} \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \sum^{-1} (\boldsymbol{x}-\boldsymbol{\mu})\right) \quad (1)$$

is parameterized by its mean $\boldsymbol{\mu} \in \mathbb{R}^2$ (the position of Gaussian kernel's center) and its 2-by-2 covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{2,2}$. Assuming the mixture consists of $m$ components, the algorithm is looking for a tuple of parameters

$$\Theta = (\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1, w_1, \ldots, \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m, w_m)$$

so that the mixture model

$$\Phi(\boldsymbol{x}|\Theta) = \sum_{i=1}^{m} w_i \phi(\boldsymbol{x}|\boldsymbol{\mu_i}, \boldsymbol{\Sigma_i}) \quad (2)$$

best approximates the spatial distribution of the vessel positions in the trajectory set $\mathcal{T}$. The expectation maximization algorithm [4] is used to find maximum likelihood estimates of parameters $\Theta^*$, i.e.,

$$\Theta^* = arg \max_{\Theta} \Phi(\mathcal{T}|\Theta) = = arg \max_{\Theta} \prod_{T=(\boldsymbol{x_1},\ldots,\boldsymbol{x_n})\in\mathcal{T}} \prod_{i=1}^{n} \Phi(x_i|\Theta) \quad (3)$$

for a given set of trajectories $\mathcal{T}$. An example of such a set is given in Figure 5.

The set of Gaussian kernels obtained is used as a basis for expressing vessel trajectories in the second level of the algorithm.

#### 4.1.2 Level 2: Modeling Temporal Dependencies

The second level captures the temporal structure of the trajectories. Vessel trajectories are expressed as sequences of the Gaussian components of the mixture model; the sequences are subsequently used to train a hidden Markov model.

Specifically, assume we have a Gaussian component model $\Phi$ (equation (2)) and a trajectory $T = (\boldsymbol{x_1}, \ldots, \boldsymbol{x_n})$. For each point $\boldsymbol{x}$, the algorithm looks for a Gaussian kernel component to which the point belongs

$$i^* = \arg \max_{1\le i\le m} \phi(\boldsymbol{x}|\boldsymbol{\mu_i}, \boldsymbol{\Sigma_i}) \quad (4)$$

By applying the this equation to all points, we obtain a sequence $Q = (i_1, \ldots, i_n)$ expressing the original sequence of locations as a sequence of (the indices of) components of the GMM; we term such a sequence a *GMM-based trajectory*. After applying the above to all trajectories $\mathcal{T}$, we obtain a set of GMM-based trajectories $\mathcal{Q}$.

In the next step, a hidden Markov model is sought which best fits the sequences in $\mathcal{Q}$. We assume that the states of the model are observable and directly correspond to the components of the GMM-based model. The *Baum-Welch algorithm*[1] is then applied to learn a set $\mathcal{P}$ of transition probabilities $P(i_j|i_k)$ specifying the probability that a vessel moves from an geographical region corresponding to component $j$ to a region corresponding to component $k$. An example of such a model is given in Figure 5.

Once a hidden Markov model is obtained, it can be used for evaluating the closeness of specific trajectories. Also in this case, the classified trajectory $T = (\boldsymbol{x_1}, \ldots, \boldsymbol{x_n})$ is first converted into its GMM-based representation $Q = (i_1, \ldots, i_n)$

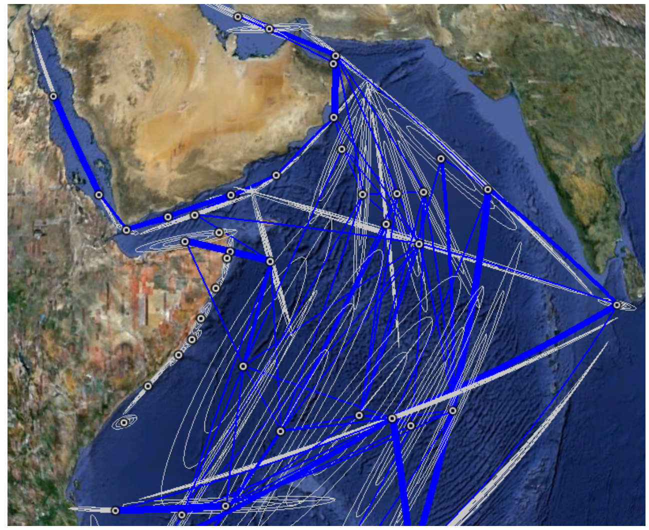

Figure 5: Probabilistic trajectory model of long-range transport. The white elipsoids represent Gaussian kernels in the Level-1 GMM; blue overlay graph represents the Level-2 hidden Markov model. The thickness of the edges corresponds to the transition probability between the two kernels, i.e., between their corresponding geographical regions.

using (4). The probability of the sequence $Q$ being produced by a model $\mathcal{P}$ is then calculated as

$$p(Q|\mathcal{P}) = \prod_{j=2}^{n} P(i_j|i_{j-1}) \quad (5)$$

Computational requirements of HMM learning grows exponentially with the number of states in the model, i.e., with the number of Gaussian kernels in the Level 1 model.

### 4.2 Classification Modes

The two-level model can be employed in two modes (1) for categorizing traffic into a predefined set of classes, and (2) for identifying anomalous and thus possibly illegitimate traffic.

#### 4.2.1 Traffic Categorization

The traffic categorization mode assumes there are labeled trajectory sets available for all categories of traffic under consideration. On Level 1, all trajectories are used for creating a single spatial model of the traffic; this is shared across all categories. On Level 2, an individual HMM for each category is created.

When an unknown vessel trace is to be classified, it is evaluated for closeness against all HMMs (using equation (5)) and the category of the closest HMM is used as the category of the vessel.

We refer to the classifier with the above described structure and operation as the *vessel categorizer*.

#### 4.2.2 Illegitimate Traffic Detection

The illegal traffic detection mode assumes that only trajectories for legitimate categories are known (and labelled); there are no known trajectories for illegitimate vessels[13].

[13]This setting is often referred to as *learning from positive examples only*

| | Actual | | | |
|---|---|---|---|---|
| **Output** | Long-range transport (1) | Local transport (2) | Fishing (3) | Pirate (4) |
| (1) | 1 | 0 | 0 | 0 |
| (2) | 0 | 0.72 | 0.18 | 0.01 |
| (3) | 0 | 0.20 | 0.77 | 0.03 |
| (4) | 0 | 0.08 | 0.05 | 0.95 |

Table 2: Confusion matrix of the vessel categorizer

The learning phase of the algorithm is similar to the vessel categorizer, only now HMMs are created only for legitimate categories of traffic.

When an unknown vessel trace is to be classified, it is evaluated for closeness against all HMMs. However, because the HMMs now do not cover all categories of traffic, the vessel is classified as "legitimate" only if the closeness of its trajectory to the closest HMM is higher than a defined *closeness threshold*. Otherwise, the vessel is categorized as "illegitimate". By varying the closeness threshold, we can moderate the trade-off between false negatives (an anomalous vessel is classified into one of the legitimate classes) and false positives (a legitimate vessel is classified as anomalous); the trade-off can be quantified using an ROC curve.

We refer to the classifier with the above described structure and operation as the *illegitimate vessel detector*.

## 4.3 Evaluation

The proposed trajectory modeling method was evaluated on vessel traces generated by the data-driven maritime traffic simulation described in Section 2. The simulation was run for 3215 simulation seconds and the trajectories sampled with 10-minute resolution, resulting in 2000 trajectories with 1794 points on average. Four vessel types (as described in detail in Section 3.3) were running in the simulation, producing four categories of traffic. The experimental scenario contained 500 vessels of each class. The execution of the simulation produced traces containing altogether 3 588 909 coordinates which were used for all subsequent analyses.

### 4.3.1 Categorization Classifier

The categorization classifier (4.2.1) was applied on all the data. 50 Gaussian kernels were used in the Level 1 model.

The classification accuracy was evaluated using 10-fold cross-validation. The resulting dependency of the accuracy on the length of test sequences, i.e. the sequences being classified, is given in Figure 6. In order to gain better insight into the operation of the classifier, we have also calculated the confusion matrix (Table 2).

It follows that long-range transport vessels are the most easily identifiable; this is because they have completely different trajectories and visit different locations than other vessels. Local transport vessels are sometimes misclassified with fishing vessels (and vice-versa); this is because both classes of vessels operate close to the coast and their trajectories overlap. For similar reasons but with lower rate, pirate vessels are sometimes misclassified as local transport.

### 4.3.2 Detection Classifier

For the detection classifier (Section 4.2.2), only data corresponding to the three legitimate types of vessels (long-range transport, short-range transport and fishing) were used. Again, 50 Gaussian kernels were used in the Level 1 model.

The accuracy of classification into legitimate vs. illegitimate class was again evaluated using 10-fold cross-validation. As noted above, the trade-off between false negatives and false positives can be regulated by setting the closeness threshold of the classifier. The resulting ROC curve is given in Figure 7.

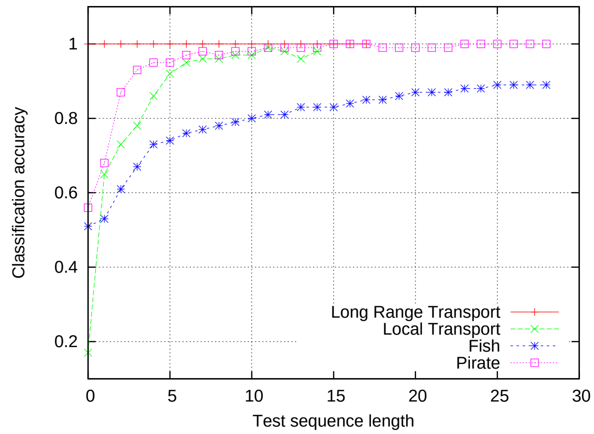


Figure 6: Classification accuracy of the vessel categorizer for different lengths of the classified trace.

## 4.4 Related Work

Perhaps the approach most similar to our work is described in [11] where traffic trajectories are represented as sequences of flow vectors, each vector consisting of four elements representing position and velocity of the object in 2D space. The patterns of trajectories are formed by a neural network. In [12], the authors use the EM algorithm for learning zones. To learn traces, they simply compare a new trajectory with all routes that are already stored in the database using a simple distance measure. The limitation of this method is that they only use spatial information and temporal information is not well represented. In [8], the authors use a system based on a 2-layer hierarchical version of fuzzy K-means clustering. They first cluster similar trajectories into the same cluster according to their spatial information. Each object in a spatial cluster is then clustered according to temporal information. In addition, there are methods for detecting anomalies by direct comparison of behaviors without learning motion patterns [21].

The specific application of motion patterns modeling on vessel traces has been studied in past few years, with existing papers mostly focused on anomaly detection. In [17], the authors use the framework of adaptive kernel estimation and hidden Markov models for the purpose of anomaly detection. In [2] and [16], the authors use a neural network trained on space-discretized AIS data to learn normal traffic and then to detect anomalies as well as to predict future vessel position and velocity.

## 4.5 Summary

The combination of Gaussian mixture models and hidden Markov models enables the proposed method to capture both spatial and temporal aspects of vessel motion patterns. On the test data set obtained from the traffic simulation, the method achieved 95% accuracy when employed on trajectory samples of length 10. The method is fully integrated with

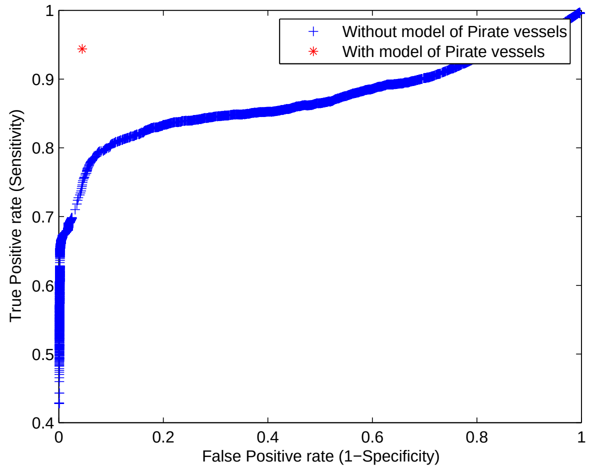


Figure 7: ROC curve for the illegitimate traffic detector for a various closeness threshold. In addition, sensitivity vs. specificity relation for the categorization classifier, calculated from the confusion matrix in Table 2 is also depicted (single point only as there is no variable parameter affecting the relation).

the platform; learning trajectory data are obtained from the platform and the learning results are displayed and can be interactively explored in the Google Earth front-end.

# 5. GAME-THEORETIC RISK-MINIMIZING TRAFFIC ROUTING

Coordination and planning represent the second broad category of security-related maritime transport problems for which the agent-based approach seems particularly fitting. In this section, we present a particular example of planning and show how game-theory-based strategic reasoning can be applied to generate risk-minimizing routes for vessels traversing adversarial areas, such as pirate waters[14].

## 5.1 Transit Game

We formalize the problem of a vessel (further referred to as *Transporter*) traversing an adversarial area (such as the Gulf of Aden). The situation is as follows: a Transporter has to repeatedly traverse a rectangular area from *origin* to *destination*. The area is, however, roamed by an *Attacker* which tries to choose an optimum ambush route starting and ending in its base and attack the Transporter[15].

This problem can be modeled as a zero-sum game between two players, where the strategy set for the Transporter is a set of all paths from the origin to the destination and the strategy set for the Attacker is the set of all possible closed walks from its base. If – following their chosen routes – the Attacker and the Transporter happen to be at the same position, the Attacker wins. If the Transporter avoids encountering the Attacker throughout its whole route, then the Transporter wins.

The solution of the game (for the Transporter) is a probability distribution over possible routes representing the *optimum randomization* of its route selection, i.e., the selection strategy which has the lowest expected risk of pirate attack. This selection strategy corresponds to a mixed Nash equilibrium of the transit game.

A similar problem has been addressed within the framework of ambush games [5, 18]; main difference is that in ambush games the Attacker does not move during the game and its set of strategies is limited to selecting one or multiple ambush locations, making the game significantly easier to solve. We remove this assumption and allow the Attacker to move throughout the adversarial area.

### 5.1.1 *Modeling assumptions*

The area of the game is discretized and represented by a mixed graph with loops (see Figure 8). The graph vertices are assumed to be located on a homogeneous grid covering the adversarial area.

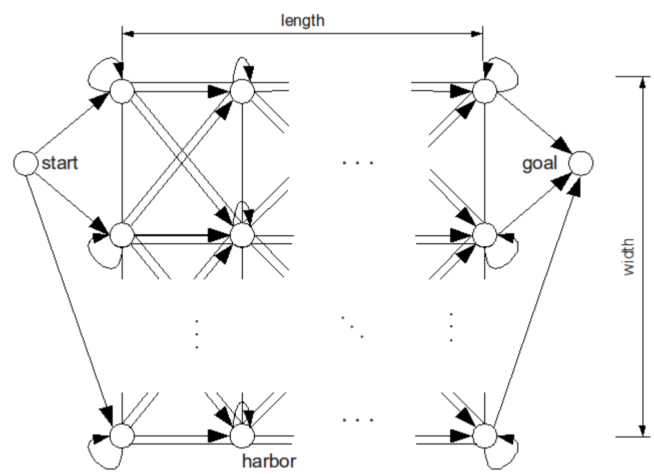


Figure 8: Transit game graph.

Let us now state explicitly the key assumptions made in our model:

- Both players have complete information about the environment but cannot observe the location of the other player.
- The route of both players is planned before the game begins. One player cannot observe the strategy selection of the other. When the game begins, players cannot react to the opponent's move or change their strategies for any other reasons.
- Both players are rational – the Transporter wants to cross the area unscathed; the Attacker wants to successfully attack the Transporter
- If both players meet at one place, the attack is always successful.
- The payoff of the successful attack does not depend on previous strategies and it does not change over time.

Transporter chooses to traverse the graph from start to goal using a path $p$ that is composed from directed edges without loops. Attacker chooses to traverse the graph from

[14]This section serves as an outline of a theoretical problem with non-elementary solution. Detailed analysis of the game as well as its algorithmic solution will be published elsewhere. Some information can already be found in [9].

[15]Several specific strategies for pirate-attacker are described e.g. in [13, 6]

harbor base $v^b$ using a closed walk $cw$ of limited length (this constraint reflects the limited resources available to the Attacker). The walk is composed from the undirected edges or loops. The set of intersecting vertices between $p$ and $cw$ is a set $I_v(p, cw)$ of those vertices where Transporter and Attacker can meet. If this set is not empty, there is a non-zero probability of the Attacker attacking the Transporter.

#### 5.1.2 Payoff

The payoff is defined as to reflect the resource expenditure on the Attacker's side and the chance of a rescue on Transporter's side:

$$u(p, cw) = \sum_{v_{ij} \in I_v(p,cw)} \frac{1}{dist\,(v_{ij}, v^b)} \tag{6}$$

where $dist(v_{ij}, v^b)$ is the distance of vertex $v_{ij}$ from the Attacker's base vertex $v^b$ (for convenience $dist(v^b, v^b) = \epsilon > 0$)[16]. The Attacker wants to maximize the payoff (6) while the Transporter wants to minimize it.

A set of all possible Transporter's strategies is a set of all possible paths from start to destination $S_T$. A set of all possible Attacker's strategies is the set of all plausible closed walks of the limited length from the base vertex $v^b$, $S_A$. It can be seen that both strategy sets grow exponentially with respect to the width and length of the game graph. Determining an optimum randomized route selection strategy for the Transporter (and the Attacker too) requires finding a Nash equilibrium of the above defined game, and is the subject of the following sections.

### 5.2 Complexity Reduction

As noted above, the size of the strategies sets for both the Transporter the Attacker grows exponentially with respect to the size of the graph. Consequently, finding a Nash equilibrium becomes intractable even for small sizes of the graph. It is necessary to reduce the size of the game matrix or express the strategies in equivalent but more compact representations.

We have therefore employed two complexity reductions based on employing alternative strategy set representations for both players; this leads to a significantly lower number of strategy combinations that need to be searched.

#### 5.2.1 Compact form of Transporter's Strategies

The set of all possible Transporter strategies is a set of all possible paths from origin to destination. A mixed strategy for the Transporter thus corresponds to a probability distribution over all possible paths.

We propose to employ an alternative representation of Transporter's mixed strategies. Instead of choosing a distribution over the set of paths, the Transporter chooses a *network flow* over the directed transporter graph, i.e., assignment of real-valued flows to the edges of the game graph respecting the network flow constraints[17]. Figure 9 depicts a particular route selection strategy expressed in flow-based representation.

The two representations are equivalent and each Transporter's strategy can be expressed both in the original path-based and the new flow-based representation. Transformation to the network flow-based strategy space brings significant reduction of the size of the game matrix. Details of the approach can be found in [5].

#### 5.2.2 Approximate Form of Attacker's Strategies

Unfortunately, the network flow-based representation of player's strategies is not applicable to the Attacker's strategy set, because the Attacker tends to use movement patterns that cannot be expressed as network flows (i.e. closed walks with loops). We therefore propose an alternative representation based on *strategy templates*. Each template $\varphi_i$ is a computable function which, given the values of its parameters, produces a closed walk $cw$ as its output. In agreement with the constraints on the game, we require that the length of the walk is limited by a maximum length.

With the introduction of parameterized strategy templates, it is possible to sample from each strategy template a set of strategies as big or small as required, and therefore create game matrix of a desired size for subsequent Nash equilibrium computation. Such a *strategy sampling* provides a trade-off between the computing time and the accuracy of the solution. When using this technique, approximate solutions are produced instead of exact solutions which would require prohibitive amount of computational time.

### 5.3 Game Solution and its Application

The solution of the game is computed by a typical construction (see e.g. [19]) of a linear programming problem. The solution can be directly used in vessel route planning when traversing high-risk areas.

#### 5.3.1 Finding Nash Equilibrium using Linear Programming

A Nash equilibrium of a two-player zero-sum game in normal form can be found by solving a linear programming problem constructed from the game matrix. Due to the linear nature of the network flow constraints, this technique can be extended also to situations where the strategies of *one* of the players are expressed in the flow-based representation.

The values of the primal solution of the linear programming problem are the edge-transition probabilities for the transporter. The network flow constraints ensure that the strategy support forms a set of valid paths from origin to destination.

The values of the dual solution give us the probabilities of the strategies used by the attacker. The expected value of the game is the same for the primal and dual solution due to the minimax theorem.

#### 5.3.2 Route Planning with the Game Solution

The solution of the game for the transporter is a set of valid paths traversing the game graph from the origin to the destination. The graph can be mapped to an adversarial area fulfilling the requirements stated at the beginning of the section. This is the case of the Gulf of Aden which is of a suitable shape.

A game-theoretic Gulf-of-Aden route planner was implemented to navigate vessels through the gulf using this risk-minimizing randomized route selection strategy. The planner selects the route for every transport ship that moves through the high-risk area from the Transporter's optimal mixed strategy distribution calculated as a Nash equilibrium

[16] The simplicity of the definition is achieved by neglecting the direction of the closed walk of the attacker.

[17] Total flow from the origin vertex is equal to 1, total flow to the destination vertex is equal to 1 and the flow to any other vertex is equal to the flow from the vertex.

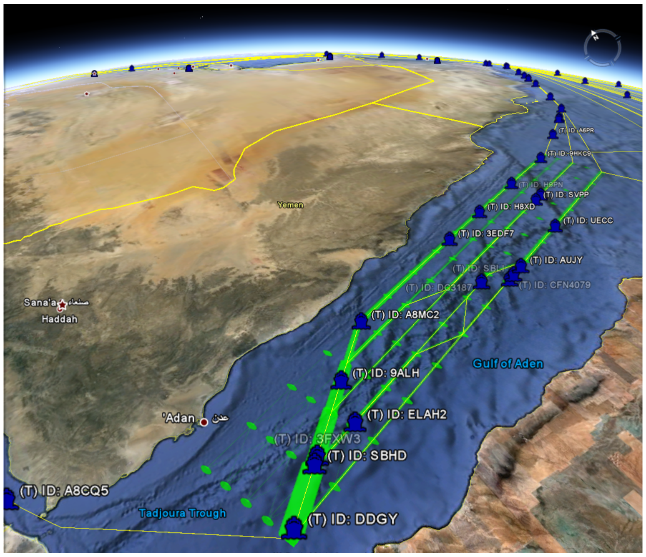


Figure 10: Game graph with computed solution and vessels traversing the graph using computed edge probability distribution. Other vessel types (such as pirates) omitted for clarity.

of the respective transit game. Figure 9 shows one such equilibrium in flow-based representation; Figure 10 than shows the realization of the equilibrium strategy in the testbed.

## 5.4 Evaluation

The proposed method was implemented and integrated with the simulation platform (Section 2). As already outlined, three techniques for determining the route selection strategy for the Transporter were used:

1. Full game with the original representation; we denote this technique as *FULL*.
2. Game utilizing compact edge-based representation for Transporter's strategies; we denote this technique *EB*.
3. Game utilizing compact edge-based representation for Transporter's strategies and template-based representation of Attacker's strategies; we denote this technique *TB*.

Two sets of measurements were conducted to analyze the computational costs of each technique. The complexity of the problem depends on the size of the graph, more specifically on the width of the graph (the width limits the number of all possible walks of the Attacker). The time and space requirements of individual techniques are given in Figure 11; the space requirements are measured as the size of the game matrix which needs to be searched, and thus kept in memory (unless other specialized techniques are employed).

As it can be seen, without the proposed complexity reductions, the transit game is solvable only for very small sizes of the game graph.

## 5.5 Conclusion

The presented method provides a theoretically well-founded route planning method for minimizing the threat to vessels transiting adversarial waters. The application of game theory is novel in this context and the preliminary experimental results show a good promise for future development of the approach. Apart from more in-depth evaluation on a wider range of test scenarios, there are a number of ways in which the approach could be made more realistic. This includes playing the game on arbitrary graphs with edges representing real distances between vertexes as well as incorporating temporal aspects of the game. In addition, the transit game could be modeled with asymmetric payoff which would transform the transit game to a general-sum game. It is also possible to extend the game for N players (it is straightforward to model the game for 1 Transporter and N Attackers, however for M Transporters and N Attackers another strategy representation is needed).

# 6. CONCLUSION

In this paper, we have explored how agent-based techniques can be employed to improve the security of international maritime transport threatened by a steep rise in maritime piracy. Our approach is novel in its integration of agent-based data-driven maritime traffic simulation with advanced analysis and planning methods. Two such methods – one for trajectory-based vessel classification and the other for risk-minimizing transit route planning – have been presented in more detail.

The empirical evaluation of both methods yields promising results, though a great deal of work needs to be done before the methods become applicable in the real world. In the future, we will continue extending the core testbed with better models of maritime activity and improved support for running large-scale experiments. Our primary focus will be on improving and generalizing the learning and planning methods described and on their more extensive evaluation on real-world data.

## Acknowledgements

The work presented is supported by the Office for Naval Research project no. N00014-09-1-0537 and by the Czech Ministry of Education, Youth and Sports under Research Programme no. MSM6840770038: Decision Making and Control for Manufacturing III.

# 7. REFERENCES

[1] L. E. Baum, T. Petrie, G. Soules, and N. Weiss. A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains. *The Annals of Mathematical Statistics*, 41(1):164–171, 1970.

[2] N. Bomberger, B. Rhodes, M. Seibert, and A. Waxman. Associative learning of vessel motion patterns for maritime situation awareness. In *Information Fusion, 2006 9th International Conference on*, pages 1–8, July 2006.

[3] S. Burton, Y. Gauthier, and J. Greiss. Matrics: A maritime traffic simulation. Technical report, Defence R&D Canada, Centre for Operational Research and Analysis, 2007.

[4] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.

[5] A. J. Farmey. Path-planing strategies for ambush avoidance. Master's thesis, Massachusetts Institute of

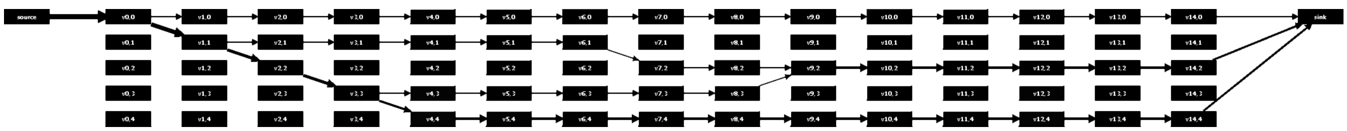

Figure 9: Flow-based representation of an optimum Transporter's transit route selection strategy. The solution has to be vertically flipped (and start and goal vertices removed) to be used for planning in the Gulf of Aden (see Figure 10).

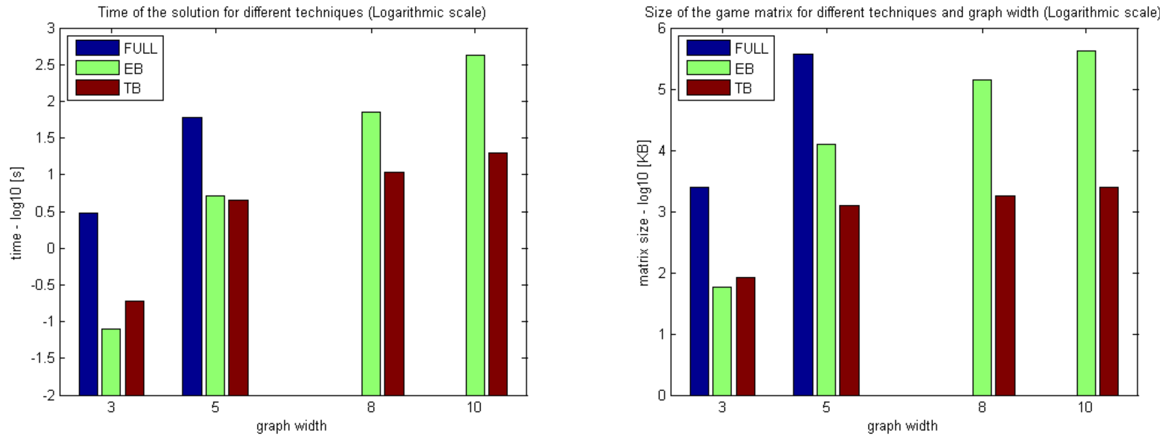


Figure 11: Computational cost of the three variants of the game-theoretic routing algorithm for different width of the adversarial area (game graph): computation time (left); space requirements measured as the game matrix size (right).


Technology, Massachusetts, USA, July 2005.

[6] R. Gilpin. Counting the costs of Somali piracy. Technical report, United States Institute of Peace, 2009.

[7] A. Glaschenko, A. Ivaschenko, G. Rzevski, and P. Skobelev. Multi-agent real time scheduling system for taxi companies. In *Proc. of 8th Int. Conf. on Autonomous Agents and Multi-Agent Systems (AAMAS 2009), Budapest, Hungary*, 2009.

[8] W. Hu, X. Xiao, Z. Fu, D. Xie, T. Tan, and S. Maybank. A system for learning statistical motion patterns. *Pattern Analysis and Machine Intelligence, IEEE transactions on pattern analysis and machine intelligence.*, 28(9):1450–1464, Sept. 2006.

[9] M. Jakob, O. Vaněk, Š. Urban, P. Benda, and M. Pěchouček. Adversarial modeling and reasoning in the maritime domain, Year 1 report. Technical report, Agent Technology Center, Department of Cybernetics, FEE, CTU Prague, 2009.

[10] M. Jakob, O. Vaněk, Š. Urban, P. Benda, and M. Pěchouček. AgentC: Agent-based testbed for adversarial modeling and reasoning in the maritime domain. *Proc. of 9th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2010) - demotrack*, May, 2010.

[11] N. Johnson and D. Hogg. Learning the distribution of object trajectories for event recognition. In *BMVC '95: Proceedings of the 6th British conference on Machine vision (Vol. 2)*, pages 583–592, Surrey, UK, UK, 1995. BMVA Press.

[12] D. Makris and T. Ellis. Learning semantic scene models from observing activity in visual surveillance. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 35(3):397–408, June 2005.

[13] B. Mpondo-Epo. Root causes of piracy in somalia. Kuala Lumpur International Conference, On Piracy And Crimes At Sea, May 2009.

[14] D. Nincic. Maritime piracy in Africa: The humanitarian dimension. *African Security Review*, 18(3):2–16, 2009.

[15] M. Pěchouček and D. Šišlák. Agent-based approach to free-flight planning, control, and simulation. *IEEE Intelligent Systems*, 24(1):14–17, Jan./Feb. 2009.

[16] B. Rhodes, N. Bomberger, and M. Zandipour. Probabilistic associative learning of vessel motion patterns at multiple spatial scales for maritime situation awareness. In *Information Fusion, 2007 10th International Conference on*, pages 1–8, July 2007.

[17] B. Ristic, B. La Scala, M. Morelande, and N. Gordon. Statistical analysis of motion patterns in AIS data: Anomaly detection and motion prediction. In *Information Fusion, 2008 11th International Conference on*, pages 1–7, 30 2008-July 3 2008.

[18] W. Ruckle, R. Fennell, P. T. Holmes, and C. Fennemore. Ambushing random walks I: Finite models. *Operations Research*, 24:314–324, 1976.

[19] Y. Shoham and K. Leyton-Brown. *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations.* Cambridge University Press, 2008.

[20] Š. Urban, M. Jakob, and M. Pěchouček. Probabilistic modeling of mobile agents' trajectories. In *Proceedings of the International Workshop on Agents and Data Mining Interaction (ADMI 2010)*, May 2010.

[21] H. Zhong, J. Shi, and M. Visontai. Detecting unusual activity in video. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'04)*, pages 819–826, 2004.

# Agent Based Intelligent Goods

Åse Jevinger
Blekinge Institute of Technology
PO Box 214
SE-374 24 Karlshamn, Sweden
+46 454 385921
ase.jevinger@bth.se

Paul Davidsson
Blekinge Institute of Technology
PO Box 214
SE-374 24 Karlshamn, Sweden
+46 457 385841
paul.davidsson@bth.se

Jan A. Persson
Blekinge Institute of Technology
PO Box 214
SE-374 24 Karlshamn, Sweden
+46 454 385906
jan.persson@bth.se

## ABSTRACT
The purpose of this paper is to present a framework for intelligent goods and to relate this framework to software agents. It includes a description of how different levels of intelligence connected to the goods can be categorized, as well as of the different possibilities of locating information and processing. Additionally, we present three different types of transport services based on intelligent goods, and show how these can be realized using agents. These agents are finally related to our presented framework.

## Categories and Subject Descriptors
I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence – *Intelligent agents.*

H.4.2 [**Information Systems Applications**]: Types of Systems – *Logistics.*

## General Terms
Management, Design

## Keywords
Intelligent goods, Smart goods, Agents, Decentralized decision-making, RFID

## 1. INTRODUCTION
The increasing demands on transports related to global flows, just-in-time deliveries, intermodality etc., cause more and more complex logistics solutions. In order to cope with this, new instruments are continuously being developed and one of the concepts sometimes mentioned in this context is intelligent goods. The research in the area of intelligent goods is becoming more mature and today there are a number of solutions on the market that are related to intelligent goods [9].

The concept of intelligent goods indicates that some kind of intelligence is put on the local level, i.e. on or close to the goods. The exact meaning of the concept is still somewhat ambiguous though (see next section). In this paper we therefore present a way to categorize the different types and levels of intelligence that are relevant in the context of intelligent goods. Furthermore, we suggest a lowest level of capabilities that the goods should have to be classified as intelligent goods.

Radio Frequency Identification (RFID) is usually assumed to be involved in solutions based on intelligent goods. In the area of RFID, there are some studies on the feasibility of mobile RFID solutions in supply chain management [4]. Furthermore, the idea of combining RFID and agent technology has recently attracted some attention. However, the question of how to use agents for implementing intelligent goods in transports, still needs further investigations. The EU funded project EURIDICE (EURopean Inter-Disciplinary research on Intelligent Cargo for Efficient, safe and environment-friendly logistics) has so far specified a number of general agents that can be used as an instrument for implementing solutions based on intelligent goods [3]. This paper aims at contributing to this area by presenting examples of agent solutions for three specific intelligent goods services. Both single agent and multi-agent solutions corresponding to these services are given and compared. The agent solutions are furthermore categorized according to the intelligent goods framework in order to find the level of intelligence. Additionally, the similarities and differences between the multi-agent solution and the EURIDICE agents are presented. Finally, different locations of the agents are discussed.

The next section outlines the intelligent goods framework. Section 3 presents the three transport services and section 4 describes the agent solutions. Finally, in section 5, some conclusions are drawn.

## 2. INTELLIGENT GOODS
### 2.1 Definition
Within the research in the area of intelligent goods, a number of different denotations for similar concepts are applied, e.g. intelligent cargo [3], smart goods [10], smart freight [8], intelligent goods [5] etc. The meanings of these concepts are usually not identical, and often not precisely defined, but they do strive in the same direction. In our view, the intelligence of the goods is a matter of degree, i.e. goods can be more or less intelligent, from simply knowing its own identity to autonomous decision making. Therefore, we suggest that the level of intelligence should be characterized based on a number of dimensions, corresponding to different capabilities. These dimensions and their different values, ordered according to the degree of intelligence, are shown in the list below. The list has been developed based on the definition of smart freight [8], the requirements from potential intelligent goods services [6] and a

**Cite as**: Agent Based Intelligent Goods, Åse Jevinger, Paul Davidsson and Jan A. Persson, *Proc. of 9th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2010)*, van der Hoek, Kaminka, Lespérance, Luck and Sen (eds.), May, 10–14, 2010, Toronto, Canada, pp. XXX-XXX. Copyright © 2010, International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). All rights reserved.

number of different agent definitions and levels of agent capabilities [1] [11].

**A. Memory storage:**

1. Ability to store ID
2. Ability to store data (other than ID) about the goods
3. Ability to store algorithms/decision rules

**B. Memory allocation:**

1. Static memory
2. Ability to change/add/delete data (other than ID)
3. Ability to change/add/delete algorithms/decision rules

**C. Communication out:**

1. Data (including ID) can be read by another entity
2. Ability to send message

**D. Communication in:**

1. None
2. Data (other than ID) can be written (changed) by another entity
3. Ability to receive message

**E. Processing:**

1. None
2. Ability to execute decision rules (e.g. If –Then statements)
3. Ability to execute algorithms (e.g. planning capability, optimization algorithms)

**F. Autonomy:**

1. None
2. Reactive capability (actions must be triggered by another entity)
3. Proactive capability (no external trigger needed)

**G. Sensor:**

1. None
2. Sensor reading capability (e.g. temperature, position, humidity)

In the list above, we assume that the ID of the goods is static and that the goods always store at least an ID. Furthermore, we assume that the capabilities are cumulative in such a way that a higher capability indicates the ability to also satisfy the lower capability requirements.

Please note that the implementations of the capability dimensions above do not have to be physically located at the goods. For instance, the decision rules may be stored on the goods whereas the processing of the rules may be performed by a nearby processing unit. Furthermore, a second close-by unit might be responsible for the sensor functionality.

The dimensions show that the set of lowest levels of intelligence (i.e. no 1 in all capability dimensions) corresponds to the simple bar codes commonly used in retail today. This level furthermore also includes the simplest form of RFID tags. Based on this, we suggest that the goods should be considered as intelligent if at least one of the capabilities lies above 1, implying an extension of the bar code situation. The concept intelligent goods hereby represents goods with varying intelligence and capabilities but that do fulfill this requirement.

A potential usage of these capability dimensions of intelligent goods, is to map different services, based on intelligent goods, to their implementation requirements. A higher level implies a higher requirement on the implementation. In particular, it is of interest to use these dimensions when multiple services should coexist. For instance, assume that the capabilities of both agents and services are mapped against the capability dimensions. A comparison between these two mappings will reveal what services can be supported by a specific set of agents. Alternatively, the total requirements of the services can be used to specify the capabilities of the agents needed to support this set of services.

A practical example of areas that potentially might benefit from using the intelligent goods framework is the RFID technological development. Depending on application and development, different levels of intelligence is today implemented on the RFID tags. Here, our framework could be used as a way to classify the different RFID tags (e.g. active and passive tags).

## 2.2 Capability dependencies

There are several dependencies between the different capability dimensions. For instance, a capability of only being able to store an ID and not any data (A1) can't be combined with the capability of changing data (B2), since there is no data to change. Naturally, different services put different requirements in terms of capability dimensions. It is thereby relevant to present the dependencies between the different capability dimensions, i.e. how the requirement of a capability propagates to other dimensions. The lowest dimension never propagates to any other dimensions. The table below shows all such propagations.

**Table 1. Propagation of capabilities into other dimensions**

| Capability | Required capabilities |
|---|---|
| Ability to change/add/delete data (B2) | Ability to store data (A2) |
| Ability to change/add/delete algorithms/decision rules (B3) | Ability to store algorithms/ decision rules (A3) |
| Ability to send message (C2) | Ability to store algorithms/decision rules (A3)<br>Ability to execute decision rules (E2) |
| Data can be written by another entity (D2) | Ability to store data (A2)<br>Ability to change/add/delete data (B2) |

| | |
|---|---|
| Ability to receive message (D3) | Ability to store algorithms/ decision rules (A3)<br>Ability to change/add/delete data (B2)<br>Ability to execute decision rules (E2) |
| Ability to execute decision rules (E2) | Ability to store algorithms/ decision rules (A3)<br>Ability to change/add/delete data (B2) |
| Ability to execute algorithms (E3) | Ability to store algorithms/ decision rules (A3)<br>Ability to change/add/delete data (B2) |
| Reactive capability (F2) | Ability to store algorithms/ decision rules (A3)<br>Ability to execute decision rules (E2) |
| Proactive capability (F3) | Ability to store algorithms/ decision rules (A3)<br>Ability to change/add/delete data (B2)<br>Ability to execute algorithms (E3) |
| Sensor reading capability (G2) | Ability to store algorithms/ decision rules (A3)<br>Ability to change/add/delete data (B2)<br>Reactive capability (F2) |

According to table 1, a reactive capability (F2) requires the ability to store algorithms/decision rules (A3) and the ability to execute decision rules (E2). This indicates that the capability of being reactive requires some kind of processing as well. We hereby do not consider for instance bar codes or passive RFID tags as reactive, since these are simply scanned by a reader, without being able to for instance decide for themselves which information to send or whether or not to send any information at all.

All other capabilities not included in table 1 are independent of each other. Based on the capability dimensions, table 1 and furthermore the requirement for being called intelligent goods, it is possible to create a list of all possible combinations of capabilities. Such a list would thereby show all possible levels of intelligence included within the group of intelligent goods.

# 3. SERVICES

The list of services that might benefit from being realized based on intelligent goods is extensive. In this paper we focus on services conducted during loading, transport and unloading. In order to show how agents can be used to realize intelligent goods and what level of intelligence is required for different types of services, we present three concrete and illustrative example services below. The services are based on previous research, presented in [6].

1. **Time of arrival service:** notifies designated receivers about actual arrival times when the goods are arriving at a stop outside the specified delivery time window.
2. **Priority service:** handles the priority requests answering and the calculations of goods priority based on estimated time of arrival (ETA) to the next stop.
3. **Transport conditions service:** continuously records temperature and humidity values, which are provided upon request.

The above services might be realized using intelligent goods (e.g. instead of centralized configuration) and the intelligent goods might in turn be realized using agents. Next we will show what requirements these services put on such agents. These requirements can then be expressed in terms of the presented capability dimensions of intelligent goods.

Please note that there might be alternatives to using agents for the realization of a service. For instance, the former company Bioett [7] used a biosensor to track the accumulated temperature, which could be read by a handheld scanner. A mapping of this solution against the intelligent goods framework, shows that it falls within the category of intelligent goods (since it at the very least stores data) and that it requires less intelligence than the services presented by us.

# 4. AGENTS

This section presents what kind of information and agents that are needed to fulfill the requirements of the services. The agents presented in the second subsection have been created under the assumption that we only have one single agent for each service, i.e. one agent is responsible for all functionalities related to a service. However, in real life, this may not be the case at all. Different parts of the functionality may be placed on different units and furthermore on different communication levels (e.g. ERP, vehicle/terminal or goods level). In such situations each agent is responsible for its own specific functionality, which may represent a subset of the agent functionality described in section 4.2.

The agents presented in subsection number three shows one way of dividing the service functionality between different agents. In these solutions functionality that can be reused by more than one service, have been extracted and placed in separate agents. The structure of the result naturally depends on the set of services considered. An investigation of a large number of potential intelligent goods services should thereby result in a number of general agents that form a basic structure for service implementation. This type of investigation represents a candidate for future research tasks.

In the descriptions of the agents, fundamental functionalities such as activation, deactivation etc. has been left out. Furthermore, some form of security will most probably be needed in order to restrict who is allowed to access the information entities and make use of the service functionalities. This has also been left out.

## 4.1 Information entities

In order to realize the services presented above, a number of information entities related to the goods are needed. For instance, the Time of arrival service needs information about the prespecified delivery time window, which is specific for each transport item. Moreover, a few information entities related to the container (e.g. transport container or vehicle) of the goods are also needed.

The table below lists the information entities required to fulfill the three services in question. The goods information entities are assumed to be stored in the databases shown in the agent figures presented in the following sections, whereas the container information entities are considered as a part of the input data messages.

**Table 2. Information entities needed for by the services**

| No | Goods information Entities (GE) | |
|---|---|---|
| **1** | GE-ID | <ID>, where ID is the unique goods identity, e.g. SGTIN, SSCC, GRAI [2] |
| **2** | GE-Next Destination | <position>, where position is an address, SGLN [2] or a set of coordinates |
| **3** | GE-Itinerary | Sequence of <position, accountable ID, time 1, time 2>, where position is an address, SGLN or a set of coordinates, accountable ID is the organization currently accountable for the goods (usually a transport company) and time 1 and time 2 represent the delivery time window |
| **4** | GE-Priority | <priority>, where priority is the delivery priority of the goods |
| **5** | GE-Customer Class | <class>, where class indicates the priority ranking of a transport customer (for instance based on the amount paid for a reliable delivery time of the goods) |
| **6** | GE-Recorded Conditions | Set of <condition type, a sequence of <time stamp, value>>, where condition type is the stored condition (e.g. temperature), and time stamp is the time of measurement and value is the condition value (e.g. temperature value) |
| **7** | GE-Set of Information Receivers | Set of <service no, receiver contact>, where the service number is a predefined number identifying the service (e.g. 1, 2 or 3 in our list) and receiver contact is the contacting details of the receiver of the information (e.g. telephone number) |
| **No** | **Container information Entities (CE)** | |
| **1** | CE-Itinerary | Sequence of <position>, where position is an address, SGLN or a set of coordinates |
| **2** | CE-Accountable | <accountable ID>, where accountable ID is the organization (currently) accountable for the container |
| **3** | CE-ETA | <ETA>, where ETA is the estimated time of arrival to the next stop of the container |

## 4.2 Single agents

### *4.2.1 Time of arrival service*

The Time of arrival service, presented in section 3, can be realized using the agent, illustrated in figure 1. This agent (denoted O1) is not triggered by an external entity. Instead it uses position data to discover that the goods have arrived to one of the stops. In order to find the required delivery time window to the next stop, the information entity GE-Next Destination is used to search through GE-Itinerary. Furthermore, GE-Set of Information Receivers is used to find the recipients of the notifications.

Based on Figure 1, it is now possible to relate the capabilities of the agent to the capability dimensions connected to intelligent goods, presented in section 2. We hereby are able to conclude that in order to implement the Time of arrival service using one single agent, the following capabilities are needed (at least) in that agent:

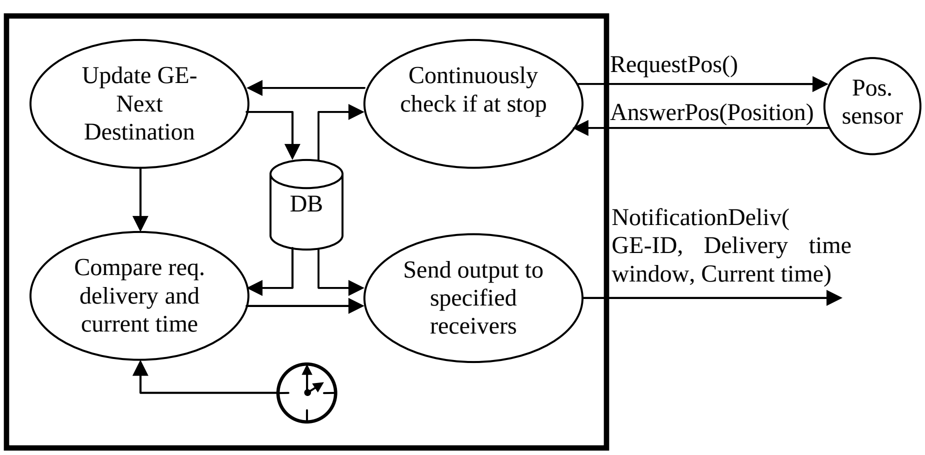


**Figure 1 Illustration of a single agent (O1) that implements the Time of arrival service**

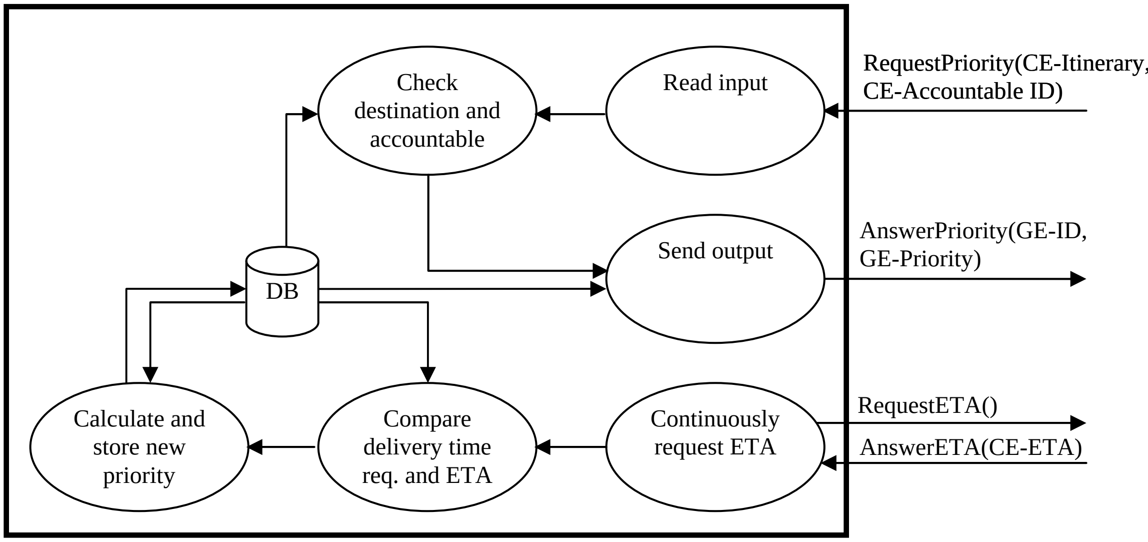


**Figure 2 Illustration of a single agent (O2) that implements the Priority service**

- Ability to store algorithms/decision rules (A3)
- Ability to change/add/delete data (B2)
- Ability to send message (C2)
- Ability to receive message (D3)
- Ability to execute decision rules (E2)
- Proactive capability (F3)
- Sensor reading capability (G2)

### 4.2.2 *Priority service*

Based on the same discussion as in section 4.2.1, we present an illustration in figure 2 of an agent (denoted O2) that can be used for the realization of the Priority service.

The agent calculates the new priority of the goods based on the estimated time of arrival (ETA) to the next stop of the container and based on the class of priority ranking (i.e. GE-Customer Class). For instance, if the goods are delayed and the class of priority ranking is high, the priority of the goods is increased. The use of ETA means that if the goods are located inside a terminal/warehouse, the terminal/warehouse needs to be able to provide an estimation of the time of arrival to the next stop. Presumably, an additional agent located in the terminal/warehouse is responsible for this estimation.

The information provision about the goods priority is based on the itinerary and who is accountable for the transport. This means that an answer to a priority question is only given if the accountable ID matches the accountable ID of the question and if the next stop of the goods can be found within the itinerary, also included in the question. The update of GE-Next Destination has not been included in figure 2 since it is identical with the one described in figure 1. This update functionality causes the sensor capability to go from G1 to G2 though (see list below).

When relating the capabilities of the agent in figure 2, to the capability dimensions connected to intelligent goods, we are able to conclude that in order to implement the Priority service using one single agent, the following capabilities are needed (at least) in that agent:

- Ability to store algorithms/decision rules (A3)
- Ability to change/add/delete data (B2)
- Ability to send message (C2)
- Ability to receive message (D3)
- Ability to execute algorithms (E3)
- Proactive capability (F3)
- Sensor reading capability (G2)

### 4.2.3 *Transport conditions service*

The transport conditions service, presented in section 3, can be realized using the agent illustrated in figure 3. The agent (denoted O3) continuously retrieves sensor data from the condition sensors and uses the information entity GE-Recorded Conditions to store the data.

When relating the capabilities of the agent in figure 3, to the capability dimensions connected to intelligent goods, we are able to conclude that in order to implement the Transport conditions service using one single agent, the following capabilities are needed (at least) in that agent:

- Ability to store algorithms/decision rules (A3)
- Ability to change/add/delete data (B2)
- Ability to send message (C2)
- Ability to receive message (D3)
- Ability to execute decision rules (E2)

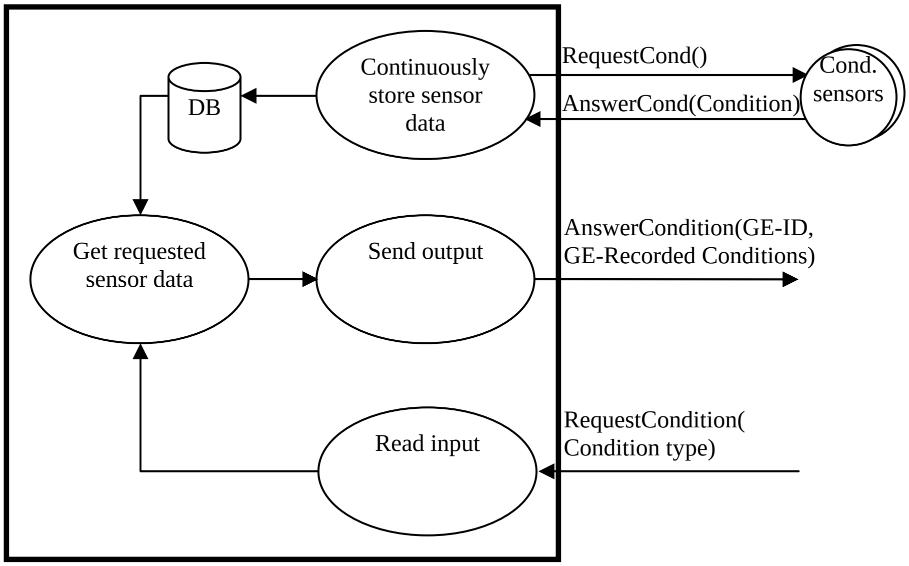


**Figure 3 Illustration of a single agent (O3) that implements the Transport conditions service**

- Proactive capability (F3)
- Sensor reading capability (G2)

### *4.2.4 Discussion*

Since the three services presented in this paper differ in functionality, the corresponding agents have different capability levels. They do have a few things in common though; they all need the ability to store and change algorithms/decision rules/data (other than ID) about the goods and they all need to be able to send and receive messages. Furthermore, they also need a high level of autonomy and sensor readability. These seem to be fundamental capabilities, at least for these three services.

The above agents all include one internal database each, which is used for storing the service specific goods information entities. Furthermore, all the agents need the ability to continuously read some kind of sensor data (either explicitly or implicitly) and two of the agents are dependent on a correct update of GE-Next Destination. These three multiplied functionalities imply that the agents might benefit from extracting these functionalities into separate agents. However, as mention before, the value of this naturally depends on the set of services to be implemented. In particular, storing two copies of the same goods information entity in two different agents might involve some risks, especially if it is a dynamic information entity that needs to be updated from time to time. In some cases though, redundant information entities might give the agent control over the information. For instance, the Time of arrival service uses the information entity GE-Next Destination to discover when the goods have reached a new stop. This implementation of the service assumes that GE-Next

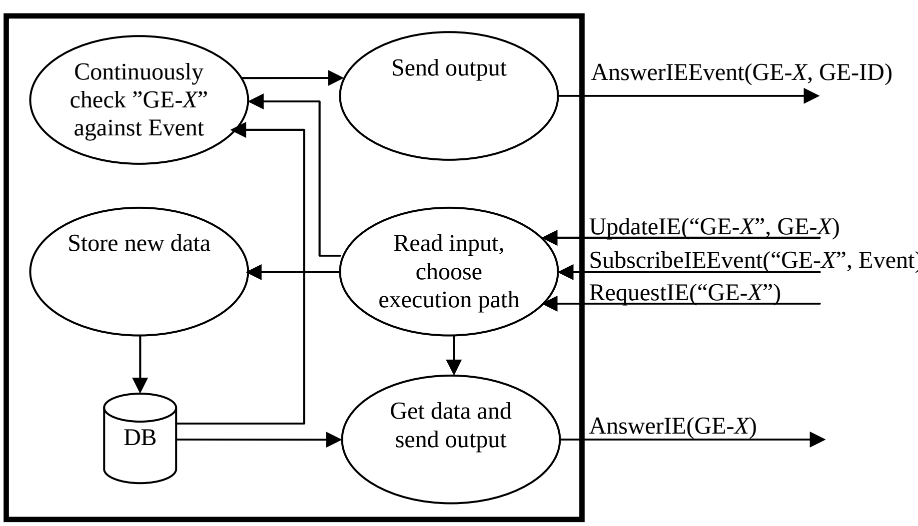


**Figure 4 Agent responsible for extracted database functionality (E1)**

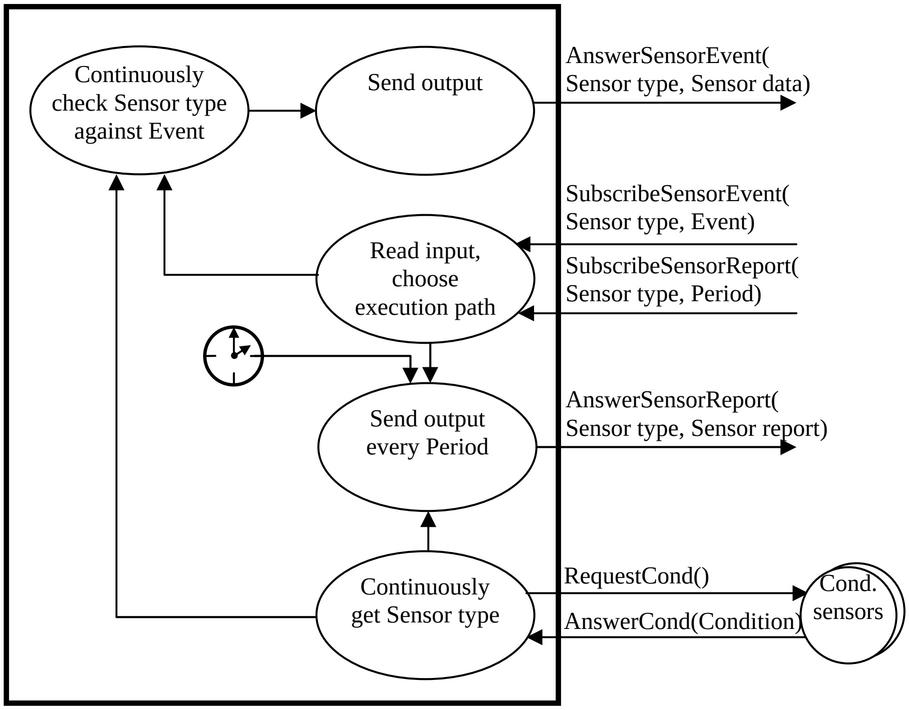


**Figure 5 Agent responsible for extracted sensor reader functionality (E2)**

Destination is updated after the check for a new stop. In this case, storing and updating the information entity inside the agent is beneficial since it gives the agent the control of deciding when to update. However, this problem can be solved in a distributed solution as well, which will be shown in the next section.

## 4.3 Multiple agents

### *4.3.1 Extracted agents*

Figure 4 shows the implementation of an agent responsible for the database that holds the relevant goods information entities. GE-*X* in the figure represents the goods information entity in question, i.e. the information entity to be updated, requested or subscribed for (triggered by an event). Figure 5 shows an agent responsible for the sensor reading. Both agents represent an extraction of the previously presented agents in figures 1-3. Finally, figure 6 shows the interaction between the service specific agents (denoted S1-S3) and the extracted agents (denoted E1-E2). The service specific agents are assumed to hold the service specific functionalities that are not supported by the extracted agents. Figure 6 also includes an additional agent, called S4. This agent is responsible for updating the goods information entity GE-Next Destination. The reason for creating this agent is that its functionality is needed by two of the services (as discussed above). This agent represents an extraction of the previous agents, but it can at the same time be seen as a subservice that assists the three service specific agents. It thereby both use the other extracted agents and is used by the service specific agents.

Figure 6 only presents the initiated communication calls. The answers are thereby not included, but only indicated by communication arrows.

### *4.3.2 Discussion*

When dividing the functionality of a service between different agents, two measurements of the level of intelligence arise: the level of intelligence at individual agents and the total level of intelligence of the set of agents responsible for the service. In our case, the level of intelligence at the individual agents varies whereas the total level corresponds to the single agent solutions. Generally, all agents involved in the realization of a service based on intelligent goods, form the capability basis of that service. The level of the total intelligence can be found using the capability dimensions.

Using more than one agent to implement an intelligent goods service usually provides a higher level of flexibility in the sense that the agents can be spread out in an optimized way. For instance, the physical location (goods level, vehicle level etc.) of an agent might influence the decision of what functionality to include in that agent. Furthermore, our three services show that instead of letting many agents include high capability levels, individual agents can be specialized on the different tasks needed. For instance, the sensor reading capability can be dedicated to our agent E2 only.

Table 3 shows the mappings of the agents representing the divided functionality (S1-S4 and E1-E2) against the capability dimensions presented in section 2. The table also includes the corresponding mappings of the original service agents (O1-3), presented in section 4.2, as reference material. As expected, these mappings show that when extracting some of the common functionality into separate agent and thereby dividing the total service functionality, the resulting agents need the same or less capability than in the single agent solutions. The fundamental capability dimensions, such as memory storage, communication

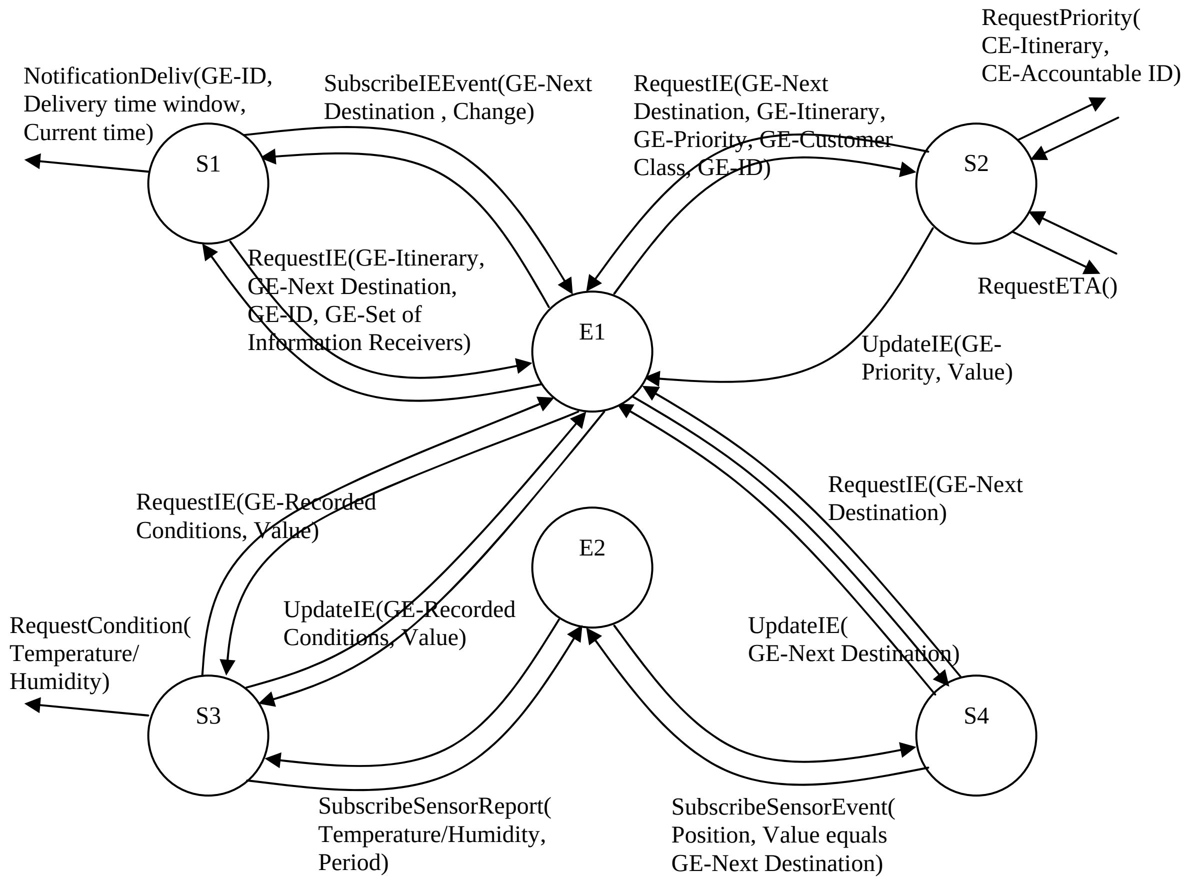


**Figure 6 Multiagent system implementing all three services**

etc. remain the same for all agents, but the rest of the dimension levels differ.

**Table 3 Mappings of agents against capability dimensions**

| | S1 | S2 | S3 | S4 | E1 | E2 | O1 | O2 | O3 |
|---|---|---|---|---|---|---|---|---|---|
| **Memory storage** | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| **Memory allocation** | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| **Com. out** | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| **Com. in** | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| **Processing** | 2 | 3 | 2 | 2 | 2 | 2 | 2 | 3 | 2 |
| **Autonomy** | 3 | 3 | 3 | 3 | 2 | 3 | 3 | 3 | 3 |
| **Sensor** | 1 | 1 | 1 | 1 | 1 | 2 | 2 | 2 | 2 |

### *4.3.3 Relations to EURIDICE*

The EURIDICE project aims at development and diffusion of the intelligent cargo, intended as a paradigmatic change in the field of information and communication technology applications for transport logistics [3]. As a part of this work, EURIDICE has defined a number of general agents that might be used in the realization of a service. Using the EURIDICE agents for our three services means dividing the functionality into several agents. A comparison between the EURIDICE agents and our multi-agent solution shows a number of similarities. Our sensor agent (E2) includes functionality that can be found in the two EURIDICE agents AgentSensorReporting and AgentSensorAccess. Furthermore, EURIDICE mention agents that are capable of long range transmitting and buffering, respectively. These two abilities are needed in our solution as well, for instance when sending notifications (agent S1) and when buffering goods information entities and sensor reports (agent E1).

An interface towards the goods information entities is created through our agent E1. Interestingly, a corresponding or similar agent can not be found in the EURIDICE specification [3], and this hence points on a potential extension of the EURIDICE specification

## 4.4 Location of agents

There are several ways of dividing the functionality of the services between different parts of a system. The information entities related to the goods and the algorithms/decision rules may be stored on the goods, on the transport container/vehicle/terminal or centrally. Moreover, this data may also be spread out between different parts of the system. Different parts of the processing of a service may similarly be conducted on various units. Furthermore,

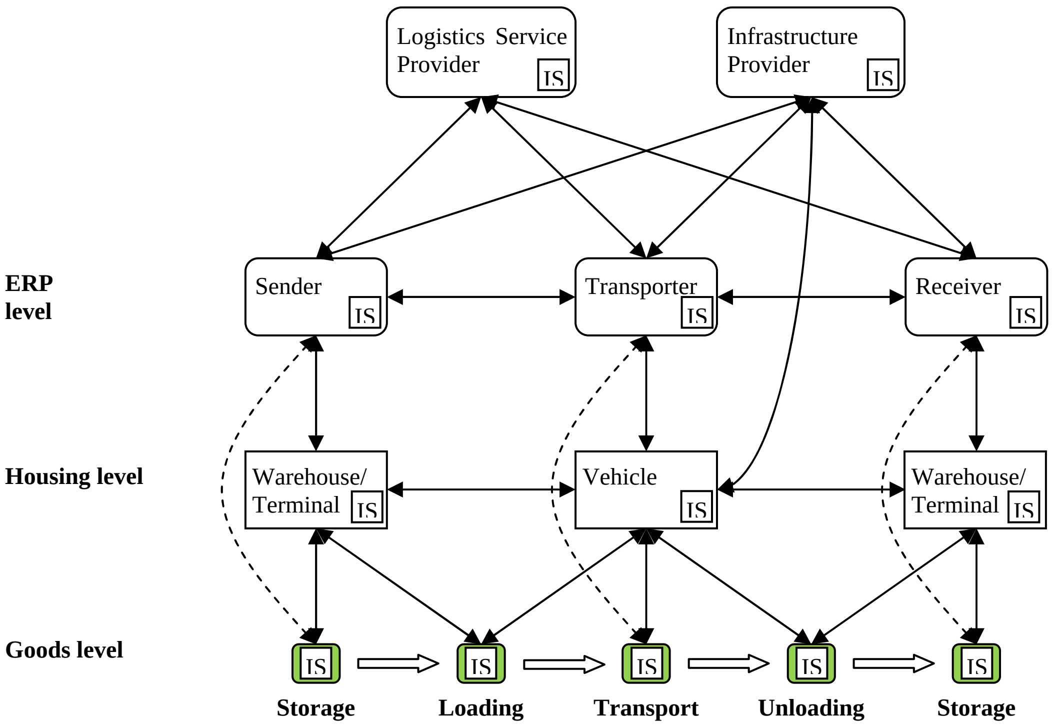


**Figure 7 Context around the goods**

the data storage might very well be separated from the processing. For instance, the data needed by an intelligent goods service might be located on the goods, whereas the corresponding processing might be conducted on the vehicle or central level.

Figure 7 shows the context around the goods. In particular, the figure shows the main alternatives for placing data, rules and processing connected to a service. There are three different information processing levels; the ERP, the Housing and the Goods level. The word *housing* is here used to denote the building (e.g. warehouse and terminal) or vehicle (e.g. truck, ship, train and airplane) in which the goods might be located. The figure furthermore shows the stages that the goods go through during transport, and the different communication paths that might exist between the information processing levels. IS denotes the information system, which is responsible for any communication and processing that might exist on a level. An RFID tag might for instance represent the IS on goods level.

The optimal solution to the problem of where to place data, rules and processing is dependent on things like the service in question, communication link availability, costs etc. Different solutions suit different situations. Generally, from a strict functional perspective, if the complete service functionality can be conducted locally (i.e. on goods, transport container and/or vehicle/terminal level), the service becomes independent of higher levels. The service may thereby be performed in situations where the communications towards other units are cut off. A higher level of autonomy is thereby obtained. For instance, if the goods get lost or stolen, they might be able to notify the owner of the incident and furthermore provide him with the current position data.

Another advantage with local services is that the closer to the goods the sensors are placed, the more precise sensor data can be obtained. All these advantages must however, as mentioned above, be valued against all other factors specific for each situation and service.

## 5. CONCLUSIONS

We have presented a framework for intelligent goods, which characterizes the level of intelligence by a number of capability dimensions. In relation to this, we have suggested a lowest requirement for when the goods should be considered as intelligent. We have furthermore shown how agents can be used to realize services based on intelligent goods. The capabilities of the agents have moreover been related to the intelligent goods framework. Finally, we have discussed some considerations and effects of different placements of the data, decision rules, algorithms and processing corresponding to a service.

## 6. ACKNOWLEDGMENTS

This research has been carried out within a project called “Intelligent industrial goods and ERP systems” [5], which is funded by the Swedish Road Administration. The project is executed by Blekinge Institute of Technology and Chalmers University of Technology.

## 7. REFERENCES

[1] Davidsson, P. and Johansson, S. J. 2005. On the Metaphysics of Agents (extended version), BTH research report 2005:4, ISSN: 1103-1581

[2] EPCglobal: http://www.epcglobalinc.org

[3] EURIDICE WP 14 D14.1, 2008: Preliminary User, System and Communication Services Definition, http://www.euridice-project.eu

[4] Holmqvist, M. and Stefansson, G. 2006. Mobile RFID A case from Volvo on innovation in SCM. Proceedings from the 39th Hawaii International Conference on System Sciences 2006

[5] Intelligent industrial goods and ERP systems: http://www.bth.se/tek/intelligent_gods

[6] Jevinger, Å., Persson, J.A. and Davidsson, P. 2009. Analysis of intelligent goods and local decision making. Proceedings from ITS World Congress 2009, Stockholm.

[7] Kerry, J., Butler, P. 2008. Smart packaging technologies for fast moving consumer goods. John Wiley & Sons Ltd. ISBN 9780470753682.

[8] Lumsden, K. and Stefansson, G. 2007. Smart freight to enhance control of supply chains. International Journal of Logistics Systems Management, v.3, n.3, 315-29

[9] Savi: http://www.savi.com

[10] Stefansson, G. and Lumsden, K. 2008. The essentials of smart transportation management. Beyond Business Logistics, Proceedings from NOFOMA 2008, 567-582

[11] Wooldridge, M. 2009. An introduction to multiagent systems. Second edition. John Wiley & Sons Ltd. ISBN 9780470519462.

# Building Agent-Based Models of Seaport Container Terminals

José M. Vidal
Computer Science and Engineering Dept.
University of South Carolina
Columbia, SC, 29208
vidal@sc.edu

Nathan Huynh
Civil and Environmental Engineering Dept.
University of South Carolina
Columbia, SC, 29208
huynh@cec.sc.edu

## ABSTRACT

Agent-based models are increasingly being used to simulate and analyze various transportation problems, from traffic flow [15] to air traffic control [1]. One transportation industry that has not received as much attention from the multi-agent systems community is seaport container terminals. It can be argued that the operations that take place at a container terminal are as complex as that of an airport. A seaport container terminal faces a myriad of operational challenges such as optimizing berth space, minimizing ship turnaround time, maximizing use of resources, and reducing wait time of drayage trucks. Due to environmental concerns, terminal operators and port planners are focusing on the problem of reducing the in-terminal wait time of drayage trucks. In this paper, we present our multiagent model of a container yard operation, its implementation using NetLogo, and some initial test results. We model yard cranes as opportunistic utility-maximizing agents using several different utility functions for comparison purposes. By using a representative layout of a terminal our simulation model allows us to analyze the behavior of the cranes and evaluate the collective performance of the system. We demonstrate that it is possible to build a realistic and useful model of yard crane operation. Our test results show that utility functions that give higher precedence to nearby trucks lead to much better results than those that favor serving trucks on a mostly first-come first-serve order.

## Categories and Subject Descriptors

I.2.11 [**Distributed Artificial Intelligence**]: Multiagent systems

## General Terms

Experimentation

## Keywords

Simulation techniques, tools and environments, emergent behavior, yard cranes, seaport container terminals

## 1. INTRODUCTION

Agent-based models have been often used to analyze automobile and air traffic. In this paper we continue that trend, but focus on an entirely topic: supply chain and logistics. In particular, we build an agent-based model of a seaport container terminal. These terminals facilitate the movement of containers between the sea and land. Once import containers are discharged from a vessel, they are stored in a container yard (see Figure 1). Rubber-tired gantry cranes (see Figure 2) or commonly known as "yard cranes" then move these containers onto the trucks. The problem faced by the crane operators is deciding which truck to service first since many arrive at the same time. The process whereby a truck goes to a seaport terminal to pick up or deliver a container with both the trip origin and destination in the same urban area is known as *drayage*.

Drayage activities play an important role in supply chain and logistics. From seaport terminals, drayage drivers and trucks transport import containers to first receivers where consolidation, stripping, transfers, and intermodal activities are undertaken. They also deliver containers to final receivers directly or via key rail intermodal terminals across the nation. This process is reversed for export containers. Drayage operations are now widely recognized as a critical emissions, congestion, and capacity issue for major container ports and rail intermodal terminals. Public agencies are rapidly developing policies and programs to reduce related emissions.[1] Concurrently, drayage firms and terminal operators are working to improve drayage operations that are highly inefficient at the present. Despite the relatively short distance of the truck movement compared to the rail or barge haul, drayage accounts for a large percentage (between 25% and 40 %) of origin to destination expenses [9]. In turn, high drayage costs seriously affect the profitability of an intermodal service.

The seaport container terminals have long been identified as bottlenecks and sources of delay for port drayage. The time drayage trucks spent in the queue at the entry gate, container yard, and exit gate are often exceedingly long during peak times at busy terminals. Drayage trucks are diesel-fueled, heavy-duty trucks that transport containers, bulk, and break-bulk goods to and from ports and intermodal rail yards to other locations[2]. Truck idling in the queues is a contributing source of emissions and noise at terminals. High truck turn time is the result of demand exceeding supply. Truck turn time refers to the time it takes a drayage truck to complete a transaction such as picking up an import container or dropping off an export container. It is a measure of a terminal's efficiency in receiving and delivering containers. For terminals that stack their containers,

[1]For example see `www.epa.gov/cleandiesel`.
[2]See the California Environmental Protection Agency `www.arb.ca.gov/msprog/onroad/porttruck/drayage truckfactsheet.pdf`

demand is mainly the number of drayage trucks coming to the terminal to pick up or drop off containers. Supply is the number of yard cranes available to serve these drayage trucks. Supply is typically low on high volume vessel days because the majority of the yard cranes are assigned to work the vessel. In such a scenario, drayage drivers must wait for a longer period of time before a yard crane is available to perform the load or unload move. This waiting process can take a considerable amount of time.

The solution of adding more yard cranes to reduce truck turn time may seem obvious for terminals that stack their containers. However, the high initial investment, plus maintenance and operating costs of these cranes often prohibit terminals from freely buying more. Also, once a drayage truck arrives at its destination in the yard, its turn time is not only dependent on the number of cranes available, but also the service strategy in which the cranes follow. To date, no study has adequately examined the effect of crane service strategy on truck turn time. The challenging issues inherent in this problem, coupled with the limitation of existing research, motivate this study. In addition, this study addresses the practical challenges of increasing supply chain efficiency while reducing the carbon footprint. Specifically, this study investigates how to deploy yard cranes in an effective manner to reduce drayage trucks in-terminal wait time. Reducing the drayage trucks in-terminal dwell time is equivalent to reducing local and regional particulate matter (PM 2.5), nitrogen oxides (NOx), and greenhouse gas (GHG) emissions. PM 2.5 emissions from diesel engines are recognized by the Environmental Protection Agency (EPA) as a serious health issue.

The following describes the study's innovative, decentralized approach to modeling yard cranes by using agent-based modeling and utility maximization to investigate the effectiveness of different crane service strategies. While agent-based models have been widely used in many different discipline, they are relatively unexplored in the area of drayage and port operations.

## 2. LITERATURE REVIEW

Much of the research directly related to yard cranes' work schedule has been carried out using mathematical programming techniques (e.g integer programs or mixed integer programs). As such, these studies seek to optimize the work flow of cranes for a given set of jobs with different ready times in the yard. The "jobs" considered vary from study to study, and they could be either drayage trucks, or other yard handling equipment such as prime movers and internal transfer vehicles. Given that the scheduling problem is NP-complete, many studies proposed algorithms or heuristics in order to solve the real-world large-scale problem in a reasonable amount of time, including dynamic programming-based heuristic [10], branch and bound algorithm [11], Lagrangean relaxation [19], and simulated annealing [7]. In the study by Kim et al. [6], a simulation study was performed to compare the performances of several heuristic rules:

First-come-first-serve: trucks are served in the order of their arrival time at the yard. Uni-directional travel: a yard crane travels in one direction and serves trucks until there are no more trucks remaining in the direction of the travel. After serving all the trucks in the direction of travel, the yard crane starts to travel in the opposite direction. Nearest truck first: a yard crane serves the truck that is located nearest to it. Shortest processing time: a yard crane serves the truck with the shortest transfer time, which is the sum of the travel time and the time for transferring the corresponding container to and from the truck. The transfer time includes the time for re-handling containers on top of the target container in the case of a delivery operation.


**Figure 1: Illustration of bay, row, and tier in a yard block.**

This study differs from the aforementioned mathematical programming related work in several ways. First, it takes a decentralized view instead of a centralized one. That is, the resulting cranes work flow is not governed by one optimal schedule. Rather the work flow stems from the individual decisions made by the crane operators. Second, it does not make any assumption regarding the ready times of the jobs. In this study, the number of drayage trucks that arrive to the yard is assumed to be Poisson distributed. Lastly, this study relies on agent-based simulation instead of a mathematical program. The agent-based feature also differentiates this study from the work of Kim et al. [6]. Moreover, each agent (i.e. crane operator) makes his decision based on a utility and not a prescribed heuristic rule.

Within the agent-based modeling community there have been some attempts at building simulations of container ports, but these address different parts of the problem. For example, in [2] the authors try to find optimal solutions for the placement of containers in the yard while assuming that the cranes use a fixed policy. In contrast, our research focuses on finding the optimal strategies for the cranes to use to minimize the trucks' wait time given random truck arrivals. Some preliminary work on simulating the ships and their allocation is presented in [12], and similarly in [14]. Most recently, the work of Henesey *et al* investigates the movement of containers from the ship into the yard [5], and looks at various policies for the sequencing of ships, berth allocation, and the use of simple stacking rules [4]. Their SimPort implementation demonstrates that a sophisticated agent-based simulation of a container port can be used to make prescriptive recommendations on how to manage the system. Our research differs from these attempts in that we do not try to model the full system, from ships to trucks, nor do we try to make recommendations on how the complete system would work best. Instead, we focus on one small part of the problem—the yard cranes' service strategies—which we believe can be improved. We believe it is unlikely that a seaport will completely change their workflow because of the results of a simulation. On the other hand, small incremental changes that have been shown to have immediate positive effects on the bottom line are likely to be adopted. Our research aims to work within the real-world constraints of a working seaport, improving its overall efficiency via continuous small improvements.

There has also been some research done in building agent-based models of traffic system. For example, some have built simulations of automated intersections [1], or of air traffic control systems [15, 16], or studied other agent-based methods for solving the urban traffic problem [3, 17]. Our simulations thus take us one step further towards being able to build complete models of transportation systems, including vehicular traffic, freight, and intermodal facilities.

**Figure 2: Yard crane working the stacks.**

# 3. PROBLEM DESCRIPTION

A typical drayage move involves either a delivery of an export container to the seaport terminal or pickup of an import container. A drayage driver arriving to pick up a loaded import container may encounter one of three basic systems.

1. At wheeled terminals the driver will simply locate and retrieve the container on its chassis in the parking area.
2. At stacked terminals, the driver will usually first retrieve a chassis and then position the chassis in the container storage stacks to receive the container from a lift machine (typically yard crane).
3. At some stacked and straddle carrier terminals, the drayage driver will retrieve a chassis and then proceed to a designated transfer zone. A lift machine then brings the container to the waiting driver.

At stacked terminals, the containers are stacked on top of one another in separate yard blocks. Each yard block has about 80 20-foot bays, each bay has 6 rows, and each row has 4 tiers (Figure 1). A yard block is used for storing import containers, export containers, or both. Import containers are typically stored in the available blocks designated for imports and where it is most convenient for the stevedores to facilitate the vessel operations. As import containers are discharged from a vessel, they are stacked in the allocated space without any segregation. Export containers, on the other hand, are methodically segregated by 1) vessel, 2) port of discharge, 3) size, and 4) weight. This is done so that when export containers are transferred from the yard to the vessel, no re-handling (i.e. reshuffling of containers to retrieve the desired one) is required. Note that both the import and export processes are done in a manner to minimize the turn-around time of vessels.

Most U.S. seaport terminals use rubber-tired gantry (RTG) cranes, often referred to as yard cranes, to load and unload containers in the yard blocks. Figure 2 shows a cross section view of a yard block, and it illustrates how a yard crane is positioned in a block. On any given day, the yard cranes are assigned to either support the vessel operation or support the road operation. Vessel operation has higher priority, so the number of yard cranes available to support road operation is the total number of yard cranes available minus the number of yard cranes assigned to vessel operation. Road operation refers to the landside process where drayage trucks come to drop off export containers and/or pick up import containers. Vessel operation refers to the waterside process where import containers are transferred from a vessel to the yard and export containers are moved from the yard to the vessel.

A typical import process involves a drayage driver moving a loaded container from the seaport terminal to the consignee location and then returning an empty to the terminal. The process of taking a loaded container out of the terminal begins with the shipping line in charge of the container requesting drayage service. The manifest is transferred to the drayage company and at the same time to the terminal. The drayage company then creates a pickup order and subsequently dispatches the driver. In order to take a loaded container out of the terminal the driver first arrives at the terminal gate. At this stage, the driver must scan or show his driver's license and then provide the container number to the gate clerk. He must also specify whether he needs to pick up a chassis. If there are no issues with his transaction, the driver receives a pick-up ticket and is cleared to enter the terminal. If the driver does not need a chassis, he then proceeds to the pre-designated pick up area and waits to be serviced by a yard crane (Figure 3).

Depending on the availability of yard cranes and their service strategies, this wait can be a source of extensive delay. Once the yard crane arrives at the bay where the truck has been waiting, the crane operator must locate the requested container and must often re-handle other containers on top before reaching the target container. After the container is loaded onto his truck, the driver must verify that it is the correct container and undamaged. He then must lock the chassis and proceed to the radiation inspection station. After the radiation inspection by Customs and Border Protection (CBP), the driver scans or shows the pick-up ticket and waits for the clerk to perform the damage inspection of the container and issues an Equipment Interchange Report (EIR), ending the out procedure and allowing the truck to exit the terminal.

The yard cranes are operated by operators who are given the freedom to make judgment calls on how to go about the yard to serve drayage trucks. At the Port of Charleston, the operators generally aim to minimize the trucks' wait time. However, they are given the flexibility to pick the

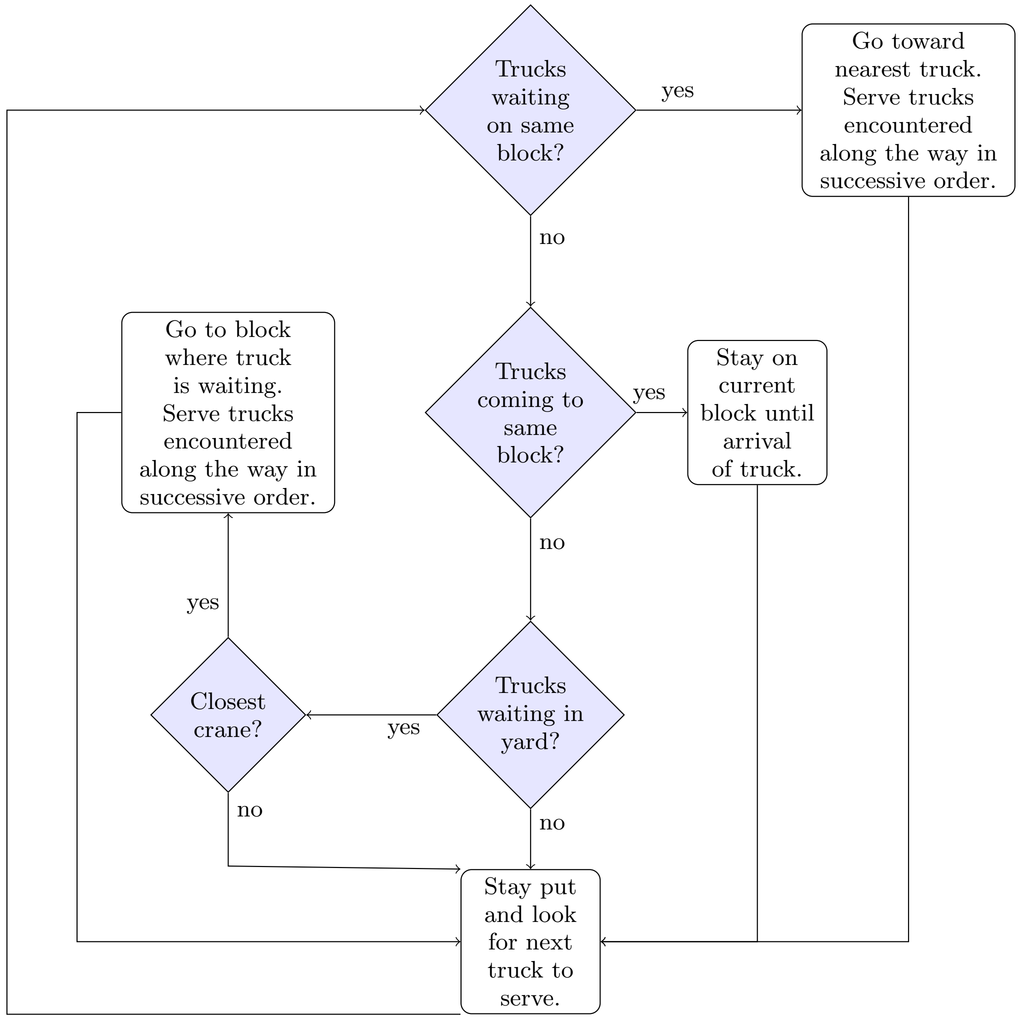


**Figure 4: Service strategy for yard cranes at the port of Houston.**

**Figure 3: Yard crane deposits container on a truck.**

next truck that makes the best sense based upon all the information they may know. At the Port of Houston, crane operators are also given the flexibility to use their judgment. In a series of interviews with different operators there, they appear to follow a strategy that is more distance-oriented (Figure 4).

To analyze the effectiveness of different yard crane service strategies, our initial models focus on stacked terminals equipped with RTGs and on the import drayage process. Also, we focus entirely on the container yard. We do not model the operations at the gate and berth (see Figure 5). We assume standard 40 foot long containers and yard blocks composed of 40 40-foot bays.

## 4. MODEL DESCRIPTION

We model the yard as a 2-dimensional grid where each cell fits exactly one container, which would make them about 40 feet wide by 10 feet tall. The yard contains 4 blocks each one with 40 bays, each bay has 6 rows of containers which can be stacked up to 4 high. The cranes move only on a specified track. The trucks appear below the bay where their desired container is located and are assumed to not interfere with the movement of the cranes. Figure 6 shows a screenshot of our

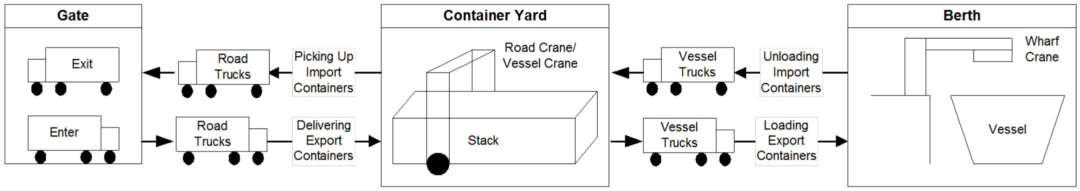


**Figure 5: Typical operations and processes at seaport container terminals.**

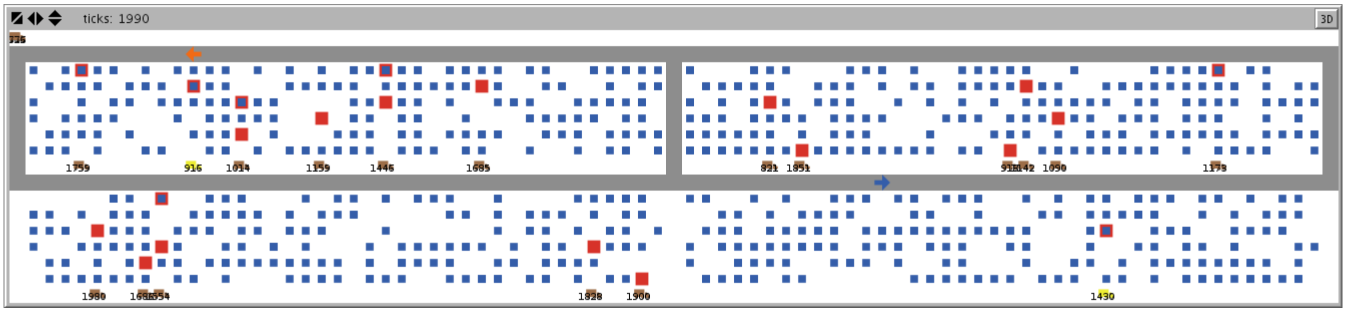


**Figure 6: Screenshot of our agent-based model of yard cranes at a seaport container terminal.**

implemented agent-based model. Notice that the cranes are represented by arrows which represent the direction in which the crane is moving. The track on which the cranes can move is represented by the gray lanes. Notice that the crane path allows the cranes to service all 4 blocks. Although not shown in Figure 6, our model includes several plots and other GUI elements which allow us to better understand what is happening in the program and to control its behavior.

Initially, the field is populated with randomly-placed containers. At run time the trucks arrive with a probability given by a Poisson distribution and each is assigned to a randomly chosen container. If the bay for that container is free then the truck is placed on that bay, otherwise the truck is placed on a waiting area. When the bay is freed, because the truck on the bay has been given the container it wanted, then the waiting truck is immediately moved to the bay.

We model the crane operators as utility-maximizing agents that can constantly re-evaluate their utility function. More formally, we say that there is a set $C$ of cranes so that each crane $c \in C$ has a utility function $u_c(t)$ over all trucks $t \in T$ in the yard. We consider several different utility functions, all of which weigh different aspects of the world. One of those factors is the shortest path between the crane and the truck. As can be seen in Figure 6, there are a number of different paths that a crane can take to get to a container, some much longer than others. Thus, we let $\textsc{path}(c,t)$ be the shortest path between crane $c$ and truck $t$. Similarly, we define $\textsc{distance}(p)$ to be the distance of a path $p$, $\textsc{has-turn?}(p)$ to be a Boolean function that returns 1 if path $p$ requires the crane to turn, that is, move from one of the top two blocks to one of the bottom two blocks or vice-versa, and $\textsc{other-crane?}(p,c)$ to be a Boolean function that returns 1 if $p$ passes over the current location of some crane other than $c$.

In our model, each crane $c$ has a current goal $g_c$ which can be either empty ($\emptyset$) or contain a truck $t$, which means that the crane's current goal is to go to the location of truck $t$, or it can have the value *deliver-container*, which means the crane is currently trying to move a container from the stack onto the truck. The cranes are opportunistic but we also implement a *decommitment-penalty* which can be set to 0 for completely opportunistic behavior or to larger numbers to make the cranes more committed to their current goal. Specifically, if $g_c = \emptyset$ or $g_c = t$ from some $t$, then the crane updates its goal at every time by first determining the optimal truck to service ($t^*$) and then switching to that truck only if its utility beats that of the current goal by more than the *decommitment-penalty*, as such:

$$t^* \leftarrow \arg_{t \in T} \max u_c(t) \tag{1}$$

$$g_c \leftarrow \begin{cases} t^* & \text{if } u_c(t^*) > u_c(g_c) + \textit{decommitment-penalty} \\ g_c & \text{otherwise,} \end{cases} \tag{2}$$

where we let $u_c(\emptyset) = 0$.

We consider three specific utility functions. The first one is a **distance-based** utility function which tries to capture the effective distance between the crane and a truck, giving higher priority to trucks that are closer to the crane. This distance is mostly just the path length between the crane and the truck, but also includes elements that consider the need for making a turn (as these take a longer time), the fact that there is another crane in the path (since then the path is blocked), whether or not the crane needs to change direction, and whether this crane is indeed the closest one to the truck. That last term provides the cranes with a slight implicit form of coordination. We believe that this utility function roughly captures what an operator means when he says he intends to always serve the closest truck. More formally, we define this utility as

$$u_c^{\text{distance}}(t) = -\textsc{distance}(\textsc{path}(c,t)) - p_1 \cdot \textsc{other-crane?}(\textsc{path}(c,t)) - p_2 \cdot \textsc{has-turn?}(\textsc{path}(c,t)) - p_3 \cdot \textsc{change-heading?}(\textsc{path}(c,t)) - p_4 \cdot \textsc{not-closest?}(c,t), \quad (3)$$

where the $p_1 \cdots p_4$ are fixed penalty constants. Their values are set to be high enough such that a crane will never choose a truck for which any one of the terms are true (that is, there is another crane on the way, or the crane must take a turn, or it must change its heading, or there is another crane closest) if there is another truck somewhere in the yard for which all terms are false. Note that the distance has a negative sign because the more a crane has to travel the less utility it receives from that delivery. Also, we let CHANGE-HEADING?($p$) be a Boolean function which returns 1 if the crane needs to change its current heading in order to follow path $p$, and we let NOT-CLOSEST?($c, t$) return 1 if crane $c$ is not the one currently closest to $t$, or 0 otherwise.

Similarly, we define a **time-based** utility function that gives higher priority to the trucks that have been waiting the longest, but also taking into account the other terms. Formally, the time-based utility is given by

$$u_c^{\text{time}}(t) = \textsc{wait-time}(t)) - p_1 \cdot \textsc{other-crane?}(\textsc{path}(c,t)) - p_2 \cdot \textsc{has-turn?}(\textsc{path}(c,t)) - p_3 \cdot \textsc{change-heading?}(\textsc{path}(c,t)) - p_4 \cdot \textsc{not-closest?}(c,t), \quad (4)$$

where WAIT-TIME($t$) is the time that truck $t$ has been waiting.

Finally, we define a **time-and-distance** utility function which merges these two into one, as such:

$$u_c^{\text{time-distance}}(t) = -\textsc{distance}(\textsc{path}(c,t)) + u_c^{\text{time}}(t) \quad (5)$$

In modeling the yard crane gantry speed and handling times, actual or empirical data are used. A typical yard crane can gantry (i.e. traverse along the yard block) at a speed of 135 meters per minute [13]. Thus, it takes a crane about 6 seconds to gantry from one 40-foot bay to the next. As mentioned previously, a truck's wait time is a combination of the time it takes a crane to arrive at the bay where the truck is parked and the time it takes the crane to perform both rehandling and delivery moves. The steps involved in performing a rehandle are as follows. These steps are repeated for every container that is sitting on top of the target container.

1. Position spreader bar on top of container to be rehandled
2. Lower the spreader bar
3. Lock the spreader bar to the container
4. Hoist the container
5. Trolley to the desired stack
6. Lower the container
7. Unlock the twist lock
8. Bring the spreader bar back to its normal position

The steps involved in performing a delivery move are similar to a rehandle move. The key difference is in step 5 where instead of setting a container onto a stack, the crane operator sets the container onto the truck, which could take much longer time if the truck is not properly positioned. If the target container is at the bottom of a stack that is four high, then a crane will need to perform three rehandling moves and one delivery move. Data gathered previously by the authors show that the average rehandling time to be about 40 seconds and the delivery time to be about 87 seconds.

## 4.1 Solution Concepts

Given the problem definition, there are several solution concepts we might consider. The most obvious one is maximizing the throughput of the port. That is, servicing the most trucks possible in a given fixed amount of time. By definition, this measure is the same as minimizing the total wait time of the trucks in that same fixed amount of time. However, it is possible that a solution that minimizes the total wait time does so at the expense of one, or a few, unlucky trucks who must spend a very long time in the truck. Thus, another solution concept tries to minimize the maximum wait time of a truck. This solution is more egalitarian and might thus be preferred by the truck drivers. A third possibility, one which we have not explored yet, would to try to even out the amount of work each crane performs so that they all do about the same amount of work.

# 5. MODEL IMPLEMENTATION

Our model is implemented in NetLogo [18], an agent-based simulation platform and programming language. We modeled four yard blocks, each one with 40 bays of 40-foot containers, and each stack has six rows of containers that can be stacked up to four high. The cranes can move around these four blocks and can position themselves at any bay. The model is implemented to work for any number of cranes. The containers are distributed randomly across the four blocks and are never stacked more than four high in any one row. No new containers are added during a run since we are only concerned with evaluating the cranes' strategies. We also implemented trucks, each of which is assigned a randomly chosen container. If there is another truck already waiting at the bay where the container resides then the truck is made to wait in a holding area until the other truck is serviced and departed, thus clearing the spot for the waiting truck. The waiting truck then takes its position on the next tick.

Our model implements a discrete simulation where every tick corresponds to one second of real-world time. At every tick, the model creates and positions any new trucks that might have arrived during that tick, asks the cranes to perform their chosen action for that tick, and updates the graphs and plots. Since the cranes' actions take more than one second to execute, our implementation incorporates wait times for each action. For example, it takes six seconds for the crane to move from one stack to the next one. Instead of having the crane move one sixth of the distance each time, our implementation makes it wait for the first five seconds and then perform the move on the sixth second. This delay technique is used for all other actions: moving a container *main*()

1 **while** user has not stopped program
2 **do** generate truck arrivals
3 **for** $c \in C$
4 **do** $g_c \leftarrow \emptyset$
5 $c$.GO()
6 $tick \leftarrow tick + 1$

*go*()

1 **if** $g_c \in T$ or $g_c = \emptyset$
2 **then** $t^* \leftarrow \arg_{t \in T} \max u_c(t)$
3 **if** $u_c(t^*) > u_c(g_c) + decommitment\text{-}penalty$
4 **then** $g_c \leftarrow t^*$
5 $g_c^t \leftarrow ticks\text{-}to\text{-}move$
6 **if** $g_c^t \neq 0$
7 **then** $g_c^t \leftarrow g_c^t - 1$
8 **return**
9 **if** $g_c \in T$
10 **then** move to the first in PATH$(c, t)$
11 **if** we are at $g_c$
12 **then** $g_c \leftarrow deliver\text{-}container$
13 $g_c^t \leftarrow ticks\text{-}to\text{-}deliver$
14 **else**
15 $g_c^t \leftarrow ticks\text{-}to\text{-}move$
16 **elseif** $g_c = deliver\text{-}container$
17 **then** take step in delivery
18 **if** container delivered
19 **then** $g_c \leftarrow \emptyset$

**Figure 7: Implementation methods.** MAIN **is the main loop and** GO **is a method implemented by every crane** $c$**. Note that** $u_x(\emptyset)$ **is assumed to be 0.**

from one row to another (40 seconds) and moving a container from a row to the truck (87 seconds). By using this wait technique, it is easy to change the times each action takes to suit the real-world data. It also lets the simulation display an accurate representation of what is happening in the model.

The implementation algorithm is shown in Figure 7. At every tick we first create any new trucks that might have arrived and assign them to their appropriate spots. The cranes are then asked to GO(). First, crane uses its function to determine which is the best truck for it to service. If the crane has a current goal of serving a truck or no goal and there is a truck with a utility greater that *decommitment-penalty* then the crane switches to that one, thus implementing (2), as shown in lines (1) – (5). Lines (6) – (8) implement the time delay (skipped ticks) required for some of the longer actions. In lines (9) – (15) the crane moves one step in the path towards its chosen goal and then either re-sets its goal or changes its goal to one of delivering the container to the truck. Finally, in lines (16) – 19 the crane has the goal of delivering a container and takes a step in delivering it. This step might require the crane to move other containers around in its current bay if the desired container lies under other containers. In these cases the crane will require more ticks to deliver the container.

Our current implementation is fast enough for our needs, but it could be improved. Our analysis shows that line (2) of the GO() procedure is where the simulation spends most of its time as it has to check every single truck in the yard. Of course, we do not need to check every truck as there are some fairly simple heuristics that could be used to eliminate some trucks from consideration. Future version of our implementation will include such heuristics, thus enabling us to test much larger yards in the same amount of time.

**Table 1: Simulation results for 2-crane scenario.**

**Distance-based** (3)

| De-commitment Penalty | Average Wait Time (minutes) | Min of Max wait time (minutes) |
|---|---|---|
| 0 | 14.37 | 41.30 |
| 100 | 15.42 | 37.93 |
| 10,000 | 15.04 | 45.65 |

**Time-based** (4)

| De-commitment Penalty | Average Wait Time (minutes) | Min of Max wait time (minutes) |
|---|---|---|
| 0 | 68.97 | 68.95 |
| 100 | 65.49 | 72.58 |
| 10,000 | 53.84 | 56.18 |

**Time-and-distance-based** (5)

| De-commitment Penalty | Average Wait Time (minutes) | Min of Max wait time (minutes) |
|---|---|---|
| 0 | 68.04 | 86.38 |
| 100 | 65.42 | 67.97 |
| 10,000 | 52.24 | 56.77 |

## 6. SIMULATION RESULTS

Our first tests simply made sure that the program will be fast enough to be usable. Our current implementation running on a standard PC is able to simulate a whole day (8 hours) in just a few seconds, as long as the graphics display is turned off. Showing the animation of the cranes moving around the field significantly slows down the simulation, but this is not a problem as the visualization is only used for debugging when we do need it to go slow enough so we can see what the cranes are doing.

We mentioned several solution concepts for this problem in Section 4.1. For our initial tests we decided to focus on the average wait time of the trucks and the wait time of the truck that waits the most. The lower the average wait time the more trucks are being served in a day, as we keep the arrival rates constant. The maximum wait time tells us how the least quickly served truck had to wait. We varied the *decommitment-penalty* from 0, which implies a completely opportunistic crane, to 100, to $10,000$ which implies a crane that once committed to a truck will never drop it for another one. Tables 1 and 2 show the average results from 100 runs for the various utility function and de-commitment penalties with trucks arriving following a Poisson distribution of with a mean of 40 trucks/hour. The tables show the average truck waiting time and the minimum of the maximum waiting time in each one of the runs, that is, the minimum of the list containing the maximum wait time for each run.

**Table 2: Simulation results for 3-crane scenario.**

**Distance-based** (3)

| De-commitment Penalty | Average Wait Time (minutes) | Min of Max wait time (minutes) |
|---|---|---|
| 0 | 6.53 | 19.05 |
| 100 | 6.82 | 20.78 |
| 10,000 | 6.77 | 19.90 |

**Time-based** (4)

| De-commitment Penalty | Average Wait Time (minutes) | Min of Max wait time (minutes) |
|---|---|---|
| 0 | 8.75 | 21.95 |
| 100 | 8.51 | 24.47 |
| 10,000 | 7.85 | 21.27 |

**Time-and-distance-based** (5)

| De-commitment Penalty | Average Wait Time (minutes) | Min of Max wait time (minutes) |
|---|---|---|
| 0 | 7.85 | 21.07 |
| 100 | 7.88 | 22.37 |
| 10,000 | 8.11 | 19.58 |

In Table 1 we note the surprising result that the average wait time for the time-based and time-and-distance-based utility functions is nearly four times as large as that of the distance-based utility. The reason for this is evident when viewing the simulation. When crane operators worked to minimize trucks' waiting time, they ended up making long runs from one end of the yard to another while ignoring nearby trucks. The model indicates that, on average, the two cranes covered a total distance of 16.25 miles when following the distance-based utilities and 25.41 miles when following the time-based utilities. The resulting effect is that many more trucks end up waiting longer.

Another surprising discovery from this study is how effective the distance-based utility is in minimizing the maximum waiting time across all the runs. It was expected that the time-based utility with the de-commitment penalty set to 10,000 would yield the lowest min-max wait time because the cranes would effectively "chase" after these longer waiting trucks. As shown in the third column of Table 1, the min-max wait times of the time-based utilities are higher than that of distance-based utilities. As explained above, when the cranes "chase" after the longer waiting trucks, they are less efficient because they are spending more time traveling to their target trucks. It would have been more efficient if they use that time to serve nearby trucks.

Table 2 shows the wait time and min-max wait time results when there are three cranes available. Note the significant drop in the average wait time and min-max wait time across all three utility types. It is also interesting to note that with three cranes, the performance of the time-based utilities is very close to that of the distance-based utilities. This is because cranes do not have to cover as much distance with three cranes. The model indicates that, on average, the three cranes covered a total distance of 13.65 miles when following the distance-based utilities, 15.47 miles when following the time-based utilities, and 16.46 miles when following the time-and-distanced-based utilities.

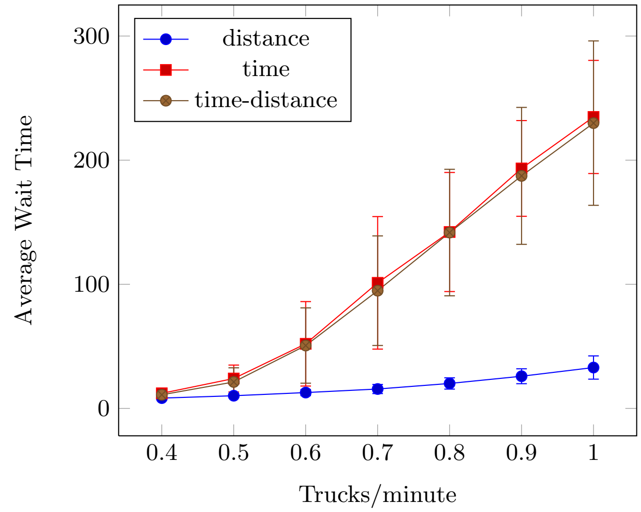


**Figure 8: Average truck waiting time (over 100 runs) as a function of arrival rate for the three utility functions, with a *decommitment-penalty* of 0 and 2 cranes. The x-axis is the mean of a Poisson arrival process. The error bars represent the maximum and minimum wait time from all 100 runs.**

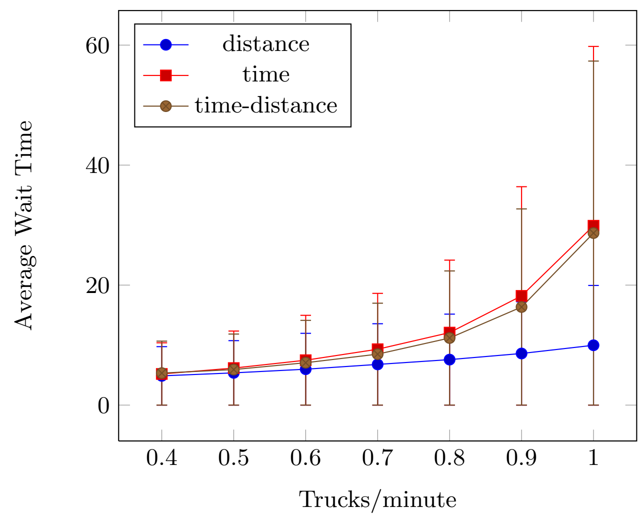


**Figure 9: The same experiment as Figure 8 but with 3 cranes instead.**

After establishing that the distance-based utility performed the best overall, we were curious as to how it would fare as the truck arrival rate changed. Thus, we performed the same experiments as with 2 cranes but we varied the truck arrival rate from .4 trucks/minute to 1 truck/minute, where our previous results used a fixed rate of .667 trucks/minute (40 trucks/hour). The results of our tests are shown in Figure 8. As expected the distance-based utility performs best but it is noteworthy how flat its curve is while the other time-based utility functions explode as trucks arrive faster. On the other hand, as the arrival rate is made smaller, to .4, the difference between the various utility functions almost

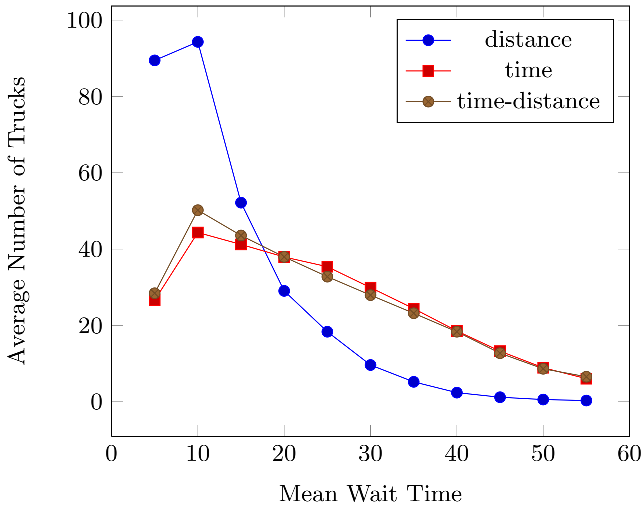


**Figure 10: Distribution of average wait times with an mean truck arrival rate of .5.**

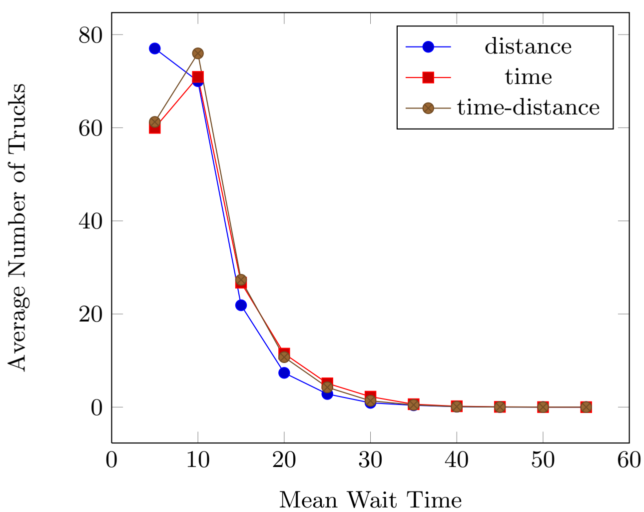


**Figure 11: Distribution of average wait times with an mean truck arrival rate of .3.**

disappears. The qualitative aspects of these results are not surprising. However, by using our agent-based model we can quantify exactly how much better one utility function is over the others given the specifics of the problem. Thus, terminal operators can use the data generated by our model to make business decision on how to direct their cranes.

Similarly, Figure 9 shows the results of the same experiment but with 3 cranes instead. As we might expect, three cranes are able to handle higher truck arrival rates thus there is a smaller, but still significant, difference between the distance-based utility function and the other ones.

Our final experiment shows the wait time distribution among the trucks. Figure 10 shows how many trucks, averaged over 100 runs, had to wait 0-5, 5-10, 10-15 minutes and so on till 55 minutes, the maximum wait time. We see that for the distance utility a great number of trucks does not have to wait more than 10 minutes, and only a small percentage has to wait more than 20 minutes. The time utilities, on the other hand, distribute wait times a bit more evenly. That is, a greater number of trucks have to wait longer. In Figure 11, we decreased the arrival rate to 0.3. As we saw in earlier results, for such low arrival rates we can expect the average wait times to be the same. The distributions largely confirm this; however, we do see that the distance utility yields about 20 more trucks with a wait time of less than 5 minutes. Another interpretation of this result is that when following the time utilities, in their effort to serve trucks in a first-come first-serve manner, the cranes miss the opportunities to serve nearby trucks.

## 7. CONCLUSION

This study introduced an agent-based utility maximization approach to modeling yard cranes at seaport container terminals to study how different service strategies affect truck turn time. The developed model provides a powerful tool terminal operators could use to assess the performance of various contemplated crane service strategies as well as the effect of having additional cranes or fewer cranes due to mechanical problems and/or scheduled maintenance. This study has identified a set of utility functions that properly captured the essential decision making criteria of crane operators in choosing the next truck to provide service to. Simulation results showed that if crane operators choose trucks that are closest to them without requiring the cranes to turn often (a time consuming process) and reverse heading, then the overall system performance in terms of average waiting time and the maximum waiting time of any truck will be better than if there were to choose trucks based on their waiting times.

Implementing the mentioned agent-based simulation model revealed some important lessons in modeling cranes as agents. Initially, we implemented the crane behaviors as procedures (e.g. choose nearest truck or choose longest waiting truck). While these procedures were easy to implement in NetLogo, as we incorporated additional complexities into the operators' decision making process, the procedures became unwieldy. The procedures ended up implementing ad-hoc rules which we could not fully explain or justify. For these reasons, we changed our approach to use utility functions and made the cranes utility-maximizing agents. By using utility functions we can clearly and explicitly capture how the cranes balance the various priorities: distance to truck, time truck spent waiting, etc. A caveat here is that the utility functions can make it harder to implement certain procedural knowledge, like "move to the closest truck and then keep going in that direction if there are more trucks waiting right behind that one." In this study, we have identified a set suitable utility functions and built a first model that implements these.

In future work, we plan to extend the model to handle larger yards and include explicit coordination with the incoming trucks, perhaps in the form of reservations or auctions. We hope to eventually build a detailed simulator of several yards as well as the trucks moving between them and their respective delivery sites. Such large-scale simulation will give us the ability to model truck traffic across a wide geographic span and see how it affects, or is affected by, seaport operations.

## 8. REFERENCES

[1] K. Dresner and P. Stone. A multiagent approach to autonomous intersection management. *Journal of Artificial Intelligence Research*, 31(1):591–656, 2008.

[2] L. M. Gambardella, A. E. Rizzoli, and M. Zaffalon. Simulation and planning of an intermodal container terminal. *Simulation*, 71:107–116, 1998.

[3] P.-L. Grégoire, C. Desjardins, J. Laumonier, and B. Chaib-draa. Urban traffic control based on learning agents. In *Proceedings of the 10th Internationnal IEEE Conference on Intelligent Transportation Systems*, 2007.

[4] L. Henesey, P. Davidsson, and J. A. Persson. Agent based simulation architecture for evaluating operational policies in transshipping containers. *Autonomous Agents and Multi-Agent Systems*, 18(2):220–238, 2009.

[5] L. Henesey, P. Davidsson, and J. A. Persson. Evaluation of automated guided vehicle systems for container terminals using multi agent based simulation. In *Multi-Agent-Based Simulation IX: International Workshop*, pages 85–96. Springer-Verlag, Berlin, Heidelberg, 2009.

[6] K. Kim. Sequencing delivery and receiving operations for yard cranes in port container terminals. *International Journal of Production Economics*, 2003.

[7] D. Lee, Z. Cao, and Q. Meng. Scheduling of two-transtainer systems for loading outbound containers in port container terminals with simulated annealing algorithm. *International Journal of Production Economics*, 107(1):115–124, 2007.

[8] K. Lewis. *Decision making in engineering design*. ASME Press, New York, 2006.

[9] C. Macharis. Opportunities for OR in intermodal freight transport research: A review. *European Journal of Operational Research*, 153(2):400–416, 2004.

[10] W. Ng. Crane scheduling in container yards with inter-crane interference. *European Journal of Operational Research*, 164(1):64–78, 2005.

[11] W. Ng and K. Mak. Yard crane scheduling in port container terminals. *Applied Mathematical Modelling*, 29(3):263–276, 2005.

[12] M. Rebollo, V. Julian, C. Carrascosa, and V. Botti. A MAS approach for port container terminal management. In *Proceedings of the Third Iberoamerican workshop on DAI-MAS*, 2001.

[13] D. Thurston. Utility function fundamental. In *Decision Making in Engineering Design* [8].

[14] T. Thurston and H. Hu. Distributed agent architecture for port automation. *Computer Software and Applications Conference, Annual International*, 0:81, 2002.

[15] K. Tumer and A. Agogino. Improving air traffic management with a learning multiagent system. *Intelligent Systems*, 24(1), Jan/Feb 2009.

[16] K. Tumer, A. K. Agogino, and Z. Welch. Traffic congestion management as a learning agent coordination problem. In A. Bazzan and F. Kluegl, editors, *Multiagent Architectures for Traffic and Transportation Engineering*, pages 261–279. Lecture notes in AI, Springer, 2009.

[17] M. Vasirani and S. Ossowski. A market-inspired approach to reservation-based urban road traffic management. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems*, pages 617–624, Richland, SC, 2009. International Foundation for Autonomous Agents and Multiagent Systems.

[18] U. Wilensky. NetLogo: Center for connected learning and computer-based modeling, Northwestern University. Evanston, IL, 1999. `http://ccl.northwestern.edu/netlogo/`.

[19] C. Zhang. Dynamic crane deployment in container storage yards. *Transportation Research Part B: Methodological*, 36(6):537–555, 2002.

# Occlusion-aware Multi-UAV Surveillance of Multiple Urban Areas

Michal Jakob, Eduard Semsch, Dušan Pavlíček and Michal Pěchouček
Agent Technology Center, Dept. of Cybernetics, FEE, Czech Technical University
Technická 2, 16627 Prague 6, Czech Republic
{jakob, semsch, pavlicek, pechoucek}@agents.felk.cvut.cz

## ABSTRACT

We present an agent-based coordination and planning method for aerial surveillance of multiple urban areas using a group of fixed-wing unmanned aerial vehicles (UAVs). The method differs from the existing work by explicit consideration of sensor occlusions that can occur due to high buildings and other obstacles in the target area. The solution employs a decomposition of the problem in two subproblems: the problem of single-area surveillance and the problem of allocating UAVs to multiple areas. Three occlusion-aware methods for single-area surveillance are presented and compared. An algorithm for UAV allocation is presented and its optimality proved. The performance of all algorithms is evaluated empirically on a realistic simulation of aerial surveillance, built using the AgentFly framework, and is compared to theoretical estimates.

## Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Coherence and coordination, multiagent systems*

## General Terms

Algorithms, Performance

## Keywords

autonomous aircrafts, UAV-based surveillance, UAV control, resource allocation, simulation, sensor occlusions

## 1. INTRODUCTION

There has been a growing interest in using unmanned aerial vehicles (UAVs) for information collection tasks, initially in military and later also in civil domains [2]. With the increasing numbers of UAVs, there is a growing need to enable the UAVs to perform their information collection missions autonomously without the need for direct human control, which is costly. Intelligent multi-agent techniques have been employed to address this problem [9][1].

*Area surveillance* is one of the most common information collection tasks, typically defined as a problem maintaining an up-to-date picture of the situation in a given area. Sometimes the task is termed *persistent* surveillance to highlight the fact that the area is to be monitored for a prolonged period of time.

In this paper, we address a particularly challenging variant of the problem – controlling a team of autonomous UAVs performing persistent surveillance of *geometrically complex environments* such as those present in dense urban areas. In such environments, the field of view of UAV's on-board sensors can get occluded in the presence of tall buildings and/or narrow streets (see Figure 1). This can result in areas left uncovered, which might be exploited by an adversary. The problem is likely to get more acute in the future as small, low-flying UAVs are going to be deployed.

In this paper, we address the problem by providing a multi-agent coordination mechanism that realistically models and explicitly eliminates the effect of occlusions. Although the problem of occlusions has been studied in other contexts [6, 12], *occlusion-aware* surveillance has not yet been considered in the field of autonomous UAV control. In addition to providing a solution to the problem of occlusions, we also show how multiple disjoint areas can be surveilled, which is novel too.

The paper proceeds as follows. In Section 2, we formally define the problem of UAV-based surveillance of occlusion-affected environments. In Section 3, we introduce our two-stage method; its two constituent components are described in two subsequent sections – single area surveillance in Section 4 and multiple-area surveillance in Section 5. Section 6 provides evaluation results, Section 7 reviews related work and, finally, Section 8 concludes.

## 2. MULTI-UAV INFORMATION AGE MINIMIZATION PROBLEM

We formally define multi-UAV area surveillance as a constraint optimization problem of finding a set of trajectories for a group of UAVs which, when followed, minimize the average age of information collected about a set of points of interest located within one or multiple disjoint areas.

### 2.1 Problem Domain

The environment is a set $\mathcal{A} = \{A_1, A_2, \ldots, A_k\}$ of $k$ rectangular areas $A_i$, each area $A_i$ defined as a quadruple $A_i = \langle x_0, y_0, w, h \rangle$, where $x_0, y_0$ are the coordinates of its bottom left corner and $w, h$ are its width and height. In each of the areas, a set of buildings may be present represented as quadrilateral prisms with their bases on the ground plane (defined by equation $z = 0$.) Furthermore, for each area $A_i$ there is a finite set of *points of interest* $P_i = \{p_1, p_2, \ldots, p_m\}$ located inside the respective area[1]; these points should be seen as often as possible.

We control a fleet of fixed-wing UAVs $U = \{u_1, u_2, \ldots, u_n\}$. The UAVs are modeled as point masses moving with a con-

[1]Typically, the points of interest will be uniformly distributed within the target area

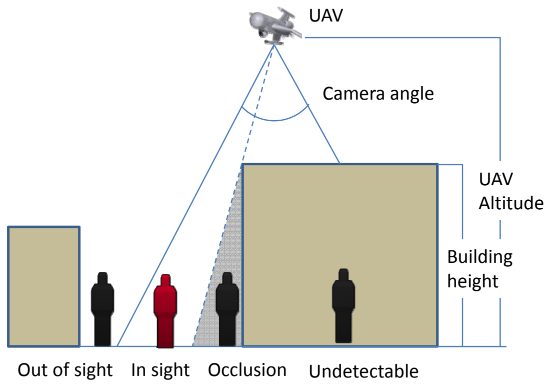


**Figure 1: Occlusions in urban environment.**

stant speed $v$ and capable of turning with a minimum turning radius $R$ (such a model is generally referred to as *Dubins vehicle* [7]). Each UAV carries a sensor of a conical field of view pointing down to the ground[2], with apex at the position of the UAV, and the field of view angle $\varphi$. The sensor is capable of observing ground points which are inside the sensor's field of view and are *not occluded* by a building. The situation is illustrated in Figure 1. We define the function $\tau(p,t)$ as the last moment in time prior to time $t$ when a point of interest $p$ was seen by a UAV. If the point has not yet been seen, we set the value to 0.

## 2.2 Objective Function

For a time instance $t$ and a point of interest $p$, we call the value $t - \tau(p,t)$ the *information age* of $p$ at time $t$. The area surveillance problem is then to minimize the average information age of all points over a period of time, i.e., to minimize the expression

$$\frac{1}{t_1 - t_0} \frac{1}{|P|} \sum_{t=t_0}^{t_1} \sum_{i=1}^{k} \sum_{p \in P_i} (t - \tau(p,t)), \tag{1}$$

where discrete time model is assumed, and $t_0$ is the time at the beginning of the evaluation period (typically zero), $t_1$ at the end of the period, and $|P|$ is the number of points of interest. We term this objective function the *information age* objective function and the resulting optimization problem the *information age minimization problem.*

A solution of the problem is a set of flight trajectories for all the UAVs; the trajectories must respect the minimum turn radius of the UAVs.

# 3. TWO-STAGE MULTI-AREA SURVEILLANCE

The information age minimization problem is an instance of constraint optimization problems which are known to be generally intractable [16]. It is also known that the traveling repairman problem for Dubins vehicle, a special case of the single-area information age minimization, is NP-hard [13].

We therefore propose an approximate solution consisting of two stages:

1. Allocate UAVs to the areas. As a result, each UAV will have exactly one area assigned. Multiple UAVs can be assigned to one area.

[2]a tilting camera mount is assumed

2. For each area separately solve the single-area information age problem employing the allocated UAVs.

Note that we assume that the number of areas is not higher that the number of UAVs – an extension to the general case is possible but not presented in this paper. We describe the two stages in the following two sections, starting with the single-area surveillance algorithm.

# 4. SINGLE-AREA SURVEILLANCE

We now describe how a single rectangular area may be surveilled by one or more UAVs in a way that gurantees 100% coverage of all the points of interest in the area. We start with solving the problem for a single UAV; a straightforward extension to multiple UAVs is given in Section 4.5.

## 4.1 Single-UAV Surveillance

Assuming the structure of the surface to be a composition of quadrangular prisms, there always exists a finite set of points in the air, all lying at the same altitude, such that every point on the surface can be seen from at least one of the points in the set (for proof see e.g. [3]). We term any such set a *covering vantage point set.* We can then decompose the construction of an UAV's flight trajectory into two steps:

1. Finding a covering vantage point set.
2. Finding the shortest trajectory travelling all the vantage points

Depending on the algorithm, the two steps can be performed in a serial order or combined into a single step.

The problem of finding a covering set of vantage points can be viewed as an instance of the *3D Art Gallery Problem*[3], which has been shown to be NP-hard [8]. An approximation approach is thus frequently employed, consisting of discretizing the surveillance area and the area where the sensors can be placed, computing the visibility between the two sets, and finding a minimum set cover. While computing the minimum set cover is also a hard problem, efficient approximation algorithms exist.

We have developed and implemented three algorithms for single-area surveillance: (1) alternating algorithm, (2) spiral algorithm and (3) ziz-zag algorithm. The first two algorithms separate the covering vantage point set generation from the construction of the flight trajectory; the zig-zag algorithms combines both steps together.

## 4.2 Alternating Algorithm

Alternating algorithm introduced in [13] is an approximation algorithm for solving the *Dubins vehicle travelling salesman problem* (DTSP) with known upper and lower bounds on solution quality. The algorithm works by employing an optimal solver for (standard) Euclidean TSP and then augmenting the solution by calculating suitable heading vectors for each waypoint.

The application to the occlusion-aware surveillance is preceded by generating a covering set of vantage points through

[3]The art gallery problem ammounts to finding the minimal number of sensors and their positions in a polygonal area with or without polygonal holes such that any point inside the polygonal area can be seen by at least one sensor. In the basic formulation, the sensors are assumed to be omnidirectional.

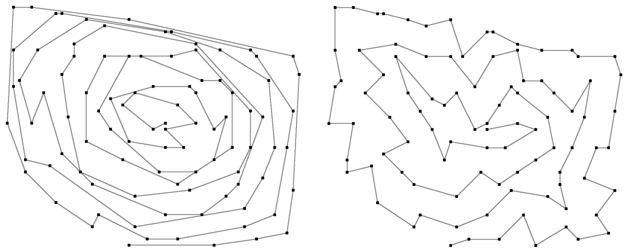

**Figure 2: Spiral trajectory consisting of non-relaxed convex hulls (left) vs. relaxed hulls (right); the latter is 40% shorter.**

which the UAV is supposed to pass. The set is then given to the alternating algorithm which produces the order in which the points should be visited. The resulting sequence determines the trajectory of the surveilling UAV.

In our implementation, we have used the freely available Euclidian TSP solver LINKERN[4] and fed the resulting waypoint sequence into AGENTFLY path planner [15] to find the shortest path respecting UAV constraints.

## 4.3 Spiral Algorithm

We have developed the spiral algorithm as a lightweight alternative to the alternating algorithm. Similarly to the alternating algorithm, the application of the spiral aglorithm also requires that a covering set of vantage points is first generated. Given a set of vantage points, the algorithm arranges the points in such an order that they can traversed using a spiral-like path respecting the UAV's turn radius constraints.

The algorithm is iterative. Given a set of points, the first iteration constructs a convex hull of the set and the points forming the boundary of the hull are then removed from the set. The remaining points are then used as the input for subsequent iterations. The process is repeated until there are no points left. The chain of hulls obtained is then linked together, starting with the outer-most hull. For each pair of neighboring hulls we search the shortest possible link, i.e. a pair of points, through which the hulls can be connected and merged.

The basic algorithm is further improved by relaxing the convex hulls (see function `RelaxConvexHull` in the pseudocode). The idea is to include as many points into each hull as possible while still keeping the hull smooth enough for the UAV to fly through. After constructing each convex hull, we inspect whether there are any points inside the hull that are close enough to the hull's edge and could be added into the (relaxed) hull without the UAV having to change its current course too much. If such a point is found, the convex hull is relaxed[5]. This relaxation process is executed recursively for each edge of the hull. Further details on the algorithm can be found in [14, 5].

## 4.4 Zig-zag Algorithm

The input to the zig-zag algorithm is the target area with the list of points of interest. In contrast to the previous two algorithms, the zig-zag algorithm does not require an a priori generated set of vantage points – the consideration of occlusions is performed *simultaneously* with planning the trajectory.

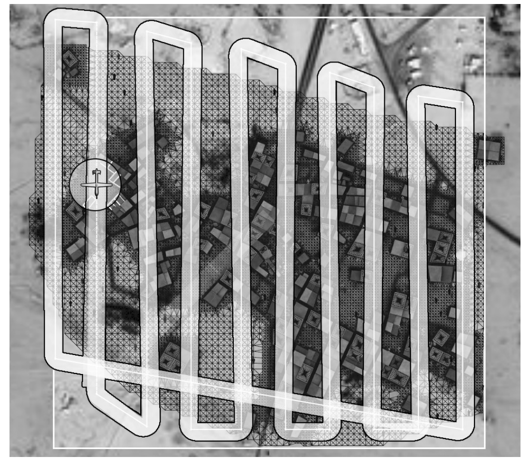

**Figure 3: An example trajectory generated by the zig-zag algorithm.**

The algorithm produces zig-zag trajectories (see Figure 3) with the number of rows minimized and the spacing between adjacent rows variable and optimized to minimize the overall trajectory length while ensuring that all ground nodes within the area of interest will be visible to the UAV's sensors (considering occlusions).

We first describe a single-UAV version of the algorithm. The algorithm takes a rectangular area of interest as its input. A regularly-spaced air navigation grid is created above the area of interest; the grid serves as a basis for occlusion-aware planning of UAV trajectories. The longer of the target area's two sides is determined; all subsequently generated trajectory rows will be parallel to that side (vertical in the case shown in Figure 3). Next, ground coverage generated by a flight along each such a row of nodes is determined, taking into account occlusions. The algorithm then iterates through individual rows and compares the set of ground nodes covered by the current row with the sets of ground nodes covered by two adjacent rows (previous and next). If all the ground nodes covered by the current row are also covered by its two adjacent rows, the current row is marked as redundant and removed. The process continues until no further row can be eliminated. See the pseudocode of the algorithm in Algorithm 1.

The algorithm produces a set of flight rows which guarantee the coverage of all points of interest. Having found the rows, a feasible trajectory for a UAV with a defined minimum turn radius $R$ can be constructed – the rows are straight segments of the trajectory, and the shortest trajectory connecting the end points of two adjacent rows can be constructed in constant time using the method of Dubins [4].

## 4.5 Extension to Multiple UAVs

There are three comparably effective ways in which the single-UAV algorithms can be extended for multiple UAVs, each with its specific advantages:

1. All UAVs travel along the whole trajectory with equal spacing between the consecutively following UAVs. The advantage of this approach is that it decreases the in-

[4] `http://www.tsp.gatech.edu/concorde/`
[5] Note that the resulting polygons are no longer convex

```
input  : area, ground_nodes, air_grid
output: asetofflightrows
begin
    rows ← GetGridRowsForArea(air_grid, area);
    foreach row in rows do
        coverage(row) ←
            CalculateRowCoverage(row,
        ground_nodes, area);
    end
    foreach curr in rows do
        prev ← GetPreviousRow(curr, rows);
        next ← GetNextRow(curr, rows);
        c ← coverage(curr) \ (coverage(prev) ∪
        coverage(next))
        if c = ∅ then
            remove curr from rows;
        end
    end
    return rows
end
```

**Algorithm 1**: Occlusion-aware zig-zag algorithm

formation age consistently across the whole area. A disadvantage is the initial coordination of the UAVs required to ensure their equal time spacing.

2. The rows covering the target area are divided between the UAVs. This approach is less effective than the first one because it is not always possible to divide the rows evenly. On the other hand, once the rows are distributed, no inter-UAV coordination process is required.

3. The target area is divided evenly between the UAVs and each UAV is then left to plan the zig-zag trajectory on its dedicated part of the area. This scheme has the advantage of simplicity and robustness againts communication failures as it does not rely on any coordination except for the straightforward division of the area.

Option 3 has been adopted in the presented work.

# 5. MULTI-AREA UAV ALLOCATION

We now describe the second building block of the two-stage surveillance – the multi-area allocation algorithm which distributes the available UAVs to the areas of interest, where they are subsequently controlled by the single-area zig-zag algorithm presented in the previous section (the other two algorithms could be used too).

## 5.1 Multi-Area UAV Allocation Problem

As discussed in Section 3, we assume there are more UAVs than areas. For this case, we define the *multi-area UAV allocation problem*.

Let $a : \mathcal{A} \to \mathbb{N}^0$ be the assignment function that specifies how many UAVs are allocated to a particular area:

Definition 1. *The estimated age function of an assignment $a$ is the function $E : a \to \mathbb{R}$ defined as:*

$$E(a) = \sum_{i=1}^{n} \frac{I(A_i)}{a(A_i)} \tag{2}$$

*where $I(A_i)$ is the estimated average information age for area $A_i$ if the area was assigned one UAV only.*

The average information age of an area $I(A)$ can be estimated using a lower bound calculated from the dimensions of the area. For an area of proportions $w, h$, in which points of interest are regularly distributed, and a UAV which travels at constant velocity $v$ in altitude $a$ having a sensor with ground radius $\rho = 2 \cdot a \cdot \sin(\frac{\varphi}{2})$, the lower bound on the quality of solution of the information age problem is

$$\underline{\mathrm{I}}(A) = \frac{1}{v}\left(\frac{h \cdot w}{2\rho} - \frac{\rho}{\pi}\right) \tag{3}$$

The first term inside the parenthesis is proportional to the number of zig-zag passes that the UAV has to make to see the whole rectangle without any occlusions. The second term compensates for the fact that due to sensor's circular ground footprint, points closer to the center of UAV's trajectory are visible longer.

The multi-area UAV allocation problem then amounts to find such an assignment $a^*$ that the corresponding estimated age $E(a^*)$ is minimal.

## 5.2 Allocation Algorithm

In this section, we show that an optimum assignment $a^*$ can be found using a greedy algorithm with $O(n^2)$ time complexity. The algorithm Algorithm 2 chooses the next step based on the best improvement of the information age estimate $E(a)$ (line 8).

```
     input  : 𝒜, U
     output: a
 1   begin
 2       // initialization
 3       for i = 1 to |𝒜| do
 4           a(A_i) ← 1
 5       end
 6       // main loop
 7       for i = 1 to |U| − |𝒜| do
 8           j =
             arg min_k (E(a) − E(a_[a(A_k)←a(A_k)+1]))
 9           a(A_j) ← a(A_j) + 1
10       end
11       return a
12   end
```

**Algorithm 2**: A greedy algorithm for the multi-area UAV allocation problem.

Theorem 1. *Algorithm 2 is optimal.*

*Proof.* The algorithm first assigns one UAV to each task. This is done in $|\mathcal{A}|$ number of steps. Such an assignment is possible because $|U| \geq |\mathcal{A}|$. Let us call $l$ the number of UAVs left unassigned after the first phase. We treat the two cases (i) $l = 0$ and (ii) $l > 0$ separately.

(i) If $l = 0$ the assignment is optimal. Because there is one UAV per task, the estimated age of the assignment is finite. For any other assignment, the estimated age would be infinite, i.e., bigger.

(ii) If $l > 0$. We consider the following construction: For each task $A_i$, we define its *age improvement function* $E^+_{A_i}$ :

$\mathbb{N} \rightarrow \mathbb{R}^+$ as

$\mathrm{E}^+_{A_i}(n) = I(A_i) \cdot \left(\frac{1}{n-1} - \frac{1}{n}\right)$ (4)for $n \geq 2$ and

$$E^+_{A_i}(n) = 0$$

for $n = 1$. The age improvement function represents the improvement of the estimated age function $E$ if the number of UAVs assigned to $A_i$ is increased from $n-1$ to $n$. Let us note that, in each step, the allocation algorithm increases the number of UAVs assigned to the task with the highest current value $E^+_{A_i}(a(A_i)+1)$ of the age improvement function. Finally, let us note that the final estimated age of the whole assignment is equal to

$$\sum_{i=1}^{|\mathcal{A}|} \left( I(A_i) - \sum_{j=1}^{a(A_i)} E^+_{A_i}(j) \right). \quad (5)$$

From this fact and from the fact that the age improvement function is strictly decreasing, it follows that choosing the best improvement at each step leads to the assignment with the minimum estimated age. □

The time complexity of the algorithm is $O(n^2)$ where $n$ is the number of UAVs. More precisely, the complexity is $O(|U| \cdot |\mathcal{A}|)$ because the main loop is executed exactly $|U|$ times and in each execution the maximum improvement of the cost function is calculated, which requires $|\mathcal{A}|$ atomic computations.

# 6. EXPERIMENTAL RESULTS

To evaluate the performance of the two-stage approach and the quality of produced trajectories, we have carried out a number of experiments, presented in the following two subsections. The first Section 6.1 presents results for single-area surveillance; the second Section 6.2 presents results for the integrated two-stage method – combining the multi-area allocation and the zig-zag algorithms – used for multi-area surveillance.

All evaluation was performed using the AgentFly framework [10] for UAV flight and air traffic simulation. The core framework – consisting of UAV flight model, accelerated flight path planning and collision avoidance – has been extended with a realistic on-board sensor model which accurately simulates the effect of occlusions. A screenshot of the simulation testbed in operation is given in Figure 4.

## 6.1 Single-Area Experiments

We first describe the experimental settings and then present two sets if experimental results. The first set evaluates the performance for a single UAV; all three single-area algorithms are tested. The second set evaluates the multiple UAV case, specificaly the dependence of the average information age on the number of employed UAVs; only the best performing zig-zag algorithm is used for this experiment.

### *Experimental Setting*

The specific scenario used for the evaluation is modeled after a real-world settlement with surroundings located in a flat 1500m-by-1500m square area. Buildings are modeled as non-overlapping but possibly adjacent quadrilateral prisms with bases on the $z = 0$ plane. There is a total of 300 buildings with heights in the 6-to-22 m range; the width of the streets range from 3 to 10 meters. The whole 1500m-by-1500m village area is to be surveilled. A visualization of the height map used for the experiments is depicted in Figure 8.

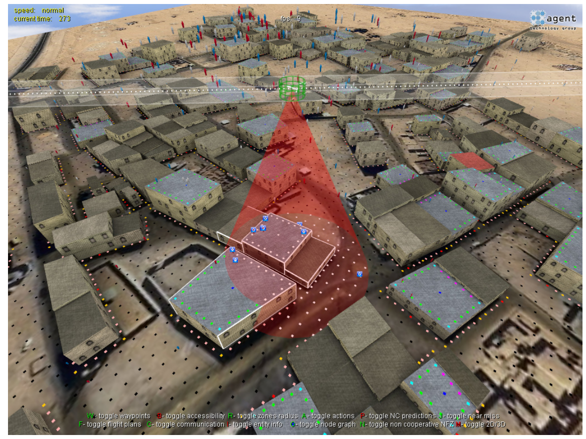

**Figure 4: AgentFly UAV simulation testbed with an occlusion-aware sensor model.**

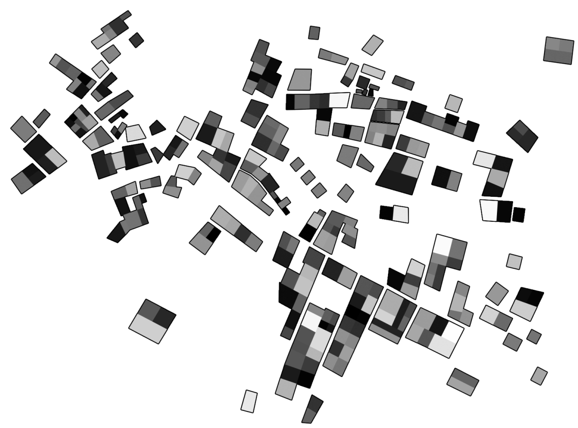

**Figure 5: Height map of the urban area used in the empirical evaluation. Highest buildings depicted in very light gray (22m), lowest buildings in black (6m).**

There are a number of configurable parameters of the scenario, summarized in Table 6.1 including the range within which they were varied.

| | |
|---|---|
| Number of UAVs | 1–12 |
| UAV altitude | 50–300 m |
| UAV minimal turning radius | $R =$ 0m-50m |
| Sensor field of view angle | $\varphi = 47^\circ$ |
| UAV speed | $v = 25$ m/s |

The points of interest consisted of a uniform grid covering all ground stretches (streets, intersections, and open areas) of the target area. The distance between the points in the grid was 5m.

### *Single-Area Single UAV*

The average information age for a single UAV surveillance was evaluated for all three occlusion-aware methods (Section 4). The results for different UAV flight altitude and

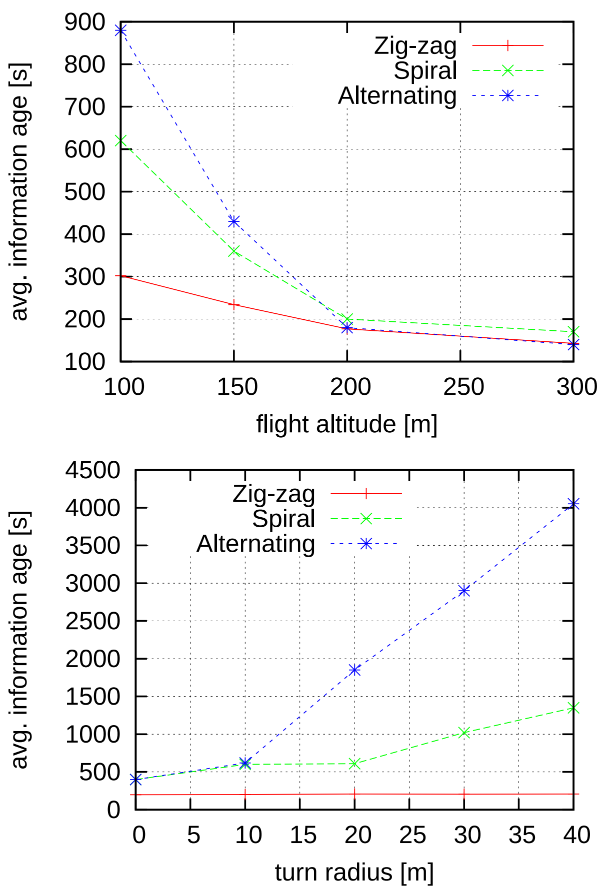


**Figure 6: Performance of single-UAV single-area surveillance for different UAV's flight altitudes (top) and UAV's minimum turn radii (bottom).**

minimum turn radius are given in Figure 6, respectively.

The results show the improving performance of the algorithm with the increasing UAV's altitude. In addition, the alternating and spiral algorithms show strong sensitivity on UAV's turn radius – higher values significantly worsen their perfromance, in particular for the alternating algorithm due to decreased maneuverability of the UAV. In contrast, thanks to composing the flight trajectories from straight row segments, the zig-zag algorithm is virtually unaffected (there is a slight increase if the turn radius is higher that the row spacing).

### *Single-Area Multiple UAVs*

Next we evaluated the performance of single-area surveillance using multiple UAVs, employing the solution proposed in Section 4.5. We only present results for the best performing zig-zag algorithm.

Our literature review has not identified a suitable candidate for comparison for the case of multiple UAVs in a single area. In order to give at least some indication of performance, we compare the information age with the information age estimate $\underline{\mathrm{I}}(A)$ (3). For multiple UAVs in one area, the estimate is divided by the number of UAVs. Figure 7 shows the measured performance of the multi-UAV

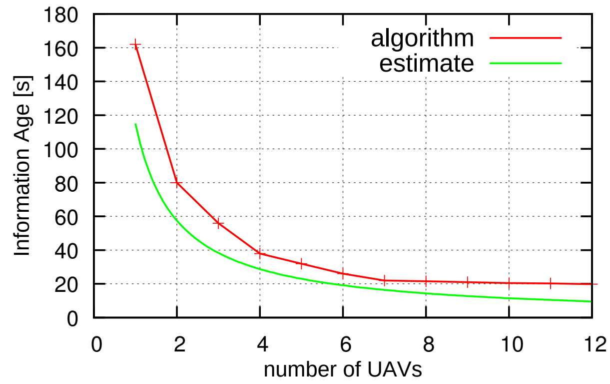


**Figure 7: Information age, measured and estimated, for multi-UAV single-area surveillance.**

algorithm and the corresponding estimate $\frac{1}{n}\underline{\mathrm{I}}(A)$[6].

## 6.2 Multi-Area Experiments

Next we evaluated the two-stage multi-area surveillance algorithm employing the zig-zag algorithm as its base single-area occlusion-aware planner. Our main objective was to understand the dependency of the algorithm's performance on the number and size of the target areas. Similarly to the multi-UAV single-area surveillance, there is no existing algorithm suitable for direct comparison, and we therefore use a theoretical baseline for comparison.

### *Experimental Setting*

The urban environment used was the same as for the single-area experiments except that more areas are defined. Several different combinations of the areas were considered. All the areas are outlined in Figure 8 with their dimensions given in Table 1. All configurations consisted of exactly four areas. The UAVs were flying at $100m$ with constant velocity of 25 m/s, had minimal turning radius $R = 20m$ and ground radius of sensors $\rho = 76.5m$.

| nr. | I | II | III | IV | nr. | I | II | III | IV |
|---|---|---|---|---|---|---|---|---|---|
| **1** | 200 | 200 | 200 | 200 | **2** | 400 | 200 | 200 | 200 |
| **3** | 400 | 100 | 100 | 100 | **4** | 400 | 400 | 200 | 200 |
| **5** | 400 | 400 | 400 | 100 | **6** | 400 | 300 | 200 | 100 |

**Table 1: Lengths of the sides of the square areas used in the experiments. All areas were squares. Organization of the table is the same as the organization of graphs in Figure 9. Arabic numerals denote experiment numbers; roman numerals denote the respective quadrants from Figure 8.**

The UAVs were initially placed at the border of the whole area. Nonetheless, because each experiment ran for an extended period of time, the effect of initial conditions was eliminated.

### *Results*

[6]The estimate is no longer a lower bound in general; it can be shown, however, that it *is* a lower bound for vast majority of practical cases, except for degenerate cases where the total footprint of all UAV sensors is close to the size of the area surveilled. We therefore use it for comparison.

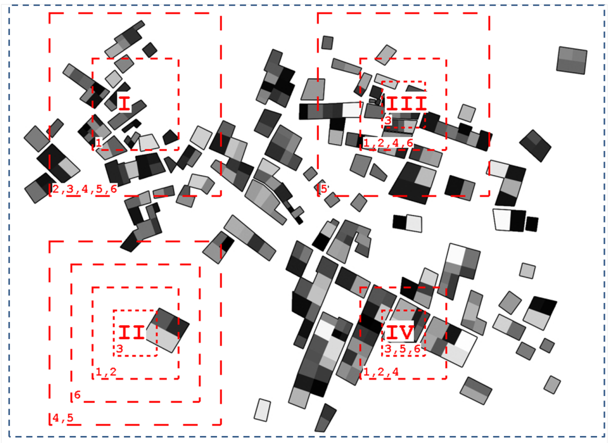


**Figure 8: The areas used in the multi-area experiments. The sizes of the numbered areas are given in Table 1.**

Before presenting the experimental results we extend the lower bound from Section 5.1 to the case of multiple areas. The extension is based on the fact that surveilling a certain spatial area is most effective if the whole area is made up by a single rectangle rather than by multiple rectangles. To obtain the estimate, we therefore substitute the term $w \cdot h$ in (3) representing the area of a single area with the sum of the areas of all the surveilled areas. The resulting estimate

$$\underline{\mathrm{I}}(\mathcal{A}, n) = \frac{1}{nv}\left(\frac{\sum_{A_i \in \mathcal{A}} A_i}{2\rho} - \frac{\rho}{\pi}\right) \tag{6}$$

is displayed alongside the empirically measured values in Figure 9. Even though there is a notable difference in the absolute values, the trends are identical. On average, when there was the same number of UAVs and areas (4 in our case) the approach performed worse by 100%. However, for twice as many UAVs as the areas (8 UAVs) it performed worse only by 30%. For three times as many UAVs (12) it performed worse by 27%.

## 6.3 Discussion

On rectangular areas with uniform distribution of points of interest, the zig-zag algorithm (significantly) outperforms the other two single-area algorithms for most combinations of UVA's altitude and turn radius. In addition, the regular shape of the generated trajectories makes the zig-zag algorithm more predictable and thus easier to work with; the zig-zag trajectories can also be easily split into multiple disjoint segments. Both properties are particularly useful when the trajectory is to be divided between multiple UAVs.

For high flight altitudes, nevertheless, the performance of the zig-zag algorithm is matched by the alternating algorithm. This is because with the increasing sensor altitude, occlusions become less critical, resulting in a decreased number of vantage points and larger distances between them. This in turn renders UAV's motion constraints relatively less critical and optimum flight paths tend to be very close to the solutions of Euclidean TSP, which is the basis of the alternating algorithm. In fact, the alternating algorithm might outperform the zig-zag algorithm for non-convex target areas or for areas with strongly non-uniform distribution of points of interest – although these cases might be handled by first segmenting each such area into a number of subareas and then employing the multi-area algorithm.

The results for multi-UAV single-area surveillance show that the simple division scheme performs surprisingly well and its performance approaches optimum. The performance for multiple areas depends on the ratio between the number of available UAVs and the total number of target areas. With the increasing ratio, the relative performance with respect to a baseline estimate improves. The requirement to have at least as many UAVs as the areas can be restrictive. A simple solution would be to alter the allocation algorithm so that the absence of UAVs in an area is not penalized by an infinite penalty but only by a finite one corresponding to the current information age of the area, and then run the modified algorithm periodically.

# 7. RELATED WORK

The problem of multi-UAV surveillance has received some attention lately and a variety of approaches from reactive policies to deliberative search-based methods have been proposed. However, none of the reported approaches explicitly deals with occlusions.

In [9] the authors present an approach for constructing a semi-heuristic control policy for multiple UAVs performing a surveillance task. [1] proposes a class of semi-distributed stochastic navigation algorithms based on the minimization of artificial potentials.

The more deliberative approaches view the surveillance problem as a *routing problem* as e.g. in [11] where the resulting problem is the *traveling salesman problem with time windows*. In some routing problems, the aircraft trajectory constraints have been explicitly considered adopting the Dubins vehicle aircraft model. Important work on routing problems with Dubins vehicles is [13], which presents several algorithms for single- and multi-UAV routing problems including the traveling repairman problem, which can be viewed as a special case of multi-UAV area surveillance problem. Again, occlusions are not considered, though.

# 8. CONCLUSIONS

We have formulated the problem of multi-UAV surveillance of multiple areas with sensor occlusions as the minimization of the average information age of a set of points of interest. Noting that the problem is intractable in its optimum formulation, we have proposed an approximate two-stage approach which first allocates the UAVs to the areas and then solves the resulting single-area surveillance problems for each of the areas with the allocated UAVs. We have proved the optimality of the allocation algorithm, assuming that there are more UAVs than areas. For the single-area surveillance, we have presented three occlusion-aware algorithms with different performance characteristics.

We have evaluated all methods empirically on a realistic UAV simulation testbed which accurately models UAVs and their on-board sensors. For the most typical rectangular areas, the zig-zag algorithm performs the best for most UAV operational conditions. For multi-UAV single-area and for multi-UAV multi-area problems, we have derived theoretical performance estimates and shown that our two-stage approach performs favorably with respect to these estimates while guaranteeing 100% coverage of the points of interest.

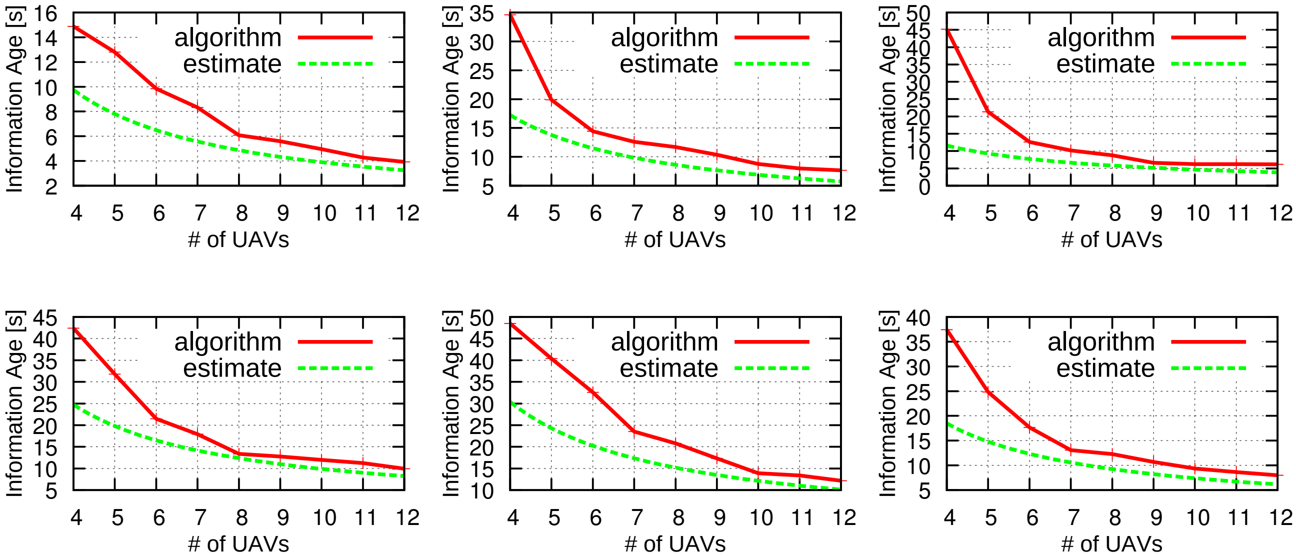


**Figure 9: Performance of the multi-area surveillance, both measured and estimated. Configurations of the respective experiments is described in Figure 8 and Table 1. (Configurations 1-3 in the upper row; configurations 4-6 in the lower row).**

## Acknowledgements

The presented research has been funded by the U.S. Army (grant no. W911NF-08-1-0521_1312AM01) and by the Czech Ministry of Education, Youth and Sports under grant "Decision Making and Control for Manufacturing III" (grant no. MSM 6840770038).

## 9. REFERENCES

[1] L. Caffarelli, V. Crespi, G. Cybenko, I. Gamba, and D. Rus. Stochastic Distributed Algorithms for Target Surveillance. *Intelligent Systems Design and Applications*, page 137, 2003.

[2] T. Cox, C. Nagy, M. Skoog, and I. Somers. Civil UAV capability assessment. Technical report, NASA Dryden Research Center, Edwards, CA, 2004.

[3] M. De Berg, D. Halperin, M. Overmars, and M. Van Kreveld. Sparse arrangements and the number of views of polyhedral scenes. *International Journal of Computational Geometry and Applications*, 7:175–196, 1997.

[4] L. Dubins. On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. *American Journal of Mathematics*, pages 497–516, 1957.

[5] M. Jakob, E. Semsch, D. Pavlíček, V. Eliáš, and M. Pěchouček. Intelligent Software Agent Control of Combined UAV Operations for Tactical Missions – Month 12 Report. Technical report, Czech Technical University, Prague, December 2009.

[6] J. Kim and Y. Kim. Moving ground target tracking in dense obstacle areas using UAVs.

[7] S. LaValle. *Planning algorithms*. Cambridge University Press, 2006.

[8] M. Marengoni, B. Draper, A. Hanson, and R. Sitaraman. A system to place observers on a polyhedral terrain in polynomial time. *Image and Vision Computing*, 18(10):773–780, 2000.

[9] N. Nigam and I. Kroo. Persistent Surveillance Using Multiple Unmanned Air Vehicles. In *2008 IEEE Aerospace Conference*, pages 1–14, 2008.

[10] M. Pechoucek and D. Sislak. Agent-based approach to free-flight planning, control, and simulation. *IEEE Intelligent Systems*, 24(1):14–17, Jan./Feb. 2009.

[11] J. Ryan, T. Bailey, J. Moore, and W. Carlton. Reactive tabu search in unmanned aerial reconnaissance simulations. In *Proceedings of the 30th conference on Winter simulation*, pages 873–880. IEEE Computer Society Press Los Alamitos, CA, USA, 1998.

[12] A. Sarmiento, R. Murrieta-Cid, and S. Hutchinson. A multi-robot strategy for rapidly searching a polygonal environment. *Lecture notes in computer science*, pages 484–493, 2004.

[13] K. Savla. *Multi UAV Systems with Motion and Communication Constraints*. PhD thesis, UNIVERSITY of CALIFORNIA, 2007.

[14] E. Semsch, M. Jakob, D. Pavlíček, and M. Pěchouček. Autonomous UAV Surveillance in Complex Urban Environments. In *Proceedings of 2009 IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT 2009)*, 2009.

[15] D. Sislak, P. Volf, and M. Pechoucek. Accelerated A* Path Planning (Extended Abstract). In *AAMAS 2009: Proceedings of the eighth international conference on Autonomous agents and multiagent systems*, 2009.

[16] H. Vollmer. Computational complexity of constraint satisfaction. In S. B. Cooper, B. Löwe, and A. Sorbi, editors, *CiE*, volume 4497 of *Lecture Notes in Computer Science*, pages 748–757. Springer, 2007.

# A Game-theoretic Approach to Leasing Agreements can Reduce Congestion

Reshef Meir
School of Computer Science and Engineering
The Hebrew University of Jerusalem, Israel
reshef24@cs.huji.ac.il

Jeffrey S. Rosenschein
School of Computer Science and Engineering
The Hebrew University of Jerusalem, Israel
jeff@cs.huji.ac.il

## ABSTRACT

Leased cars with pre-paid fuel are a significant part of traffic today in many countries. Incentivized to drive as much as possible, their users contribute to pollution, congestion, and other negative societal effects. Calls for change of these leasing arrangements, by environmental organizations and others, are often rejected due to alleged economic rationales. We analyze from a game-theoretic perspective an alternative leasing model, where each driver pays for her own fuel. We show the emergence of a unique equilibrium in which everybody gains: the drivers, their employers who are paying for fuel, and, of course, the environment.

## 1. INTRODUCTION

There are over 28 million private cars today in the UK alone, driving over 400 *billion* km each year.[1] Approximately one fifth of this vast amount of traffic is attributed to daily commuters, many of them driving leased cars owned by the company that employs them (60% of the cars in the UK are company-owned). The rapidly increasing number of cars on the roads overloads existing infrastructure, and causes a set of environmental and economic problems. These include air, ground, and river pollution, an increase in accidents, the depletion of global oil reserves, and long-term atmospheric impact [8]. In the US, the time overhead due to rush-hour congestion is estimated at 1.2 minutes per kilometer, which adds up to billions of dollars of direct economic losses [2, 24]. In addition, congestion is responsible for increased fuel consumption, and thus further aggravates environmental effects.

Given the large scale of the problem, any change in local or global policies that could result in less traffic should be welcome. Yet many companies provide a strong incentive to their employees to drive *more*, in the form of a leased car with pre-paid fuel.[2]

We compare two leasing policies and how they affect employees (i.e., drivers), their companies, and the environment. Under the Common Policy (CP), the company pays for the employees' fuel, whereas in our suggested Alternative Policy (AP), fuel is not included in the leasing agreement.

While it seems intuitive that the Alternative Policy would be in the best interests of the environment, employees often prefer to get pre-paid fuel, which they see as consistent with their own interests. Our main claim against such an attitude on the part of employees is not that it is selfish, but that it is *wrong*. That is, self-interested employees should *prefer* a policy where the fuel is not pre-paid, as should their companies. What we intend to show is that the Alternative Policy induces a "win-win-win" situation: there are fewer cars on the road (thus the environment benefits); companies spend less money; and employees are better off. Moreover, this utopia not only exists—it is also an equilibrium state, meaning that the parties have no incentive to diverge from their behavior.

In the interests of the company and its employees we only consider monetary payments and car usage. That is, the environment is not a player in the game, and we do *not* take into account the effects of driving on the environment in the interests of employees/companies. We note however, that adding such considerations would only strengthen our conclusion that the Alternative Policy should be preferred.

The intuition behind our argument is as follows. Under CP, since no additional cost is incurred by driving additional kilometers, employees will use their car at every opportunity, even when there is an alternative. Possible alternatives include cheap substitutes such as public transportation, carpooling, or simply canceling unnecessary trips, but may also be more expensive, for example, using a taxi or a plane. While we do not explicitly study alternative means of transport, we make the plausible assumption that every car ride has some measurable money-equivalent *utility* to the driver.[3] Some rides are more urgent, important, or harder to replace than others, and thus the utility of a 100km monthly quota is the cost of replacing *the most expensive* 100km with the cheapest available alternative. Naturally, the utility of driving 200km a month is higher than that of 100km, but not necessarily double, since the quota now includes less expensive rides. Similarly, we would probably be willing to pay even less in order to increase the quota by another 100km, and so on.

This assumption is known as *decreasing marginal utilities*, and is a standard assumption in economic situations. It is also clear that at some point, the marginal utility from increasing the quota becomes zero. Otherwise, drivers with pre-paid fuel (that have an unlimited quota) would drive indefinitely.[4]

In exchange for a leased car and pre-paid fuel, under the

[1]Statistics are taken from the UK Department for Transport [23].

[2]In Israel, where pre-paid fuel is the standard, leased cars are responsible for 5% of the *total annual mileage* (by combining data from [25] and the Central Bureau of Statistics [18]).

[3]We also note that cheap alternatives are in general more environment-friendly.

[4]And there is, in any case, a limited number of hours per month available for driving.

CP policy a fixed amount is deducted from an employee's salary. This amount depends on the contract between the company and its employees. In our proposed alternative policy, each employee pays for her fuel, and receives in turn a fixed salary increase (or a smaller salary deduction). We show that in each policy there is a unique equilibrium between the company and the employee, and that the equilibrium attained in the Alternative Policy is better for both sides. This occurs because the employee (i.e., the driver) stops using her vehicle for unnecessary trips, whose utility is lower than the cost of fuel.

While we make some general assumptions on the behavior of involved parties, we do not assume any specific values for the parameters of the leasing agreement (such as the price of fuel, or the distance to work). Our analysis is thus not restricted to a specific company or country.

## 1.1 Related Work

The multiagent approach has been applied to the traffic and transportation domain in two main ways. The first addresses various organizational problems by modeling the involved parties as self-interested agents, for example, to improve logistics within a freight fleet [13], to increase coordination between transportation companies [14], or to upgrade the service to clients of a public transportation system [12]. We take a similar approach in modeling the parties involved in a leasing arrangement. However, in our model we provide a formal analytic treatment of agent behavior, whereas the complexity of the aforementioned systems typically requires simulations (whose results are dependent on the specific configurations of the systems).

A different route in traffic research focuses directly on the problem of congestion. The agents in this case are typically the drivers that are making routing decisions (such as using the main road / side road) and timing decisions (such as leaving 15 minutes before their preferred time of arrival). Congestion is aggravated by the fact that if all drivers are making decisions that are *privately optimal*, then sometimes the *global* outcome is far from optimal. A famous example is the Braess paradox [9], which is a traffic-related instantiation of the problem known in economics and game theory as the *tragedy of the commons* [17]. Several attempts to alleviate this problem use some manipulation of the information that is available to the drivers [6, 4, 26], whereas others apply direct intervention to drivers' incentives via external payments/tolls. Methods are evaluated mainly by simulations [22, 5, 10], but also using a formal analysis of the equilibrium, where possible (e.g., [1]). A paper by Balbo and Pinson [3] combined the two research routes by modeling both the vehicles and the control components as agents, with the goal of regulating traffic. The proposed system (SATIR) has also been implemented and tested on data gathered in Brussels.

Crucially, all aforementioned work about congestion makes the assumption that the total amount of traffic should be considered as *fixed*, and concentrates on preventing a "bad" scenario, in which there are too many cars at the same place at the same time. This assumption was even made explicit by Tumer et al. [22], who stated that "no individual action is intrinsically bad, but that combinations of actions among agents lead to undesirable outcomes".

The grave implications of excessive traffic, described in the previous section, make us question this assumption, as we believe that driving a car can be *intrinsically worse* than using an alternative. The goal of this paper is not to disperse traffic in time and space, but rather to reduce the total amount of traffic, thus alleviating congestion, but also all other negative consequences of traffic.

Complex models that take into account changes in traffic volume (due to tolls and due to the congestion itself) have also been proposed [19]. These models still assume that drivers are very sophisticated and that they always find the exact equilibrium of the network (see [15, 7] for a critique of this assumption). In contrast, our model suggests a simple policy transition, leaving the players with an obvious optimal strategy that does not depend on the topology of the network.

## 1.2 Structure of the Paper

We first clarify some game-theoretic concepts, and formalize our intuition from the first section by considering a simplified game where only the employee acts strategically. We then add the company as a strategic player, and analyze the equilibrium in the induced game. In the remaining sections we show how our results extend to more realistic situations, where there are multiple employees with different preferences, and where employees also get the option of dropping the leasing contract. In the last section, we discuss some implications of our results, and compare them with the situation in practice.

# 2. PRELIMINARIES

### *Game.*

A *game* consists of a set of agents $N$, a set of strategies for each agent $\{A_i\}_{i\in N}$, and a utility function for each agent $U_i : \times_{j\in N} A_j \to \mathbb{R}$. The set of strategies $A_i$ does not have to be discrete. For example, the strategy may be to decide on an amount of money to spend. A joint selection of strategies for each agent $\mathbf{a} = \{a_j \in A_j\}_{j\in N}$ is called a *strategy profile*. The profile of all agents *except* $i$ is denoted by $a_{-i} = \{a_j \in A_j\}_{j\neq i}$.

### *Equilibrium.*

We say that the strategy profile $\mathbf{a}$ is an *equilibrium*, if no agent can gain by choosing a different strategy, assuming that all other agents keep theirs. Formally, $\mathbf{a}$ is an equilibrium if for any agent $i$ and any strategy $a'_i \neq a_i$, we have that $U_i(\mathbf{a}) \geq U_i(a_{-i}, a'_i)$. Our definition coincides with the standard definition of a *pure Nash equilibrium*. Since we do not allow agents to randomize between strategies, we only consider pure equilibria. Thus, it is possible that a game does not contain any equilibrium.

### *Dominant strategies.*

$a^*_i \in A_i$ is a *dominant strategy* of $i$ if agent $i$ always prefers $a^*_i$, regardless of the choices of other agents. Formally, for all $\mathbf{a}$, $U_i(a^*_i, a_{-i}) \geq U_i(\mathbf{a})$. Note that if some player has a unique dominant strategy, then all other players can assume that this strategy will be played. This simplifies the game, as the size of the strategy space is significantly reduced. In particular, it is possible that in the new, simplified game, there is an agent $j \neq i$ that has a dominant strategy (under the assumption that $i$ plays $a^*_i$). It is sometimes possible to continue to remove strategies from the game until there is only one strategy profile left. In this case we say that the game is *iterated dominance solvable*. The outcome $\mathbf{a}^*$ is called the *iterated dominant strategy equilibrium*, and it is also the unique Nash equilibrium of the game.

For a detailed discussion regarding these definitions and for more background in game theory, see, for example, [20].

# 3. INITIAL MODEL

## 3.1 The Common Policy

In the simplest case, we model the interaction between a single company $c$ and a single employee $e$. The utility of the employee (denoted by $u = U_e$) is composed of two factors: one factor is her income, which we denote as $s$. The other factor is the number of kilometers she drives in a month (mileage), denoted by $x$. While the income $s$ is not controlled by the employee, she is free to choose how much to drive; thus her strategy space is $A_e = \mathbb{R}_+$ (and $x \in A_e$).

In the common leasing policy (which we denote by CP), the utility of the employee can be decomposed as

$$u_{CP}(s, x) = s + f(x) \quad ,$$

i.e., there is some function $f$ that makes the two factors comparable. As explained in the introduction, we make the following assumptions regarding the utility of the employee:

ASSUMPTION 1. *The employee has decreasing marginal utility from driving more, and there is a maximal mileage that the employee has no reason to exceed. Formally:*

*a. $f$ is non-decreasing and continuous.*

*b. $f$ is* concave*, i.e., for all $y < z$ and $\epsilon > 0$, $f(y + \epsilon) - f(y) \geq f(z + \epsilon) - f(z)$.*

*c. There is some $x^*$ s.t. $f$ has a maximum in $f(x^*)$.*

For simplicity, we will make the technical assumption that $f$ is *strictly concave* in the range $[0, x^*]$, although this assumption is not necessary and can be relaxed. Thus, for all $0 \leq y < z \leq x^*$ we have that $f(y+\epsilon)-f(y) > f(z+\epsilon)-f(z)$.

Clearly, in CP the dominant strategy of the agent is to drive $x^*$, thus maximizing utility. This holds for any fixed income $s$, and therefore the company has no influence on the strategy of the employees regarding their mileage.

We also compute the utility of the company (denoted by $v = U_c$), although for now we will not treat the company as a player in the game (i.e., we will not consider the rationality of its actions). The fuel cost is linear in the mileage and we denote by $k$ the average cost of fuel per 1 kilometer of driving; thus $v_{CP}(s, x) = -s - x \cdot k$.

If we assume that the employee follows her dominant strategy, we get that in the Common Policy, the utility of the employee is $u_{CP}(s, x^*) = s + f(x^*)$, while for the company $v_{CP}(s, x^*) = -s - x^* \cdot k$.

## 3.2 The Alternative Policy

Now suppose that in addition to the fixed income, our employee also has to pay for consumed fuel. We define a new game for the alternative leasing policy (AP), with the same strategies but different utility functions. Fuel cost is linear in the mileage, thus the utility of the employee in AP is:

$$u_{AP}(s, x) = s + f(x) - k \cdot x \quad .$$

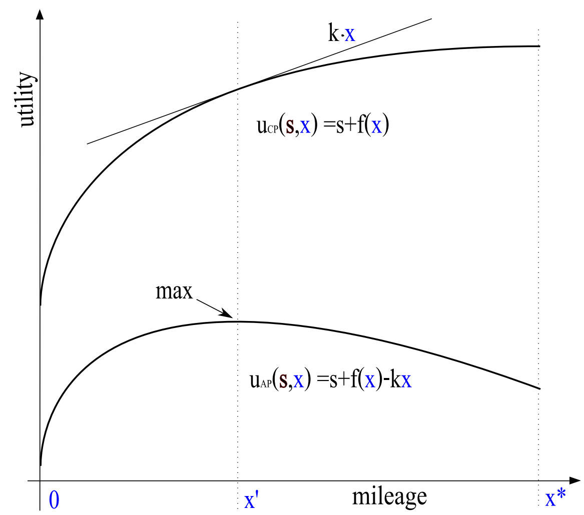


**Figure 1: Utility of the employee as a function of the mileage in the common leasing policy (top) and in the alternative policy (bottom). $x^*$ and $x'$ are the employee's dominant strategies in both policies, respectively.**

The relation between the common and alternative policies w.r.t. the utility of the employee is demonstrated in Figure 1. The best strategy for the employee in AP depends on both $k$ and $f$, and we make the following observations:

- $f(x) - k \cdot x$ is still concave.
- $f$ has a peak in some $x' < x^*$ (see Figure 1).
- Regardless of $s$, the dominant strategy of the employee in AP is to drive $x'$.
- The utility for the dominant strategy is $u'_{AP}(s) = s + f(x') - k \cdot x'$.

It is also clear, that if the income of the employee remains the same, then paying for fuel will only decrease her utility, i.e., $u_{AP}(s, x) < u_{CP}(s, x)$. However, we should keep in mind that the company saves money by not paying for the employee's fuel.

The utility of the company becomes even simpler in the Alternative Policy, as the only factor that has an effect is the salary itself, i.e., $v_{AP}(s, x) = -s$. If we assume that the employee follows her dominant strategy, we have that in AP, the company may increase the salary by $\Delta$ and still gain (compared to CP), as long as $v_{AP}(s + \Delta, x') > v_{CP}(s, x^*)$. This supplies us with a simple formalization of the intuition given earlier.

PROPOSITION 2. *There is a strategy for the company such that both company and employee gain by switching to the Alternative Policy. Formally, there exists $\Delta > 0$ such that*

*1. $u_{AP}(s + \Delta, x') > u_{CP}(s, x^*)$, and*

*2. $v_{AP}(s + \Delta, x') > v_{CP}(s, x^*)$.*

PROOF. The constraint on $v$ is satisfied as long as $\Delta < k \cdot x^*$, since $v_{AP}(s + \Delta, x') - v_{CP}(s, x^*) = s + k \cdot x^* - s - \Delta$.

Also, as long as $\Delta > f(x^*) - f(x') + kx'$, the constraint on $u$ is satisfied, as

$$\begin{aligned} u_{AP}(s+\Delta, x') - u_{CP}(s, x^*) \\ &= s + \Delta + f(x') - k \cdot x' - (s + f(x^*)) \\ &= \Delta - (f(x^*) - f(x') + kx') > 0 \quad . \end{aligned}$$

It is thus left to show that both constraints can be satisfied at the same time. Recall that $x'$ was defined such that $f(x') - kx' = max_{x \geq 0}(f(x) - kx)$. In particular,

$$\begin{aligned} f(x') - kx' > f(x^*) - kx^* \quad &\Rightarrow \\ kx^* > f(x^*) - f(x') + kx' \quad &\Rightarrow \end{aligned}$$

$$\exists \Delta, \quad kx^* > \Delta > f(x^*) - f(x') + kx' \quad . \tag{1}$$

□

### 3.3 Weaknesses of the Initial model

Our basic model provides some formal flavor for the intuition that win-win-win situations can be achieved simply by transferring the fuel cost from the company to the employee, with very few additional assumptions. Unfortunately, such a simple analysis still suffers from several weaknesses.

First, although the employee had a dominant strategy in both policies, we did not analyze the actions available to the company from a strategic point of view. Therefore it is possible in principle that the new state is not an equilibrium.

Second, we only modeled a single employee. Although the results hold if we add *identical* employees, we have a problem when employees have different utility functions. For example, if one employee lives closer to the train station, or happens to like bicycling, then the utility of 100km for her might be lower than the utility for her colleague.

Third, we did not consider other actions available to the employee, such as dropping the leasing deal altogether.

In the following sections, we address these issues by refining our model and adding more assumptions where needed.

## 4. LEASING AS A TWO-PLAYER GAME

In this section we will extend our initial model by formally defining the different factors affecting the company's utility from the leasing interaction, and analyze the stability of the possible outcomes w.r.t. both the company and the employee.

We first note that our original definition of the company's utility, which only considered expenses, ignored an important factor. The company *gains* something from the employee, otherwise she would not be employed in the first place. As with $f$, the "gain" function may take different forms, but we can make similar plausible assumptions about it. Formally, we denote the gain function by $g(u)$, where $u$ is the utility of the employee.

ASSUMPTION 3. *The company has a decreasing marginal profit from the utility of the employee. Formally:*

*a. $g$ is non-decreasing and continuous.*

*b. $g$ is strictly concave.*[5]

[5]As with $f$, we make this assumption to simplify the analysis, and in practice a weaker restriction on $g$ would suffice.

There are several justifications for the monotonicity assumption. This can be interpreted as "happy employees work harder", but also as "better conditions attract better employees". We do not search for the "correct" interpretation, as the implication is the same. The decreasing marginal profit is a standard economic assumption.

We can now rewrite the utility of the company, considering the productivity of the employee for both policies:

$$\hat{v}_{CP}(s,x) = g(u_{CP}(s,x)) + v_{CP}(s,x) = g(s+f(x)) - s - kx \quad ,$$

and

$$\hat{v}_{AP}(s,x) = g(u_{AP}(s,x)) + v_{AP}(s,x) = g(s+f(x)-kx) - s \quad . \tag{2}$$

Recall that in either policy, the employee has a dominant strategy (either $x^*$ or $x'$). Assuming that the employee is indeed using her dominant strategy, we expect the company to optimize the salary, so as to maximize its profit. Thus there is an optimal salary $s^*$ that maximizes $v_{CP}(s, x^*)$, and the strategy profile $(x^*, s^*)$ is the iterated dominant strategy equilibrium of CP.

As for the Alternative Policy, recall that in order to make both sides benefit, the company needs to increase the salary of the employee by $\Delta$, under the constraints described earlier (Equation (1)). Let $\hat{\Delta}$ be an arbitrary amount that satisfies the constraints. Note that

$$\begin{aligned} & u_{AP}(s^* + \hat{\Delta}, x') > u_{CP}(s^*, x^*) && \Rightarrow \\ & && \text{(from Prop. 2)} \\ & g(u_{AP}(s^* + \hat{\Delta}, x')) > g(u_{CP}(s^*, x^*)) && \Rightarrow \\ & && (g \text{ is monotone}) \\ & \hat{v}_{AP}(s^* + \hat{\Delta}, x') = g(u_{AP}(s^* + \hat{\Delta}, x')) + \quad v_{AP}(s^* + \Delta, x') \\ & > g(u_{CP}(s^*, x^*)) + v_{AP}(s^* + \Delta, x') \\ & > g(u_{CP}(s^*, x^*)) + v_{CP}(s^*, x^*) && \text{(from Prop. 2)} \\ & = \hat{v}_{CP}(s^*, x^*) && \Rightarrow \end{aligned}$$

$$\hat{v}_{AP}(s^* + \hat{\Delta}, x') > \hat{v}_{CP}(s^*, x^*) \quad , \tag{3}$$

so the company still gains according to $\hat{v}$. However, this alone does not guarantee stability. Hypothetically, it is possible that in AP there is no equilibrium, or that there is an equilibrium that is worse for either the company or the employee. We will now see that this is not the case.

We denote by $s' = s^* + \Delta'$ the best response of the company to the dominant strategy of the employee (i.e., to $x'$) in AP, thus $(s', x')$ is the (unique) iterated dominant strategy equilibrium of AP. We intend to show that the equilibrium in the Alternative Policy (i.e., $(s', x')$) is preferred by *both* players over the equilibrium $(s^*, x^*)$ in the Common Policy. To this end, we first prove a (positive) lower bound on the salary increase that the company must give in the Alternative Policy.

Denote by $\Delta_{inf}$ the infimum of $\Delta$ that obeys the constraints imposed by (1), i.e., $\Delta_{inf} = f(x^*) - f(x') + kx'$.

LEMMA 4. *If $\Delta < \Delta_{inf}$ then*

$$\hat{v}_{AP}(s^* + \Delta, x') < \hat{v}_{AP}(s^* + \Delta_{inf}, x') \quad .$$

PROOF. Assume that $g$ is continuously differentiable, thus $v_{CP}$ is also continuously differentiable in $s$. Also, $\hat{v}_{CP}(s, x^*) = g(s + f(x^*)) - s - kx^*$ (as a function of $s$) has a maximum in $s^*$, i.e., its derivative in $s = s^*$ is 0, and is strictly positive

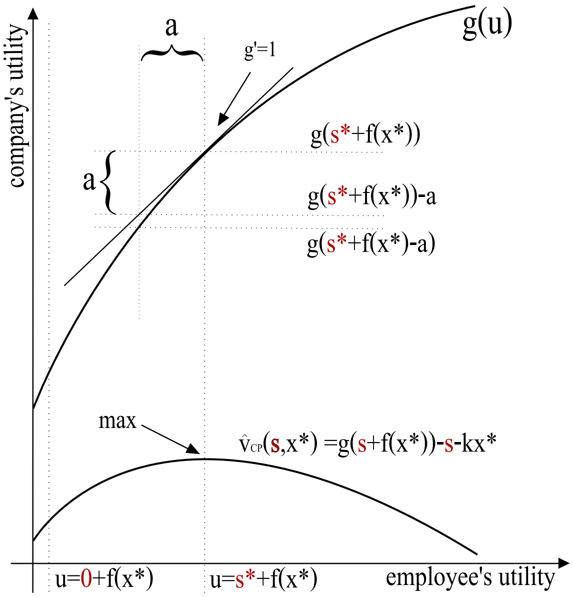


**Figure 2: The gain function (top) and the utility of the company in the Common Policy (bottom), as a function of the salary.**

in any $s < s^*$ (from concavity). By differentiating $g$ we have that

$$\frac{\partial \hat{v}_{CP}(s, x^*)}{\partial s} = \frac{\partial g(s + f(x^*)) - s - kx^*}{\partial s} = \frac{\partial g(s + f(x^*))}{\partial s} - 1 \ ,$$

thus the slope of $g$ in $u = s + f(x^*)$ is *higher than* 1 in any $s < s^*$. Consider the interval $[u_1, u_2]$, where $u_1 < u_2 \leq s^* + f(x^*)$. The slope of the straight line connecting the values of $g$ in both edges of the interval (i.e., $\frac{g(u_2)-g(u_1)}{u_2-u_1}$) cannot be lower than 1, as 1 is a lower bound of the derivative of $g$ in the interval (see Figure 2), thus for any $a > 0$ we get that

$$g(s^*+f(x^*)) - g(s^*+f(x^*)-a) > s^*+f(x^*) - (s^*+f(x^*)-a) = a \ . \tag{4}$$

We now add that Equation (4) holds even if $g$ is *not* continuously differentiable, since its slope is still lower bounded by 1 (although it requires some technical work to show that).

We now take $a > 0$ to be the difference $\Delta_{inf} - \Delta$.

$$\begin{aligned}
&\hat{v}_{AP}(s^* + \Delta, x') - \hat{v}_{AP}(s^* + \Delta_{inf}, x') \\
&\quad = [g(s^* + \Delta_{inf} - a + f(x') - kx') - (s^* + \Delta_{inf} - a)] \\
&\qquad - [s^* + \Delta_{inf} + f(x') - kx') - (s^* + \Delta_{inf})] && \text{(from (2))} \\
&\quad = g(s^* + f(x^*) - f(x') + kx' - a + f(x') - kx') + a \\
&\qquad - [g(s^* + f(x^*) - f(x') + kx' + f(x') - kx') \\
&\quad = a + g(s^* + f(x^*) - a) - g(s^* + f(x^*)) \\
&\quad < a - a = 0 && \text{(from (4))}
\end{aligned}$$

□

In other words, the company is unwilling to reduce the salary of the employee (or even to increase it by less than $\Delta_{inf}$), since otherwise it would have also been profitable to pay the employee less in the first place.

As an immediate corollary from Lemma 4 we get that for the best response $s'$, $\Delta'$ must be at least $\Delta_{inf} = f(x^*) - f(x') + kx'$, which means (as seen in the proof of Proposition 2) that

$$u_{AP}(s', x') = u_{AP}(s^* + \Delta', x') \geq u_{CP}(s^*, x^*) \ .$$

That is, the employee is indeed not harmed in the new equilibrium.

As for the company, we have that

$$\hat{v}_{AP}(s', x') \geq \hat{v}_{AP}(s^* + \Delta', x') > \hat{v}_{CP}(s^*, x^*) \ ,$$

where the first inequality is due to the fact that $s'$ is the best response to $x'$, and the second is due to (3). Thus the company is even better off with the new equilibrium. We restate this result in the following proposition.

Proposition 5. *The equilibrium profile $(s', x')$ in the Alternative Policy is preferred by both players to the equilibrium $(s^*, x^*)$ in the Common Policy. Formally,*

1. *$u_{AP}(s', x') \geq u_{CP}(s^*, x^*)$, and*
2. *$\hat{v}_{AP}(s', x') > \hat{v}_{CP}(s^*, x^*)$.*

## 5. MULTIPLE EMPLOYEES

We now turn to an extension to multiple employees. As noted, adding *identical* employees makes no difference, as the equilibrium described in previous sections will satisfy all of them independently. Unfortunately (at least from an analytic point of view) different people do have different preferences, which are reflected in our model as different functions $f_i$ for each employee. Proposition 2 cannot be extended to this case, as the following example shows:

Example 6. *Suppose $k = 1$. For the first employee, $x_1^* = 10$; $f_1(x_1^*) = 10$; $x_1' = 5$; $f_1(x_1') = 8$. For the second employee $x_2^* = 20$; $f_2(x_2^*) = 20$; $x_2' = 12$; $f_2(x_2') = 15$.*

*Since the fuel cost of the first employee in the Common Policy was $kx_1^* = 10$, the company must limit the salary increase (when switching to the Alternative Policy) to* at most *10, or otherwise the leasing arrangement of employee 1 will become less profitable.*

*On the other hand, after the switching the utility of employee 2 decreases by $f_2(x_2^*) - f_2(x_2') + kx_2' = 17$, thus employee 2 is worse off in the Alternative Policy unless the salary increase is* at least *17.*

*Therefore, any fixed salary increase $\Delta$ cannot improve the situation of all 3 players: it will either disappoint employee 2, or will make the leasing deal of employee 1 less desirable for the company (or both).*

In the general formulation of the multi-employee problem, there are $n$ employees. Together with the company, the game now has $n+1$ players. The strategy space of each employee is her mileage, as in Section 4. As for the company, it is possible in theory to give a different salary increase $\Delta_i$ to every employee. This would break down the game to $n$ independent games that can be solved as in Section 4. However, this would be unfair and impractical, since this increase would be based on personal habits—some employees will get more just because they like to drive more. Moreover, employees might behave strategically by increasing their mileage before the new policy takes effect, thus manipulating the value of $x_i^*$ and $\Delta_i$.

The salary increase may also be based on the distance from the employee's home to her workplace. However, this idea only supplies us with a partial solution, since the prepaid fuel is also used for private needs, and also because some alternative commuting solutions may be unavailable or inconvenient for some of the employees.

We therefore assume that the strategy of the company has a single parameter, $\Delta$, which is the salary increase given to all employees in the Alternative Policy. Thus the utility function of the employees remains the same:

$$u_{i\ AP}(\Delta, x_i) = f_i(x_i) + s_i^* + \Delta - kx_i \quad ,$$

where $s_i^*$ is the salary of the employee in the Common Policy (assumed to be fixed). The utility of the company from each interaction is $\hat{v}_{i\ AP}(\Delta, x_i) = g(u_{i\ AP}(\Delta, x_i)) - s_i^* - \Delta$, and its total utility in the game is

$$\mathbf{v}_{AP}(\Delta, x_1, \ldots, x_n) = \sum_{i=1}^{n} \hat{v}_{i\ AP}(\Delta, x_i) \quad .$$

The dominant strategy of each employee does not depend on $\Delta$, nor on the mileage of the other employees. Thus, we can continue to assume that employee $i$ drives $x_i'$ kilometers in the Alternative Policy.

We take the simple approach of computing the *average* value $\bar{\Delta} = \frac{1}{n}\sum \Delta_i$, where $\Delta_i$ is determined according to the two-player game between the company and employee $i$, as in Section 4. We compute the social welfare in the new policy (the company's utility is computed separately and is *not considered* part of the social welfare). We find that

$$\begin{aligned}\sum_{i=1}^{n} u_{i\ AP}(s_i^* + \bar{\Delta}, x_i') &= \sum_i \left(s_i^* + \bar{\Delta} + f_i(x_i') - kx_i'\right)\\ &= n\bar{\Delta} + \sum_i \left(s_i^* + f_i(x_i') - kx_i'\right)\\ &= \sum_i \Delta_i + \sum_i \left(s_i^* + f_i(x_i') - kx_i'\right)\\ &= \sum_i \left(s_i^* + \Delta_i + f_i(x_i') - kx_i'\right)\\ &= \sum_i u_{i\ AP}(s_i^* + \Delta_i, x_i') \geq \sum_{i=1}^{n} u_{i\ CP}(s_i^*, x_i^*) \quad ,\end{aligned} \tag{5}$$

i.e., the social welfare still improves in the Alternative Policy (although some employees may be unhappy).

We now compute the utility $\mathbf{v}$ of the company, when summing over all interactions. Since $\Delta_i < kx_i^*$, we get that

$$n\bar{\Delta} < \sum_{i=1}^{n} kx_i^* \quad , \tag{6}$$

thus the company still saves money w.r.t. the Common Policy. This does not mean that the utility of the company improves, since we did not consider the gain ($g$) yet.

Unfortunately, even though the expenses of the company are lower *and* the social welfare increased (suggesting employees are happier), it is not guaranteed that the overall utility of the company increases. This is due to the non-linearity of the gain function $g$. It suggests that there may potentially be an embittered employee whose productivity now deteriorates significantly, dragging down the average gain. A closer look at this scenario reveals that not every employee can have such a negative effect. The happier employees are (before the change), the smaller their effect on the change in the average gain (due to the concavity of $g$). If indeed the Alternative Policy is more profitable to those who are initially worse off, then the increase in social welfare will induce an increase in the average gain—and hence in the utility of the company. Moreover, it is quite reasonable to assume that in reality, the employees who benefit the most from the common leasing policy are indeed those who exploit it the most by accumulating very high mileage. Thus these employees will indeed benefit less than others from the new policy, as in the case in Example 6. We now formalize and prove this intuition.

ASSUMPTION 7. *The happier an employee is in the Common Policy CP, the smaller her benefit from the Alternative Policy AP (it may be negative). Formally, if*

$$u_{i\ CP}(s_i^*, x_i^*) > u_{j\ CP}(s_j^*, x_j^*) \quad ,$$

*then*

$$u_{i\ AP}(s_i^* + \bar{\Delta}, x_i^*) - u_{i\ CP}(s_i^*, x_i^*) < u_{j\ AP}(s_j^* + \bar{\Delta}, x_j^*) - u_{j\ CP}(s_j^*, x_j^*) \quad .$$

ASSUMPTION 8. *The gain of the company $g$ is uniform and does not depend on the identity of the employee.*

The justification of Assumption 8 is as follows. Unlike $f_i$, which reflects the private enjoyment of each employee from using her car, the gain function $g$ depends more on the specific job requirements. Although it is unlikely that one gain function will fit *all* employees, it is still reasonable to make this assumption for a group of employees in a similar position. Thus the leasing agreement can be retuned for each such group separately.

PROPOSITION 9. *Both social welfare and the utility of the company increase in the Alternative Policy. Formally,*

$$\sum_i u_{i\ AP}(s_i^* + \bar{\Delta}, x_i') \geq \sum_i u_{i\ CP}(s_i^*, x_i^*) \quad , \tag{7}$$

$$\mathbf{v}_{AP}(s_i^* + \bar{\Delta}, x_i') > \sum_i \hat{v}_{i\ CP}(s_i^*, x_i^*) \quad . \tag{8}$$

PROOF. We get (7) directly from Equation (5), so we only need to prove the company's side.

LEMMA 10.

$$\sum_i g(u_{i\ AP}(s_i^* + \bar{\Delta}, x_i')) > \sum_i g(u_{i\ CP}(s_i^*, x_i^*)) \quad .$$

PROOF. We denote $a_i = u_{i\ CP}(s_i^*, x_i^*)$ and $b_i = u_{i\ AP}(s_i^* + \bar{\Delta}, x_i')$. Assume w.l.o.g. that employees are sorted according to $a_i$ (increasing). From Assumption 7, this also means that the difference $b_i - a_i$ is *decreasing*. That is, the employee with the lowest index has the largest benefit from AP, then the benefit gets smaller and smaller (and possibly negative) for larger $i$.

We now define a new set of points, $b_i'$, in the following way. We take every $j$ s.t. $b_j < a_j$, and "push" it up toward $a_j$, until $b_j' = a_j$. We then compensate by pushing $b_1$ (down) toward $a_1$. If $b_1' = a_1$ already, we continue to push the next point $b_2$ and so on, until $\sum b_i' = \sum b_i$. This step is demonstrated in Figure 3. We repeat the process as long as there are points such that $b'_j < a_j$. Note that the process must end, since $\sum b'_i = \sum b_i > \sum a_i$ (from Equation 5).

From the concavity of $g$, when two points $j < i$ are pushed in opposite directions by $\epsilon$, we get that

$$g(b_j - \epsilon) + g(b_i + \epsilon) \leq g(b_j) + g(b_i) \quad ,$$

since $g$ is steeper around $j$. Thus, after all steps are performed $\sum_i g(b'_i) \leq \sum_i g(b_i)$. Also, after the final step there are no points such that $b'_i < a_i$, thus

$$\sum_i g(a_i) \leq \sum_i g(b'_i) < \sum_i g(b_i) \quad ,$$

as required. $\square$

We continue with the proof:

$$\begin{aligned}
\mathbf{v}_{AP}(\bar{\Delta}, x_1, \ldots, x_n) &= \sum_i \hat{v}_{i\ AP}(s^*_i + \bar{\Delta}, x'_i) \\
&= \sum_i \left(g(u_{i\ AP}(s^*_i + \bar{\Delta}, x'_i)) - (s^*_i + \bar{\Delta})\right) \\
&> \sum_i g(u_{i\ CP}(s^*_i, x'_i)) - \sum_i (s^*_i + \bar{\Delta}) \quad \text{(from lemma 10)} \\
&= \sum_i g(u_{i\ CP}(s^*_i, x'_i)) - \sum_i s^*_i - n\bar{\Delta} \\
&> \sum_i g(u_{i\ CP}(s^*_i, x'_i)) - \sum_i s^*_i - \sum_i kx^*_i \qquad \text{(from (6))} \\
&= \sum_i g(u_{i\ CP}(s^*_i, x'_i)) - s^*_i - kx^*_i = \sum_i \hat{v}_{i\ CP}(s^*, x^*) \quad .
\end{aligned}$$

$\square$

# 6. DROPPING THE CONTRACT

So far we assumed that the strategy of the employee is limited to the mileage she drives in her car. However, an employee who does not believe that her leasing deal is profitable, will simply drop the contract. By doing so, our employee will typically start using her private car.[6]

To incorporate this type of behavior in our model, we will formulate the utility of the agent when not engaged in a leasing deal at all:

$$u_0(x) = s_0 + f(x) - k_0 \cdot x \quad .$$

The base salary is of course higher, since no money is deducted from it, thus $s_0 > s^*$. However, the employee now needs to pay even more per kilometer, as there are other expenses on top of fuel, thus $k_0 > k$.

It is easy to see that the maximum of $u_0$ is reached for some $x'' < x'$, due to the increased cost per kilometer. Now, if we keep our analysis restricted to an interaction with a single employee, there is no problem. We know that if the employee used a leased car in the first place (i.e., in the Common Policy), then $u_{CP}(s^*, x^*) > u_0(s_0, x'')$. As we already showed in Section 4, in the Alternative Policy the employee only ends up happier, and there is no reason for her to drop the contract after the policy has been switched.

Of course, in a typical company with many employees, some of them might become disappointed with the new deal (as we saw in Section 5) and renounce it altogether. At least from the environmental point-of-view, this is not at all bad, as the mileage will decrease even more to $x''$. Moreover, if an employee decides to drop the leasing deal, it is because this decision is better for him, which means social welfare increases even more. Also, leasing deals are typically subsidized by companies; the company is almost never damaged when an employee returns her car. To formalize these statements, we will add some notation. $c$ is the fixed cost of each leasing deal to the company (not including fuel). $v_0(x)$ represents the basic utility of the company when the employee does not use a leased car:

$$v_0(x) = g(u_0(x)) - s_0 + c \quad .$$

The $+c$ represents the fixed leasing cost which is saved when the employee does not lease a car. As leasing deals are subsidized (on average), we assume that

$$c \geq s_0 - (s^* + \bar{\Delta}) \quad . \tag{9}$$

We now incorporate the new strategy (of the employee) $z \in \{TAKE, DROP\}$ into the utility functions:

$$\bar{u}_{AP}(s, x, z) = \begin{cases} u_0(x) & \text{,if } z = DROP \\ u_{AP}(s, x) & \text{,if } z = TAKE \end{cases} \quad , \tag{10}$$

and

$$\bar{v}_{AP}(s, x, z) = \begin{cases} v_0(x) & \text{,if } z = DROP \\ v_{AP}(s, x) & \text{,if } z = TAKE \end{cases} \quad . \tag{11}$$

Clearly, there are only two possible outcomes: either the employee takes the deal, in which case she plays $x'$ and the company plays $s^* + \bar{\Delta}$ (as described in previous sections). Otherwise, the employee drops the deal and plays $x''$, while the company pays the base salary $s_0$. We assume that the employee chooses the better possibility for her from these two options. We denote by $(BEST_i)$ the strategy vector preferred by the employee $i$ (i.e., $(BEST_i)$ is either $(s^*_i + \bar{\Delta}, x'_i, TAKE)$ or $(s_{i0}, x''_i, DROP)$, whichever maximizes $\bar{u}_{i\ AP}$).

PROPOSITION 11. *Even if employees can choose to drop the leasing deal, the Alternative Policy still increases both social welfare and the company's utility (under all the assumptions described so far). Formally:*

$$\sum_i \bar{u}_{i\ AP}(BEST_i) \geq \sum_i u_{i\ CP}(s^*_i, x^*_i) \quad ,$$

*and*

$$\sum_i \bar{v}_{i\ AP}(BEST_i) > \sum_i \hat{v}_{i\ CP}(s^*_i, x^*_i) \quad .$$

PROOF. We begin with the social welfare. Since for every employee

$$\bar{u}_{i\ AP}(BEST_i) \geq \bar{u}_{i\ AP}(s^*_i + \bar{\Delta}, x'_i, TAKE) = u_{i\ AP}(s^*_i + \bar{\Delta}, x'_i) \ ,$$

then from Equation (7) of Proposition 9,

$$\sum_i \bar{u}_{i\ AP}(BEST_i) \geq \sum_i u_{i\ CP}(s^*_i, x^*_i) \quad .$$

We also get from monotonicity of $g$ that

$$g(\bar{u}_{i\ AP}(BEST_i)) \geq g(u_{i\ AP}(s^*_i + \bar{\Delta}, x'_i)) \quad . \tag{12}$$

This means that the company is never damaged even if the employee returns the car, since in this case $(BEST_i) =$

[6]It is quite unlikely that a leasing deal was profitable in the first place for a person who can manage without a car at all.

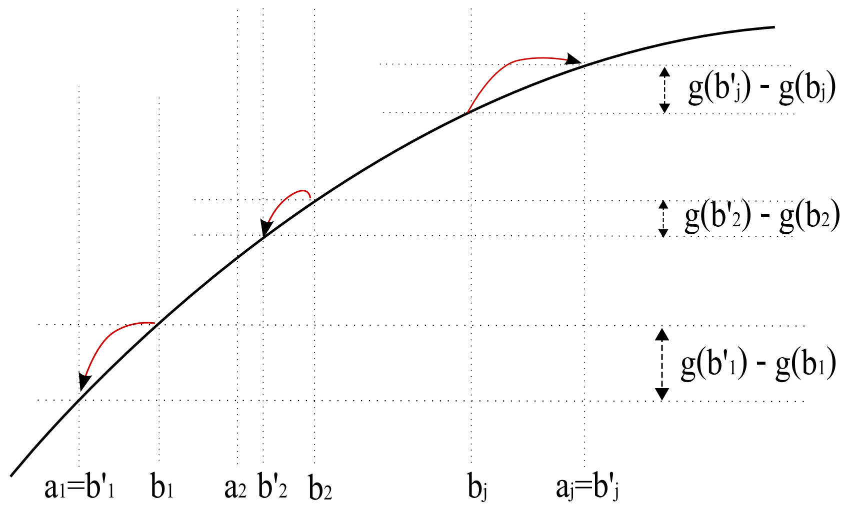


**Figure 3: The relation between the different sets of points before and after translation. The points $b_1$ and $b_2$ were moved left to compensate for moving $b_j$ right. It is easy to see that $|b'_j - b_j| = |b'_1 - b_1| + |b'_2 - b'_2|$, but $g(b'_j) - g(b_j) < (g(b_1) - g(b'_1)) + (g(b_2) - g(b'_2))$ which means $g(b_1) + g(b_2) + g(b_j) > g(b'_1) + g(b'_2) + g(b'_j)$.**

$(s_{i0}, x''_i, DROP)$ and

$$\begin{aligned}\bar{v}_{i\ AP}(BEST_i) &= v_{i0}(x''_i) = g(u_{i0}(x''_i)) - s_{i0} + c\\ &= g(\bar{u}_{i\ AP}(BEST_i)) - s_{i0} + c\\ &\geq g(u_{i\ AP}(s^*_i + \bar{\Delta}, x'_i)) - s_{i0} + c \qquad \text{(from (12))}\\ &\geq g(u_{i\ AP}(s^*_i + \bar{\Delta}, x'_i)) - (s^* + \bar{\Delta}) \qquad \text{(from (9))}\\ &= \hat{v}_{i\ AP}(s^*_i + \bar{\Delta}, x'_i) \quad .\end{aligned}$$

If the employee keeps the car, then $\bar{v}_{i\ AP}(BEST_i) = \hat{v}_{i\ AP}(s^*_i + \bar{\Delta}, x'_i)$ by Definition (11).

Finally, when we sum up the gains, then from Equation (8) of Proposition 9,

$$\sum_i \bar{v}_{i\ AP}(BEST_i) \geq \sum_i \hat{v}_{i\ AP}(s^*_i + \bar{\Delta}, x'_i) > \sum_i v_{i\ CP}(s^*_i, x^*_i) \ ,$$

as required. □

# 7. DISCUSSION

We showed that transferring the fuel cost from the company to the employee has more benefits than "just" helping the environment and reducing congestion. It will actually leave both employees and their employer richer. This result holds under quite weak and realistic assumptions on the preferences of the involved parties. Moreover, results still hold when we consider stability issues, multiple interactions, and the option to return the car. The underlying idea that is responsible for this situation is *marginal benefit vs. marginal cost*. When an employee does not pay for fuel, her marginal cost of driving more is 0, which gives her an incentive to use her car even when the marginal benefit from it is negligible. On the other hand, using the car does not really come for free—it does have a cost, which is externalized and incurred on the company (and on the environment). The company, in turn, rolls some of this cost back on the employee, "hidden" inside the salary deduction of the leasing deal.

## 7.1 Equilibrium and Commitments

In Section 4 we showed that the Alternative Policy not only enables a situation that is better for everyone, but that this situation is also an equilibrium (in iterated dominant strategies). In Sections 5 and 6 we also suggested a strategy for the company that makes the Alternative Policy better for all the involved parties. However, this strategy (i.e., $\bar{\Delta} = \frac{1}{n}\sum_{i=1}^{n} \Delta_i$) is not necessarily the optimal strategy of the company, and so the new state does not have to be an equilibrium. This means that employees may have a justified objection to the transition, if they do not trust the company to carry out the suggested strategy.

In order to solve this issue, we will use the notion of *commitments*. A player that commits to a strategy (i.e., *limits* his own freedom) can in certain situations create an equilibrium where it does not exist, or shift an existing equilibrium in a game towards one that is preferable to him [11]. Our intent is slightly different, as the commitment is supposed to convince the other players to play a different game (i.e., AP instead of CP). A company that is interested in carrying out such a transition (to the Alternative Policy), could alleviate the suspicions of its employees by *committing* to the aforementioned strategy. That is, the company will publicly announce the intended raise $\Delta$ due to the policy shift, and a binding contract can be used to enforce such a commitment. This makes the outcome we analyzed in the last two sections an equilibrium in dominant strategies, as only the employees are free to change their strategy (and they have no incentive to do so).

## 7.2 The Real World

The immediate question that arises from our results is about their validity in the real world. If the Alternative Policy is indeed so desirable, we would expect companies in the market to have adopted it by now. We supply two possible explanations for the current situation, although these should be taken only as preliminary suggestions, as this question is not the focus of this paper (we do intend to study this question more deeply in future research).

The first reason is that local or national taxation policy makes the Common Policy (with pre-paid fuel) more profitable for companies, thus effectively subsidizing fuel that is paid for by the company. The taxation system in the UK until recently was shown to act in this manner [24].[7]

A second reason, that is perhaps harder to verify, is that the benefit of free fuel is *perceived* (in the eyes of the employee) as better than it really is. While the assumption that players in a game behave rationally is usually valid for *companies* (which seek to maximize their profit), employees as individuals may be much more affected by irrational thought patterns [21]. Default-bias and loss-aversion could possibly account for the reluctance of employees to adopt the Alternative Policy, whereas companies refrain from a policy change that is perceived as hurting its employees.

A key difficulty in applying the theory directly, is that even if these utility functions of companies and employees indeed exist, and even adhere to the properties we demanded, they are often not explicitly accessible. That is, a typical person does not know how much utility she gains from driving, say, an additional 100km per month. Nevertheless, employees do reach agreements that are more or less stable with their employers, even though utilities are implicit and difficult to estimate. As our proposed policy does not require more complicated decision making (it might even be simpler), we have every reason to believe that participants will continue to reach stable agreements (that cannot be too far from the equilibria we described), even after the policy change takes place.

### 7.3 Sustainable Transportation

The validity of our argument crucially depends on the existence and availability of alternative transportation methods, and alternative commuting solutions in particular. In the absence of these, employees will not be able to reduce their mileage,[8] and will not be able to benefit from the Alternative Policy. Direct and indirect subsidies for private cars were pointed out as having a negative effect on the development and embedding of alternative means of transportation [16]. Similarly, it is likely that the Common Policy creates a "vicious circle" in a way, since the large number of employees that have pre-paid fuel lowers the demand (and in turn the availability) of alternative solutions such as public transportation and car-pooling. The low availability of alternatives is used to justify the benefit of pre-paid fuel, and so on. Steps that are taken at the national level to support the alternatives, such as subsidies for trains or taxes on fuel, are less effective since they are not relevant for a significant part of the population.

Making *all* drivers face the true costs of their behavior should also assist in breaking this cycle and promote the availability of alternatives that would benefit the rest of the population.

Our paper demonstrates how economic theory supports an environment-friendly policy by eliminating externalities that affect both the players and the environment. In the words of P. B. Goodwin, this is an example of how a "gold-green" coalition can emerge [15].

### 7.4 Conclusion and Future Work

This paper deals with the optimal behavior of *rational* players under "pure" conditions, that is, with no external intervention. We proved that under these conditions, pre-paid fuel should not be considered a benefit. We hope that arguments such as the one presented in this paper can assist in removing obstacles such as the ones described in Section 7.2, by highlighting the negative role of a given taxation policy and convincing policymakers of the benefits of the transition.

We emphasize the fact that we do not suggest adding a new mechanism that will help reduce congestion, but rather to *remove* any external intervention in the form of a fuel subsidy, and let the market do its work.

We believe that some of the assumptions used in our model are perhaps too strong, and we intend to obtain stronger theoretical results by relaxing these assumptions where possible. It may also be possible to take into account irrational (yet predictable) aspects in the behavior of the involved parties in order to better understand the implications of each policy (similar to the approach taken by Bazzan et al. w.r.t. recommendation systems [7]). Finally, our work should be complemented by an experimental study on the effects of the suggested policy transition on real companies.

### Acknowledgments

We are indebted to the Transportation Today and Tomorrow organization and to its director Tamar Keinan for sharing with us their databases and ideas, and for their willingness to turn theory into practice. We also thank Ilan Nehama, Adi Meir, and Giora Alexandron for commenting on a draft of this paper.

## 8. REFERENCES

[1] Richard J. Arnott, André de Palma, and Charles Robin Lindsey. Departure time and route choice for the morning commute. *Transportation Research, Part B*, 24(3):209–228, 1990.

[2] Richard J. Arnott and Kenneth Small. The economics of traffic congestion. *American Scientist*, 82:446–455, 1994.

[3] F. Balbo and S. Pinson. Dynamic modeling of a disturbance in a multi-agent system for traffic regulation. *Decision Support Systems*, 41(1):131–146, 2005.

[4] Ana Bazzan, Felipe Boffo, and Franziska Klügl. Reducing the effects of the Braess paradox with information manipulation. *Application of Agent Technology in Traffic and Transportation*, pages 85–98, 2005.

[5] Ana Bazzan and Robert Junges. Congestion tolls as utility alignment between agent and system optimum. In *AAMAS '06: Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 126–128, 2006.

[6] Ana Bazzan and Franziska Klügl. Case studies on the Braess paradox: Simulating route recommendation

[7] Taxation policy was also pointed out by Windsor and Omer [25] as one of the contributors to the ubiquity of the leasing arrangement in Israel.

[8] In our model, this means that the function $f$ is almost linear. Conversely, when there are more alternatives, $f$ becomes "more concave" and the potential savings for all sides increases.

and learning in abstract and microscopic models. *Transportation Research, Part C*, 13:299–319, 2005.

[7] Ana Bazzan, Joachim Wahle, and Franziska Klügl. Agents in traffic modelling — from reactive to social behavior. *Advances in Artificial Intelligence, no. 1701 in LNAI*, pages 303–306, 1999.

[8] William R. Black. *Sustainable Transportation: Problems and Solutions*. The Guilford Press, 2010.

[9] Dietrich Braess. On a paradox of traffic planning. *Transport. Sci.*, 39(4):446–450, 2005.

[10] John Mc Breen, Pablo Jensen, and Fabrice Marchal. An agent based simulation model of traffic congestion. In *ATT '06: Proceedings of the 4th workshop on Agents in Traffic and Transportation*, pages 43–49, 2006.

[11] Vincent Conitzer and Tuomas Sandholm. Computing the optimal strategy to commit to. In *EC '06: Proceedings of the 7th ACM Conference on Electronic Commerce*, pages 82–90. ACM, 2006.

[12] Claudio Cubillos, Franco Guidi-Polanco, and Claudio Demartini. MADARP: Multi-agent architecture for passenger transportation systems. In *Proceedings of the 8th International IEEE Conference on Intelligent Transportation Systems*, 2005.

[13] Paul Davidsson, Lawrence Henesey, Linda Ramstedt, Johanna Törnquist, and Fredrik Wernstedt. Agent-based approaches to transport logistics. *Application of Agent Technology in Traffic and Transportation*, pages 1–16, 2005.

[14] Klaus Fischer, Jörg P. Müller, Markus Pischel, and Darius Schier. A model for cooperative transportation scheduling. In *ICMAS '95: Proceedings of the First International Conference on Multi-Agent Systems*, pages 109–116. The MIT Press, 1995.

[15] Phil B. Goodwin. Solving congestion. Inaugural Lecture for the Professorship in Transport Policy, University College London, 1997.

[16] Kerry Hamilton. If public transportation is the answer, what is the question? In Amanda Root, editor, *Delivering Sustainable Transport*, pages 49–59. Elsevier, 2003.

[17] Garrett Hardin. The tragedy of the commons. *Science*, 162:1243–1248, 1968.

[18] Israel's central bureau of statistics, motor vehicles 2008. Available online from http://www.cbs.gov.il/publications09/1381/pdf/e_print.pdf.

[19] Anna Nagurney. *Sustainable Transportation Networks*. Edward Elgar Publishing, 2000.

[20] J. Von Neumann and 0. Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, 1944.

[21] Richard H. Thaler and Cass R. Sunstein. *Nudge: Improving Decisions About Health, Wealth, and Happiness*. Yale University Press, 2008.

[22] Kagan Tumer, Zachary T Welch, and Adrian Agogino. Aligning social welfare and agent preferences to alleviate traffic congestion. In *AAMAS '08: Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 655–662, 2008.

[23] UK Department for Transport, transport statistics 2008. Available online from http://www.dft.gov.uk/pgr/statistics/datatablespublications.

[24] UK Department for Transport, A New deal for Transport: Better for everyone, 2000.

[25] Ahuva Windsor and Moshe Omer. Changing commuting patterns (in Hebrew). Available online from http://www.s-t.org.il/index_en.asp, 2007.

[26] Tomohisa Yamashita, Kiyoshi Izumi, and Koichi Kurumatani. Analysis of the effect of route information sharing on reduction of traffic congestion. *Application of Agent Technology in Traffic and Transportation*, pages 99–111, 2005.

# Promoting Effective Exchanges Between Vehicular Agents in Traffic Through Transportation-Oriented Trust Modeling

Umar Farooq Minhas† Jie Zhang† Thomas Tran‡ Robin Cohen†
†David R. Cheriton School of Computer Science, University of Waterloo, Canada
{ufminhas, j44zhang,rcohen@uwaterloo.ca}
‡School of Information Technology and Engineering, University of Ottawa, Canada
{ttran@site.uottawa.ca}

## ABSTRACT

In this paper, we focus on the problem of enabling vehicles in mobile vehicular ad-hoc networks (VANETs) to exchange information about traffic and road conditions in a way that makes it possible for each agent to assess the trustworthiness of the reports received. In particular, we develop a multi-faceted trust modeling framework that is designed specifically for VANET contexts, providing for trust modeling that includes reasoning about time and location and about agent roles, as part of the overall processing. We demonstrate the value of our trust modeling framework through simulated traffic environments, clarifying the importance of distinct elements of our multi-faceted model. In addition, we comment on the value of our chosen simulation environment towards future research to support more effective agent exchanges in VANETs.

## Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence - Intelligent agents, Multiagent systems; C.2.0 [**Computer Communication Networks**]: General – Security and protection; I.6.m [**Computing Methodologies**]: Simulation and Modeling

## General Terms

Algorithms, Design, Security, Experimentation

## Keywords

Trust modeling, VANET, Traffic management, Information exchange

## 1. INTRODUCTION

With the advancement in technology more and more vehicles are being equipped with GPS and Wi-Fi devices that enable them to communicate with each other, creating a vehicular ad-hoc network (VANET). Various studies have established the fact that the number of lives lost in motor vehicle crashes world-wide every year is by far the highest among all the categories of accidental deaths [1]. It is apparent that there is a dire need to enhance passenger and road safety which is precisely one of the goals of deploying vehicle to vehicle (V2V) communication systems. Another supporting goal is to be able to effectively route traffic through dense urban areas by disseminating up to date information regarding road condition through the VANET.

Some car manufacturers have already started to fit devices that will help achieve the goals mentioned above. For example, GM has rolled out V2V communication in its Cadillac STS Sedans. GM's proprietary algorithm called "threat assessment algorithm" keeps track of the relative position, speed and course of other cars (also equipped with V2V technology) in a quarter-mile radius and issues a warning to the driver when a crash is imminent [6]. Similar prototypes by other car manufacturers are currently in the testing phase, scheduled to hit the markets over the coming years.

Even though the initial algorithms and protocols that are being proposed by the car manufacturers are proprietary, it is believed that the standardization efforts carried out by Car-2-Car Consortium [25] will help to define a common interface for V2V communication technologies allowing its wide-spread use. Following this, it is very natural to assume that agent applications will be deployed whose main goal will be to assist the user in various ways using V2V communication. One such example is of an agent that gathers road congestion information and calculates the optimal route from a user's origin to destination thus bringing utility to the user. In such a scenario, we can view cars in a VANET as autonomous agents acting on behalf of their owners thus constituting a multi-agent network.

The agent would represent the motives of car owners who might as well decide to behave selfishly every now and then. For example, consider a user who instructs his agent to report the roads on his path as congested with the hope that other agents would avoid using these roads, thus clearing the path. Therefore one important issue among others that may arise in VANETs is the notion of trust among different agents. The goal of incorporating trust is to give incentives for these agents to behave honestly and to discourage self-interested behavior. These details are captured through what is called a *trust model*. Defined formally, "trust is a belief an agent has that the other party *will do what it says it will* (being honest or reliable) or *reciprocate* (being reciprocative for the common good of both), given an opportunity to defect to get higher payoffs" [17]. Here it is important to clarify that our notion of trust always refers to the trust placed by one agent in another agent which is different from the trust placed by the user (or driver) in the agent itself and is beyond the scope of this work. A closely related notion called reputation is defined as the opinion or view of an agent about another agent that is either directly acquired from the environment or from other agents and ultimately leads to building of trust [17]. Given the critical nature of agent applications in the context of VANETs, it is crucial to associate trust with agents and the data that they spread.

With respect to the general topic area of agents in traffic and transportation, our research can be characterized most appropriately as focusing on the challenge of enabling autonomous vehicles to engage in collaborative driving and in intelligent peer-to-peer interactions to enable distributed decision making in traffic.

### 1.1 The challenges of VANET trust modeling

Modeling trustworthiness of agents in VANETs presents some unique challenges. First of all, the agents in a VANET are constantly roaming around and are highly dynamic. On a typical highway the average speed of a vehicle is about 100 kilometers an hour. At high speeds the time to react to an imminent situation is very critical [2], therefore, it is very important for the agents to be able to verify/trust incoming information in *real-time*. Second, the number of agents in VANET can become very large. For example, in dense urban areas the average amount of vehicles that pass through the network may be on the order of millions and several thousand vehicles will be expected to be present in the network at any given time. Also this situation is exacerbated during the rush hours when, for example, majority of the people commute to and back from work in a metropolitan area. This may introduce several issues some of which include network congestion - since vehicles are communicating on a shared channel, information overload - resulting from vehicles receiving a lot of data from the near-by vehicles in a congested area etc. Hence there will be a need to have intelligent vehicle communication systems that are *scalable* and can detect and respond to these potentially hazardous situations by effectively deciding with which agents to communicate [11].

Another key challenge in modeling trust in a VANET environment is that a VANET is a *decentralized*, open system i.e. there is no centralized infrastructure and agents may join and leave the network any time respectively. If an agent is interacting with a vehicle now, it is not guaranteed to interact with the same vehicle in the future [5]. Therefore, it is not possible to rely on mechanisms that require a centralized system (e.g. the Centralized Certification Authority and the Trusted Third Party etc) or social networks to build long-term relationships.

Also, information about road condition is rapidly changing in VANET environments, e.g. a road might be busy 5 minutes ago but now it is free, making it hard to detect if the agent spreading such information is malicious or not. This also brings out an important challenge that the information received from VANETs needs to be evaluated in a particular context. The two key context elements in VANETs are *location* and *time*. Information which is closer in time and location of an event is of more relevance. We explain this in more detail in Section 2.

Various trust and reputation models (e.g. [20] and [30]) have been studied with reference to multi-agent environments, however, given the unique characteristics of agents in VANETs the existing models cannot be applied directly. For example, several trust and reputation models are built around the assumption that the agents can have multiple direct interactions with other agents and hence they fail when applied to VANETs, since the interactions between agents in this environment may be quite sparse.

The main goal of this work is then to develop a framework that can effectively model the trustworthiness of the agents of other vehicles in VANETs. We propose a novel multi-faceted approach for modeling trust in VANET environments that incorporates role-based trust, experience-based trust, priority-based trust and majority-based trust and that is able to restrict the number of reports that are received from other agents. Our expanded trust model is aimed to be decentralized, location/time specific, event/task specific, able to cope with the data sparsity problem, cumulative in order to be scalable, sensitive to privacy concerns, and able to support system-level security. We present the design of this model in detail, clarifying how it meets various critical challenges for trust modeling in VANET environments. We also step through a detailed procedure of computing trustworthiness of agents and generating effective responses to information sent by those agents. We finally demonstrate its value in a simulated vehicular setting. The result is an important first step towards the delivery of effective intelligent vehicular communication, one that is sensitive to the trustworthiness of the vehicular agents.

As will be seen, we introduce a framework that is amenable to dynamically changing networks of agents (a desirable quality for VANETs, as explained in [19]), in contrast with other trust models that are designed to operate in more stable environments (e.g. [27]) or that assume complete knowledge of all the agents in the system (e.g. [15]). In addition, our inclusion of roles as part of the trust modeling framework can be seen as an element to overcome the more typical sparsity of relationships which compromises an approach to trust modeling relying solely on social networks (e.g. [29]).

## 2. EXPANDED TRUST MANAGEMENT

From the discussion in previous sections, it becomes apparent that no single trust or reputation mechanism can work particularly well for the challenge of modeling trust effectively for VANET environments. Instead of just having one or two trust metrics for evaluating trust, there is a need to have several different trust metrics with various key properties in order to capture the complexity that arises between interacting agents in VANET. We propose that in order to derive a rather complete and comprehensive view of trust for agents in VANET we will need to integrate security solutions (at the system level) for trust management, i.e. secure storage of role identities for role-based trust in our proposal.

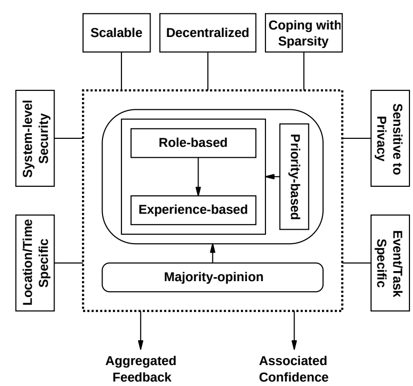


**Figure 1: Expanded Trust Management**

Figure 1 illustrates the design of our expanded trust management. The core of the management is grouped by the dashed rectangle in the middle. This core consists of two parts. One part maintains trustworthiness of agents in order for trusted agents (advisors) to be chosen to ask for their feedback. More specifically, in this part, the trustworthiness of agents is modeled based on role-based trust and experience-based trust, which are both combined into the priority-based model that can be used to choose proper advisors.

Our role-based trust exploits certain predefined roles that are enabled through the identification of agents (vehicles). For example, agents can put more trust in certain agents as compared to others, i.e. agents identified as law enforcing authorities or owned by government [19]. Our experience-based trust represents a component of trust that is based on direct interactions. It is in the same spirit of incorporating evidence from direct interactions into trust calculation through Interaction Trust as proposed by [8] or the Individual Dimension of trust in the model as proposed by [23]. Implementation and formalization of these two trust metrics will be presented in Section 2.2.

The other part of the core is a majority-opinion approach to aggregate feedback from selected advisors. Detailed procedures for these processes will be further discussed in Section 2.2. More importantly, our management of trust has several key properties represented by rectangles around the core in the figure. Our trust management is aimed to be decentralized, location/time specific, event/task specific, able to cope with the data sparsity problem, cumulative in order to be scalable, sensitive to privacy concerns, and able to support system-level security. These properties will be extensively discussed in Section 2.1, respectively. Note that the property of system-level security is mentioned in different places where we discuss other properties and our model, i.e. secure storage of role identities in Section 2.1.1, verification of time/location of reported events in Section 2.1.3, and identification of agents' roles in Section 2.2.3.

The outcome of our trust management is aggregated feedback for a certain request/event and an associated confidence value for it. The aggregated feedback is eventually affected more heavily by highly trusted advisors. The value of confidence would depend on the reliability of estimated experience-based trust of each other agent and the maximum accept error rate for the aggregated feedback. In general, a higher value of confidence, i.e. a value closer to 1, would result from considering more evidence or metrics having high reliability, for a fixed error rate. We can view confidence as a parameter that adds another dimensionality to the output generated by the model allowing the agent applications to have a richer notion of trust and finally decide how to react on the reported event. Our notion of confidence is somewhat tantamount to the notion proposed in [24, 8].

## 2.1 Key Properties

We provide here detailed discussion of the seven key properties that our trust management incorporates. These properties guide our design of the expanded trust management, which can be applied to the problem of trust management in VANET.

### *2.1.1 Decentralized Trust Establishment*

Models which depend on a central entity for the reliable establishment of trust are not appropriate for the domain of VANET because of its highly distributed property. Therefore, we propose that trust establishment should be fully decentralized to be applicable to the highly dynamic and distributed environment of VANETs.

Our experience-based trust model makes use of agents' direct interactions to update one agent's belief in the trustworthiness of another. This one-to-one interaction can easily be implemented in a distributed manner. Our role-based trust can also be done in a totally decentralized manner among the vehicles themselves. For this to work, we may involve the car manufacturers, or transportation authorities to issue certificates at the manufacture or registration time respectively. A public-private key infrastructure for verifying each other's roles can be implemented in a distributed manner. Also there would be a need to store these certificates and keys in a way that they cannot be manipulated or tampered with, to archive high security. To this end, researchers [16] who have done studies with the goal of securing VANET communications have unanimously proposed the use of a tamper proof device that stores e.g. the cryptographic keys issued by authorities. If any attempt to tamper the device is made, the keys are destroyed automatically stripping the agent from its ability to communicate with other agents thus effectively destroying its means of deriving any utility at all.

### *2.1.2 Coping with Sparsity*

Effective trust establishment should not be contingent upon a minimum threshold for direct interactions. As we have described at several places, it should not be expected that an agent in VANET would possibly interact with the same agent more than once. However, it is important to clarify here that the trust models should still be able to effectively take into consideration any data available from direct interaction (even though it might happen just once). Thus, in a scenario where the number of agents that are able to spread information has gone down to the extent that the condition of information scarcity or a total lack of information is prevalent, any data might be termed valuable. In the trust calculation, the weight for the data can be raised in this scenario while it may have a lower default value, to cope with the data sparsity problem in VANET.

We also have the role-based trust approach to distinguish trustworthy agents from untrustworthy ones to some extent. When an experience-based trust approach is used, we also introduce the idea of allowing agents to send testing requests, to deal with sparsity. The senders of these testing requests basically know the solution to these requests in advance. Imaging a group of agents driving in a city from one location to another, they remain in contact range for a certain period of time. These agents can send testing requests to each other and evaluate their feedback. Trust between them can then be established through the experience-based trust in our management model.

### *2.1.3 Event/Task and Location/Time Specific*

Since the environment of the agents in VANET is changing constantly and rapidly, a good trust model should introduce certain dynamic trust metrics, capturing this dynamism by allowing an agent to control trust management depending on the situation at hand [19, 4]. Here, we separately deal with two particularly important dynamic factors in the context of VANETs, event/task and location/time.

Agents in general can report data regarding different events e.g. car crashes, collision warnings, weather conditions and information regarding constructions etc. Our trust management should therefore be event/task specific. For example, some of these tasks may be time sensitive and require quick reaction from the agent that receives them. In this case, this agent can only consult a very limited number of other agents to verify whether the reported information is true. In another case, reporting agents having different roles in VANET may have more or less knowledge in different types of tasks. For example, a police may know more about car crash information while city authorities may know more about road construction information. Thus, our role-based trust should be task specific. One way to implement this in our role-based trust model is to have a set of events associated with a set of roles of agents (e.g. law enforcement, municipal authorities). This information can be used later for an agent to choose particular other agents to consult regarding a particular event. Our experience-based trust is also event specific. An agent updates the reporting agent's trust by taking into account the type of the reported event. For example, life-critical events will certainly have more impact on the reporting agent's trust.

We also note that location and time are another two particularly important dynamic metrics. For example, if the origin of a certain message is closer to the location of where the reported event has taken place, it might be given a higher weight, relying on the underlying assumption that an agent closer to the event is likely to report more realistic data about the event (given that they are not malicious themselves). Similarly, we can apply this concept to time. If the message reporting a certain event is received closer to the time when the reported event has taken place, it might be allowed a higher weight in trust calculation. Another suggestion that naturally follows from time based trust is that, since the relevance of data in VANET is highly dependent on when it was received, it would make sense to assign a decay factor to the message. The message further away from the time of evaluating trust would be assigned a lower weight. In other words, we should decay the impact of message relative to the time of the trust evaluation. The decay factor is somewhat analogous to the time-to-live (TTL) field used in IP packets.

The first issue that may arise with calculating time or location specific trust is how to get location and time of the actual event. We expect that whenever a report regarding an event is generated to be shared among other agents it will hint to the time at which this event has taken place, giving us the required time information. Also we assume that every agent while transmitting the report appends its location with the report. The next issue is to verify whether the time and location information contained within a report is real or spoofed. With this regard, [7] has proposed a method to accurately estimate the location of nearby agents. However, complete treatment of this issue is beyond the scope of this work. Now the next task would be to actually use the location/time information in trust management. In the calculation of subjective reputation as proposed by [23] they use a weighted sum of trust values suggesting that the weights should be adjusted such that higher weights are assigned to the agents closer to the agent which is calculating trust. In a similar fashion, we can extend their model by instead of defining the closeness between agents; we define the location closeness between the actual event and the agent reporting this event. For the time based trust a similar calculation can be done by modifying the notion of time closeness as that between the time when the event has taken place and that of receiving the report.

#### 2.1.4 Scalable

Scalability is an important aspect in trust management in VANET environments. In our system, each agent consults only a number of other trusted agents. This number can be fixed or slightly updated with the changes in, for example, VANET size or the task at hand. However, it is always set to a value small enough to account for scalability.

Establishing trust in VANETs using experience-based trust requires each agent to store the history of past interactions with other agents and to compute their trust based on that information. For the purpose of being scalable, our experience-based trust model updates agents' trustworthiness by accumulatively aggregating agents' past interactions in a recursive manner, similar to [10]. The computation of our experience-based trust is thus linear with respect to the number of interactions. And only the most recent trust values are needed to be stored and used for computation. This design makes our trust management scalable.

#### 2.1.5 Sensitive to Privacy Concerns

Privacy is an important concern in a VANET environment. In this environment, the revealing of a vehicle owner's identity (e.g. the owner's home address) may allow a possibly malicious party to cause damage to the owner. Our trust management makes use of a public key infrastructure (PKI) allowing agents to authenticate each other. In our system, when an agent sends a report to another agent, the sender needs to authenticate itself to the receiver that it has a certain role. Although these keys do not contain any sensitive identities of the sender, the receiver may be able to track them by logging the messages containing the key of the sender. For example, the receiver can track the likely home address of the sender by finding out the route of the sender if the receiver has sufficient information about different locations that the sender has been to, and therefore other identities. This issue can be addressed by changing keys, as suggested in [18]. Each agent in VANET will store a large set of pre-generated keys and certificates. It will change keys while sending information to others regarding some privacy sensitive locations of the sender (i.e. places nearby home), so that others do not recognize this sender as one of the previous senders that they have interacted with. In this way, others will not be able to discover the sender's privacy sensitive identities, while they will still be able to keep track of experience with this sender regarding some insensitive locations of the sender.

## 2.2 Computation Procedure

In this section, we briefly outline the procedure taken by an agent to make a decision for a (requested) task/event by aggregating reports about this task from other trusted agents and to update their experience-based trust values afterwards.

#### 2.2.1 Scenarios

An agent in a VANET environment may actively send a request to a list of trusted neighboring agents about a task, i.e. weather or direction information. In another scenario, it may passively wait for other agents to send reports about an event, i.e traffic or collision ahead of the agent. Once it receives a report about an event from another agent, it may trust the information if it has high confidence that the report sender can be trusted. Otherwise, it may need to verify (double check) if the information given by the sender is reliable by asking other trusted agents. In both scenarios, the agent will need to aggregate senders' reports. Values calculated in this manner can then be used by the agent to decide whether to believe a particular report and take corresponding actions. For this purpose, each agent in our system keeps track of a list of other agents. This agent updates all report senders' trustworthiness after the truth of their reported events is revealed. The above two processes of aggregating reports and updating trust will take into account the context in general, this agent's notion of which other agents it is interacting with, the notion of which group the other agents belong to or the roles assigned to the other agents, the time of reported event together with the time of message arrival, the relative locations of the other agents, and the actual contents of the message to evaluate task/event specific trust etc. Next, we provide detailed description and formalization of each step in our computation procedure.

#### 2.2.2 Computation Steps

Four elements are incorporated into our overall trust management as its core, shown in Figure 1: 1) Experience-based trust; 2) Role-based trust; 3) Majority opinion (or social network of trust); 4) Priority-based trust. Our computation procedure consists of four steps.

**Step 1:** Depending on the task at hand, set a value $n$ = number of agents whose advice will be considered. This incorporates task-based trust. For example, if you need a very quick reply, you may limit $n = 2$ or 3; if you are planning ahead and have time to process responses, $n$ could potentially be larger.

**Step 2:** Using $n$, construct an ordered list of agents to ask. The list will be partitioned into groups as follows:

$$\begin{bmatrix} G_1: & a_{11}, & a_{12}, & a_{13}, & ..., & a_{1k} \\ G_2: & a_{21}, & a_{22}, & a_{23}, & ..., & a_{2k} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ G_j: & a_{j1}, & a_{j2}, & a_{j3}, & ..., & a_{jk} \end{bmatrix}$$

where $kj = n$.[1] This priority list is ordered from higher roles to lower roles, for example, $G_1$ being the highest role. Within each

[1]There is no need for each group to have the same number of elements. We provide here only an simplified example.

group of agents of similar roles, the group is ordered from higher (experience-based) ratings to lower ratings. Thus, $a_{ij}$ represents the agent in role class $i$ that is at the $j^{th}$ level of experience, relative to other agents at that level. Hence, role-based trust and experience-based trust are combined into this priority-based approach. These two trust metrics will be further discussed later in this section.

**Step 3:** When an agent requires advice, the procedure is to ask the first $n$ agents the question, receive the responses and then perform some majority-based trust measurement.

Step 3B: The processing of the responses is as follows: if there is a majority consensus on the response, up to some tolerance that is set by the asker (e.g. I want at most 30% of the responders to disagree), then this response is taken as the advice and is followed. We will formalize this majority-based trust in Section 2.2.5.

Step 3C: Once this advice is followed, the agent evaluates whether this advice was reliable and if so, personal experience trust values of those agents are increased; if not, personal experience trust values of those agents are decreased. Detailed formalization of this process will be given in Section 2.2.4.

Step 3D: If a majority consensus cannot be reached, then requiring majority consensus for advice is abandoned. Instead, the agent relies on role-based trust and experience-based trust (e.g., taking the advice from the agent with highest role and highest experience trust value)[2].

**Step 4:** In order to eventually admit new agents into consideration, when advice is sought, the agent will ask a certain number of agents beyond agent $a_n$ in the list. The responses here will not count towards the final decision, but will be scrutinized in order to update personal experience trust values and some of these agents may make it into the top $n$ list, in this way.

Algorithm 1 is a pseudo code summary of the proposed algorithm. Note that this pseudo code covers the main scenario where an agent actively requests other agents for advice and does not include the exploration/testing step (Step 4).

**Algorithm 1:** Computation Steps

```
while on the road do
    if in need of advice then
        Choose n; //number of agents to ask for advice
        //according to roles and experience
        Prioritize n agents;
        Send request and receive responses;
        if response consensus > acceptable ratio then
            Follow advice in response;
        else
            Follow advice of agent with highest role and
            highest trust value;
    Verify reliability of advice;
    Update agents' trust values;
```

### 2.2.3 *Role-based Trust*

Our role-based trust exploits certain predefined roles assigned to all agents in the system. The underlying assumption here is that the agents identified by authorities are more closely monitored and are expected to behave in a certain way. We can also conceptualize roles as an expected behavior of a certain group or class of agents where all the agents belonging to a group would behave similarly. We propose a role-based approach because the expected number of possible roles and the rules to assign these roles would be very few in the domain of VANETs and thus can be manually managed and/or updated by a trusted authority. Note that the concept of seniority (expertise in a certain context/task, for instance) could be incorporated into role-based trust, as mentioned in Section 2.1.3.

To demonstrate our role-based approach, let's consider a simple system that recognizes the following four different roles listed in decreasing order[3], i.e. from the highest role to the lowest one: 1) authority, 2) expert, 3) seniority, and 4) ordinary. Each role level may also be associated with a trust value $T_r \in (0, 1)$ where higher level roles have larger $T_r$ values. The rules for assigning and authenticating these roles can be structured as follows:

1. Agents representing authorities such as traffic patrols, law enforcement, state or municipal police etc. assume the authority role.

2. Agents specialized in road condition related issues such as media (TV, radio or newspaper) traffic reporters, government licensed and certified instructors of driving school etc. receive the expert role.

3. Agents familiar with the traffic or road conditions of the area in consideration, e.g. local people who commute to work on certain roads or highways or have many years of driving experience with a good driving record (e.g. taxi drivers), are given the seniority role.

4. All other agents are considered having the ordinary role.

All agents should possess certificates issued by a trusted certificate authority for authentication purpose. Note that we need a way for an agent to tell if another agent is indeed having the role that he is claiming to have. One possible solution to this problem is to make use of public-key certificates in an asymmetric cryptosystem as follows: Each agent should have a public key certificate, which can simply be a document containing the agent's name, his role and his public key. That document is signed by a trusted certificate authority (with the certificate authority's private key) to become the agent's public key certificate. Everyone can verify the authority's signature by using the authority's public key. Now, when agent $A$ sends a message to agent $B$, $A$ must sign the message with his private key. $B$ then can verify (using $A$'s public key) that the message was truly sent by $A$.

### 2.2.4 *Experience-based Trust*

We track experience-based trust for all agents in the system, which is updated over time, depending on the agent's satisfaction with the advice given, when asked. As mentioned in the previous section, our experience-based trust is cumulative in the sense that it updates agents' trust recursively. Thus, only the most recent trust values and the number of interactions between agents are needed to be stored in the system, to make the system scalable. We here formalize the computation of this trust. If we define the range of all personal experience trust values to be the interval $(-1, 1)$, where $1$ represents absolute trust and $-1$ represents absolute distrust, then

[2]Note that an additional motive for modeling the trustworthiness of a variety of agents is to be able to learn about these agents for future interactions, for example in the calculations of experience-based trust and majority-opinion trust.

[3]Our experience-based trust may be helpful for role categorization. When agents have sufficient experience-based trust information about each other, they may report this information to a trusted authority (i.e. the transportation department of government). A mapping between agents' real-world profiles and their trustworthiness can then be derived for helping categorize their roles.

we can use the following scheme to update an agent's personal experience trust value[4], as suggested by [26]:

Let $T_A(B) \in (-1, 1)$ be the trust value indicating the extent to which agent $A$ trusts (or distrusts) agent $B$ according to $A$'s personal experience in interacting with $B$. After $A$ follows an advice of $B$, if the advice is evaluated as reliable, then the trust value $T_A(B)$ is increased by

$$T_A(B) \leftarrow \begin{cases} T_A(B) + \alpha(1 - T_A(B)) & \text{if } T_A(B) \geq 0, \\ T_A(B) + \alpha(1 + T_A(B)) & \text{if } T_A(B) < 0, \end{cases} \quad (1)$$

where $0 < \alpha < 1$ is a positive increment factor.

Otherwise, if $B$'s advice is evaluated as unreliable, then $T_A(B)$ is decreased by

$$T_A(B) \leftarrow \begin{cases} T_A(B) + \beta(1 - T_A(B)) & \text{if } T_A(B) \geq 0, \\ T_A(B) + \beta(1 + T_A(B)) & \text{if } T_A(B) < 0, \end{cases} \quad (2)$$

where $-1 < \beta < 0$ is a negative decrement factor.

The absolute values of $\alpha$ and $\beta$ are dependent on several factors because of the dynamics of the environment, such as the data sparsity situation mentioned in Section 2.1.2 and the event/task specific property mentioned in Section 2.1.3. For example, when interaction data is sparse, these values should be set to be larger, giving more weights to the available data. For life-critical events (i.e. collision avoidance), $|\alpha|$ and $|\beta|$ should be larger, in order to increase or decrease trust values of reporting agents more rapidly. Also note that we may set $|\beta| > |\alpha|$ by having $|\beta| = \mu|\alpha|$ and $\mu > 1$ to implement the common assumption that trust should be difficult to build up, but easy to tear down.

We also incorporate a forgetting factor $\lambda$ ($0 < \lambda < 1$) in Equations 1 and 2, allowing $A$ to assign less weight to older interactions with $B$. This is to cope with the possible changes of $B$'s behavior over time. If we define $t$ as the time difference between the current interaction and the previous one[5], the equations then become

$$T \leftarrow \begin{cases} \lambda^t(1 - \alpha)T + \alpha & \text{if } T \geq 0, \\ \lambda^{-t}(1 + \alpha)T + \alpha & \text{if } T < 0, \end{cases} \quad (3)$$

$$T \leftarrow \begin{cases} \lambda^t(1 - \beta)T + \beta & \text{if } T \geq 0, \\ \lambda^{-t}(1 + \beta)T + \beta & \text{if } T < 0, \end{cases} \quad (4)$$

where we substitute $T_A(B)$ by $T$ for the purpose of clarity. The trust values $A$ has of $B$ will increase/decrease more slowly than those in Equations 1 and 2 because older interactions between them are discounted and have less impact on the current trust values.

The number of interactions between agent $A$ and agent $B$, $N_A(B)$, should also be discounted accordingly. This can also be done recursively as follows:

$$N_A(B) = \lambda^t N_A(B) + 1 \quad (5)$$

Note that the experience-based formulae are also valuable to cope with agents who try to build up trust and then deceive. Once deception is detected, trust can be torn down quite quickly.

### 2.2.5 Majority Opinion and Confidence

Suppose agent $A$ in VANET receives a set of $m$ reports $\mathcal{R} = \{R_1, R_2, ..., R_m\}$ from a set of $n$ other agents $\mathcal{B} = \{B_1, B_2, ..., B_n\}$ regarding an event. Agent $A$ will consider more heavily the reports sent by agents that have higher level roles and larger experience-based trust values. When performing majority-based process, we also take into account the location closeness between the reporting agent and the reported event, and the closeness between the time when the event has taken place and that of receiving the report. We define $C_t$ (time closeness), $C_l$ (location closeness), $T_e$ (experience-based trust) and $T_r$ (role-based trust). Note that all these parameters belong to the interval $(0, 1)$ except that $T_e$ needs to be scaled to fit within this interval.

For each agent $B_i$ ($1 \leq i \leq n$) belonging to $\mathcal{B}(R_j) \subseteq \mathcal{B}$ that report a same report $R_j \in \mathcal{R}$ ($1 \leq j \leq m$), we aggregate the effect of its report according to the above factors. The aggregated effect $E(R_j)$ from reports sent by agents in $\mathcal{B}(R_j)$ can be formulated as follows:

$$E(R_j) = \sum_{B_i \in \mathcal{B}(R_j)} \frac{T_e(B_i)T_r(B_i)}{C_t(R_j)C_l(B_i)} \quad (6)$$

In this equation, experience-based trust and role-based trust are discounted based on the two factors of time closeness and location closeness. The summation is used to provide the aggregated effect of the reporting of the agents.

Note that location closeness $C_l(B_i)$ depends only on the location of agent $B_i$ while time closeness $C_t(R_j)$ depends on the time of receiving the report $R_j$. $C_t(R_j)$ can also be written as $C_t(B_i)$ because we can assume that each report is sent by an unique agent in possibly different time.

To consider the effect of all the different reports, the majority opinion is then

$$M(R_j) = \arg\max_{R_j \in \mathcal{R}} E(R_j) \quad (7)$$

which is the report that has the maximum effect, among all reports.

A majority consensus can be reached if

$$\frac{M(R_j)}{\sum_{R_j \in \mathcal{R}} E(R_j)} \geq 1 - \varepsilon \quad (8)$$

where $\varepsilon \in (0, 1)$ is set by agent $A$ to represent the maximum error rate that $A$ can accept. A majority consensus can be reached if the percentage of the majority opinion (the maximum effect among different reports) over all possible opinions is above the threshold set by agent $A$.

If the majority consensus is reached, the majority opinion is associated with a confidence measure. This measure takes into account the number of interactions taken for modeling experience-based trust values of reporting agents and the maximum accepted error rate $\varepsilon$. We define $N(R_j)$ as the average of the discounted number of interactions used to estimate experience-based trust values of the agents sending the majority report $R_j$ calculated using Equation 5. Based on the Chernoff Bound theorem [14], the confidence of the majority opinion can be calculated as follows:

$$\gamma(R_j) = 1 - 2e^{-2N(R_j)\varepsilon^2} \quad (9)$$

## 3. EXPERIMENTAL EVALUATION

In this section, we present preliminary evaluation of our trust model. We use SWANS (Scalable Wireless Ad-hoc Network Simulator, jist.ece.cornell.edu) with STRAW (STreet RAndom Waypoint) mobility model [3]. SWANS is entirely implemented in Java and can simulate networks with potentially thousands of nodes while using incredibly small amount of memory and processing power. STRAW allows to simulate real world traffic by using real

[4]A "commuter pool" might for instance offer significant experience

[5]The value of $t$ may be scaled within the range of $[0, 1]$. This can be achieved by setting a threshold $t_{max}$ of the maximum time for an agent to totally forget the experience happened at the time that is $t_{max}$ prior to the current time.

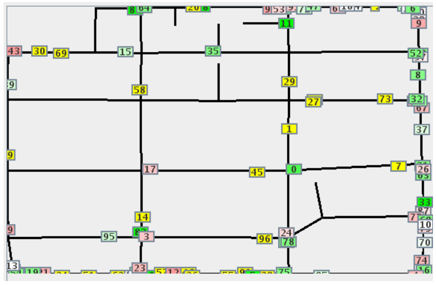

**Figure 2: Simulating VANET using SWANS Simulator with STRAW Mobility Model**

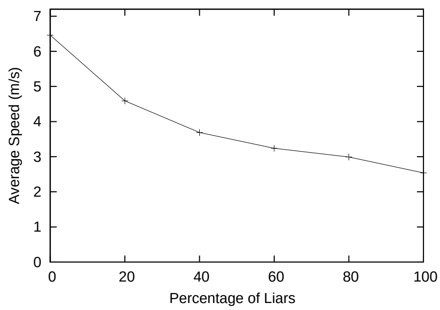


**Figure 3: Average Speed of All Cars When There are Different Percentages of Liars**

maps with vehicular nodes that follow rules such as speed limits, traffic signals, stop signs etc.

We use a map of north Boston, USA. Figure 2 shows a snapshot of one of our simulation runs. The bold lines are the extracted road segments from the map. The small rectangles labelled by integers represent vehicles running on the streets. For all our experiments we fix the total number of vehicles to 100 and run the simulation for a total duration of 900 seconds of simulation framework time. Note that in this paper we only experiment with the role-based and experienced-based dimensions of our trust model while leaving more comprehensive experimental evaluation for future work.

## 3.1 Performance Metric

One of the applications of V2V communication is to be able to route traffic effectively through the VANET and to avoid congestion or hot spots. Malicious agents in the network may send untruthful traffic information, to mislead other agents and cause traffic congestion. We measure the performance of our proposed trust model by observing to what extent it can cope with deceptive information sent by malicious agents. According to [3], we can measure congestion based on the average speed of vehicles. Lower average speed implies more traffic congestion. The performance of our model can then be measured as the increase in average speed of all agents by incorporating our model under the environment where malicious agents exist.

## 3.2 Results

We present experimental results to clearly show the value of different trust metrics integrated in our expanded trust management and to demonstrate that the combined one is the most effective.

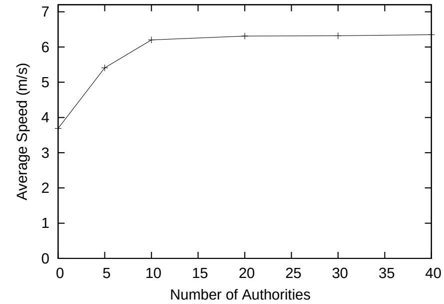


**Figure 4: Average Speed of All Cars When There are Different Numbers of Authorities**

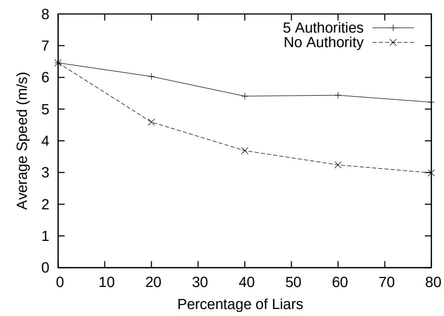


**Figure 5: Average Speed of All Cars When There are Five Authorities**

### 3.2.1 *Effect of Liars on Average Speed*

In our first experiment, we vary the percentage of malicious nodes in the environment and measure the change in average speed of the vehicles in the network (a measure advocated in [3]). We choose a lying strategy for the malicious nodes where they always lie about congestion on a particular road segment i.e., report congestion when there is no congestion and vice versa. We present the results in Figure 3. As expected, average speed of vehicles in the network decreases as the percentage of liars increases.

### 3.2.2 *Countering Liars with Role-based Trust*

Next we experiment with role-based trust where we introduce some agents in the environment with the role of authorities as mentioned in Section 2.2.3. In our simulation, authorities are assumed to be always trustworthy. In this experiment, we fix the number of malicious agents to be 40% and then vary the number of agents with the role of authority between 0 and 40. These results are presented in Figure 4. With an increase in the number of authorities in the environment, the overall average speed of the nodes increases, countering the effect of malicious agents. This shows the effectiveness of role-based trust in our model. From Figure 4, we can see that the average speed reaches a maximum with about 20 authorities. Figure 5 shows that even if we have a small number of agents with a role of authority in the system, we can still effectively cope with an increasing percentage of malicious nodes.

### 3.2.3 *Countering Liars with Experience-based Trust*

In this experiment, we employ only the experience-based dimension of trust. We vary the percentage of liars and measure the overall average speed of vehicles. As we can see from Figure 6, using experience-based trust results in an increase in the average speed of vehicles. This trend is consistent for all percentages of liars in

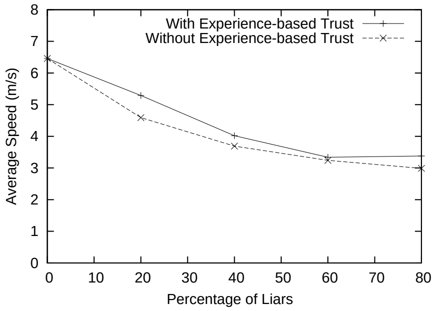


**Figure 6: Average Speed of All Cars with Experience-based Trust**

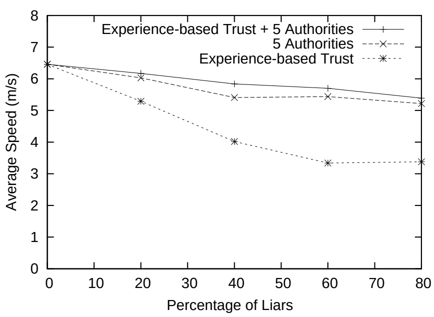


**Figure 7: Average Speed of All Cars with Role-based and Experience-based Trust**

the system which shows that experience-based trust is able to cope with the lying behavior of malicious agents.

Note that the performance of our trust model, namely the speed of vehicles, is averaged over the total duration of only 900 seconds of the simulation framework time. At the beginning of the simulation an agent does not yet have any experience with other agents. This explains the model's moderate performance during this early period.

### *3.2.4 Combining Role-based and Experience-based Trust*

From Figures 5 and 6, we can see that even though experience-based trust results in an increase in the average speed of vehicles in the network with the presence of malicious agents, role-based trust does this job more effectively. In this experiment, we combine both dimensions together and measure the average speed. These results are presented in Figure 7. As we can see, by combining these two dimensions we can achieve an average speed which is higher than when using any one of these two dimensions individually. This shows that a trust model for agents in VANETs can greatly benefit by combining several dimensions of trust as proposed in this work.

### *3.2.5 Coping with Sparsity*

This experiment is carried out to demonstrate the property of our model in coping with the data sparsity problem. In this experiment, we involve 50 nodes and run the simulation for 300 seconds of simulation framework time. We reduce the ratio of communication between nodes. The available data for modeling the trustworthiness of nodes is more sparse when the communication ratio is lower. As can be seen from Figure 8, the percentage of detecting malicious nodes decreases when the ratio of communication is reduced. By decreasing the value of $\beta$, the ability of detecting malicious nodes

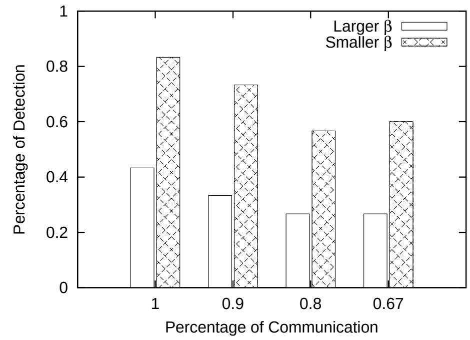


**Figure 8: Coping with Sparsity**

is increased dramatically[6]. This indicates that our model is able to cope with the data sparsity problem by changing the parameter $\beta$ to adjust the weight of available data.

The role-based trust in our model is also able to cope with data sparsity. As shown in Figure 7, with only the experience-based trust, the performance difference of our model between more and fewer liars is large. This difference is reduced when the role-based dimension is also used. The role-based trust reduces the impact of more liars and therefore is able to begin to cope with the data sparsity problem.

## 4. RELATED WORK

Lin et al. [12] have investigated the benefits achieved by self-interested agents in vehicular network through simulations. They consider a scenario where agents can achieve road congestion information from other agents through Gossiping. Two different behaviors of self-interested agents are investigated 1) Agents want to maximize their own utility 2) Agents want to cause disorder in the network. Simulation results indicate that for both behaviors, self-interested agents have only limited success in achieving their goals, even if no counter measures are taken. However, the authors realize the need to take these preliminary results to more complex and potentially more damaging scenarios that may arise in VANETs. They also identify the need to establish trust in vehicular ad-hoc networks through distributed reputation mechanisms, motivating our work. In contrast to the traditional view of entity-level trust, Raya et al. [19] propose that data-centric trust may be more appropriate in the domain of Ephemeral Ad Hoc Networks such as VANETs. Data-centric trust establishment deals with evaluating the trustworthiness of the data reported by other entities rather than trust of the entities themselves. Even though there are some commonalities between our approach and theirs, for example, they also propose the use of task/event specific trust metrics as well as time and location closeness but we combine these metrics in a fundamentally different way taking the traditional view of entity-level trust instead of data-centric trust. One of the shortcomings of their work is that trust relationships in entities can never be formed, only ephemeral trust in data is established, and because this is based on a per event basis, it needs to be established again and again for every event. This will work so long as there is enough evidence either in support of or against a specific event, but in case of data sparsity we believe our model would perform better. We leave a detailed comparison between these two models for future work.

[6]The absolute value of $\beta$ in Equation 2 reflects the weight placed on available data. Since $-1 < \beta < 0$, decreasing the value of $\beta$ will increase its absolute value, and the weight of data will also be increased.

Dotzer [4] has suggested building a distributed reputation model that exploits a notion called opinion piggybacking where each forwarding agent (of the message regarding an event) appends its own opinion about the trustworthiness of the data. They provide an algorithm that allows an agent to generate an opinion about the data based on aggregated opinions appended to the message and various other trust metrics including direct trust, indirect trust, sender based reputation level and Geo-Situation oriented reputation level. This last trust metric allows their model to introduce some amount of dynamism in the calculation of trust by considering the relative location of the information reporting node and the receiving node. Additionally, the situation oriented reputation level allows a node to consider certain situational factors e.g. familiarity with the area, rural or metropolitan area etc. again introducing some dynamism in trust evaluation based on context. Our model has direct trust in the form of experience-based trust, indirect trust in the form of role-based trust. Furthermore, we also use location closeness in our model that is similar to Geo-Situation oriented reputation level in their model. However, we provide an algorithm to combine, for example, experience-based and role-based trust into a priority-based trust, at the same time taking the majority opinion into account. This way of combining these different metrics is a novel feature of our model and is tailored specifically for the domain of VANET. Additionally, our model does not rely on introducing opinion piggybacking in message passing and the associated algorithms to generate and aggregate opinions at each individual node.

Golle et al. [7] present a technique that aims to address the problem of detecting and correcting malicious data in VANETs. The key assumption of their approach is in maintaining a model of VANET at every node. This model contains all the knowledge that a particular node has about the VANET. Incoming information can then be evaluated against the agent's model of VANET. If all the data received agrees with the model with a high probability then the agent accepts the validity of the data. However, in the case of receiving data which is inconsistent with the model, the agent relies on a heuristic that tries to restore consistency by finding the simplest explanation possible and also ranks various explanations. The data that is consistent with the highest ranking explanation(s) is then accepted by the node. The major strength of this approach is that it provides strong security against adversaries that might even be highly trusted members in the network or might be colluding together to spread malicious data. The approach that we present in this paper is orthogonal to their approach. In particular, we do not aim to detect and correct malicious data in the network, instead we want to detect the entities (agents or cars) that are generating this malicious data, establishing trust or distrust in the entity itself. This allows an agent to avoid an interaction with a distrustful agent in future.

A number of researchers have proposed trust and reputation models with role-based approach and the notion of confidence [19, 24, 14]. In particular, [9] introduced FIRE, a framework that integrates direct trust and role-based trust, in which the direct trust model of [23] is proposed as the method for capturing this element of the overall calculation, with some adjustment to consider more carefully the decay of trust values over time. In contrast, our model incorporates role-based trust and experience-based trust, which are combined using a priority-based approach, together with majority-based trust to aggregately evaluate the trustworthiness of agents while taking into consideration the important properties specific to VANET environments.

# 5. CONCLUSION AND FUTURE WORK

The question of placing trust in the data received from other agents in VANETs can potentially become a question of life and death. The success of deploying VANETs, therefore, is contingent upon the success in establishing effective methods of trust establishment [12]. In this work we started by discussing some of the key challenges to modeling the trust of agents in VANET environments followed by identifying the areas where the existing trust models in the domain of multi-agent systems are lacking in their applicability to VANETs. We then presented our expanded trust model for agents in VANETs. Our model is a novel integration of several trust metrics, including role-based trust, experience-based trust, priority-based trust, and majority-based trust. It is also important to note that our expanded trust is decentralized, location/time specific, event/task specific, able to cope with the data sparsity problem, cumulative in order to be scalable, sensitive to privacy concerns, and able to support system-level security. Experimental results demonstrate that our approach works effectively for the domain of VANETs.

For future work, we plan to explore various extensions to our current model. One interesting topic to explore is how to make use of a "commuter pool" – a set of agents that travel the same route with some regularity, as mentioned in Section 2.1.2. This would provide a social network where trust may be built up and frequent encounters may occur. This scenario would heighten the value of experience-based trust as part of the model.

Considering effective modeling of location information could also form an important thread for future research, due to its role in the calculation of majority-based opinion. For example, to avoid spoofing of location information, independent methods for vehicle tracking may need to be incorporated. We may also explore how to integrate incentives for drivers to opt into honest location reporting (e.g. as a precondition to receiving information from other vehicles).

To cope with various malicious attacks in general is another interesting topic of research. Collusion is notoriously difficult to address, but individual vehicles that are misreporting may possibly be detected due to differences with other vehicles, through our majority opinion algorithm. The case where agents fail to report events is also an interesting one to explore, for future research. If location tracking information becomes more prevalent, failure to report a life critical event at that location may be independent reason to decrease trustworthiness; vehicles in special roles (such as police) would likely serve to confirm the presence of such a life critical event. Current models of trust and reputation in multiagent systems have focused more on evaluating the trustworthiness of information that has been received, rather than considering the lack of reporting. Perhaps some new ground in trust modeling would be introduced by this research.

For future work we also plan to expand our experimental evaluation to include more complex scenarios where we test the effectiveness of other components including *event/task* and *location/time* specific components. Approaches such as that of [22] or of [21] may be particularly valuable to consider, as they propose methods to also be context-sensitive when modeling multidimensional trust. Furthermore, it is also important to measure scalability of our trust model with an increasing number of agents in the system. In fact, increasing the number of vehicles in our simulations may also provide additional insights into how best to set the value of $n$ in Step 1 of our algorithm. We could also experiment with different settings in our experimental evaluation, for instance allowing nodes to randomly lie about congestion on a road.

We could also consider a scenario where more than one agent (vehicle) in VANET forms a coalition with other agents to achieve a common goal. For instance, one such goal could be to cause mayhem in the network which can be attributed to vandalism or terrorism [12]. The consequences can be very critical and might end up claiming many lives. Future experimentation could also include cases where life critical events such as accidents are at play. In these cases, some kind of authority should be involved and this can serve to keep the other vehicles on the road honest in their reporting. A false report would differ with that of the authority. These experiments would therefore provide greated insights into the value of our concept of role-based trust.

As a final thread for future research, we may investigate the approaches of other authors who are also concerned with the issues of scalability and privacy that we are interested in addressing within our model, in order to determine new directions, for example, a position-based clustering technique for communication between agents as proposed in [28]; preserving the privacy of an agent through the use of proxies in peer-to-peer data sharing as in [13] which suggests that proxies may provide valuable masking of the identity of an agent, as long as they are trusted.

## 6. REFERENCES


[1] Wikipedia on road traffic safety. *http://en.wikipedia.org/wiki/Road-traffic_safety*.

[2] I. Chisalita and N. Shahmehri. On the design of safety communication systems for vehicles. *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans*, 37(6):933–945, 2007.

[3] D. R. Choffnes and F. E. Bustamante. An integrated mobility and traffic model for vehicular wireless networks. In *Proceedings of VANET*, 2005.

[4] F. Dotzer. Vars: A vehicle ad-hoc network reputation system. In *Proceedings of WoWMoM*, 2005.

[5] S. Eichler, C. Schroth, and J. Eberspacher. Car-to-car communication.

[6] GM. Threat assessment algorithm. *http://204.68.195.151/people/injury/research/pub/acas/acas-fieldtest/*.

[7] P. Golle, D. Greene, and J. Staddon. Detecting and correcting malicious data in vanets. In *Proceedings of VANET*, 2004.

[8] D. Huynh, N. Jennings, and N. Shadbolt. Developing an integrated trust and reputation model for open multi-agent systems. In *Proceedings of the Fifth International Conference on Autonomous Agents Workshop on Trust in Agent Societies*, 2004.

[9] T. Huynh, N. Jennings, and N. Shadbolt. An integrated trust and reputation model for open multi-agent systems. *Autonomous Agents and Multi-Agent Systems*, 13:119–154, 2006.

[10] A. Jøsang and R. Ismail. The beta reputation system. In *Proceedings of the 15th Bled Electronic Commerce Conference*, 2002.

[11] C. Leckie and R. Kotagiri. Policies for sharing distributed probabilistic beliefs. In *Proceedings of ACSC*, pages 285–290, 2003.

[12] R. Lin, S. Kraus, and Y. Shavitt. On the benefit of cheating by self-interested agents in vehicular networks. In *Proceedings of International Autonomous Agents and Multi Agent Systems (AAMAS)*, 2007.

[13] Y. Lu, W. Wang, B. Bhargava, and D. Xu. Trust-based privacy preservation for peer-to-peer data sharing. *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans*, 36(3):498–502, 2006.

[14] L. Mui, M. Mohtashemi, and A. Halberstadt. A computational model of trust and reputation. In *Proceedings of the 35th Hawaii International Conference on System Science (HICSS)*, 2002.

[15] R. Mukherjee, B. Banerjee, and S. Sen. Learning mutual trust. *Trust in Cyber-societies, Springer-Verlag*, pages 145–158, 2001.

[16] S. Rahman and U. Hengartner. Secure vehicle crash reporting. In *Proceedings of MOBICOMM*, 2007.

[17] S. D. Ramchurn, D. Huynh, and N. R. Jennings. Trust in multi-agent systems. *The Knowledge Engineering Review*, 19(1):1–25, 2004.

[18] M. Raya and J.-P. Hubaux. Securing vehicular ad hoc networks. *Journal of Computer Security*, 15:39–68, 2007.

[19] M. Raya, P. Papadimitratos, V. Gligor, and J. Hubaux. On data-centric trust establishment in ephemeral ad hoc networks. *Technical Report, LCA-REPORT-2007-003*, 2007.

[20] K. Regan, P. Poupart, and R. Cohen. Bayesian reputation modeling in e-marketplaces sensitive to subjectivity, deception and change. In *Proceedings of the Twenty-First Conference on Artificial Intelligence (AAAI)*, 2006.

[21] M. Rehak and M. Pechoucek. Trust modeling with context representation and generalized identities. In *Proceedings of the Eleventh International Workshop on Cooperative Information Agents (CIA 2007)*, 2007.

[22] A. Rettinger, M. Nickles, and V. Tresp. Learning initial trust among interacting agents. In *Proceedings of the Eleventh International Workshop on Cooperative Information Agents (CIA 2007)*, 2007.

[23] J. Sabater and C. Sierra. Regret: A reputation model for gregarious societies. In *Proceedings of the Fifth International Conference on Autonomous Agents Workshop on Deception, Fraud and Trust in Agent Societies*, pages 61–69, 2001.

[24] W. Teacy, J. Patel, N. R. Jennings, and M. Luck. Travos: Trust and reputation in the context of inaccurate information sources. *Auton Agent Multi-Agent Sys*, 12:183–198, 2006.

[25] T. C. to Car Communication Consortium (C2CC). *http://www.car-to-car.org/*.

[26] T. Tran. A reliability modelling based strategy to avoid infinite harm from dishonest sellers in electronic marketplaces. *Journal of Business and Technology (JBT), Special Issue on Business Agents and the Semantic Web*, 1(1):69–76, 2005.

[27] Y. Wang and J. Vassileva. Bayesian network-based trust model. In *Proceedings of the 6th International Workshop on Trust, Privacy, Deception and Fraud in Agent Systems*, 2003.

[28] Z. Wang, L. Liu, M. Zhou, and N. Ansari. A position-based clustering technique for ad hoc intervehicle communication. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 38(2):201–208, 2008.

[29] B. Yu and M. P. Singh. A social mechanism of reputation management in electronic communities. In *Proceedings of the 4th International Workshop on Cooperative Information Agents*, pages 154–165, 2000.

[30] J. Zhang and R. Cohen. Trusting advice from other buyers in e-marketplaces: The problem of unfair ratings. In *Proceedings of the Eighth International Conference on Electronic Commerce (ICEC'06)*, 2006.

# Space and Space-Time Organization Model for the Dynamic VRPTW


Besma Zeddini
University of Evry Val d'Essonne
IBISC Laboratory
523 place des Terrasses de l'Agora
91000 Evry, France
besma.zeddini@ens.univ-evry.fr

Mahdi Zargayouna
INRETS Institute
GRETIA Laboratory
2 rue de la Butte Verte
93160 Noisy le Grand Cedex
France
zargayouna@inrets.fr

Adnan Yassine
University of Le Havre
LMAH Laboratory
25 rue Philippe Lebon
76063 Le Havre Cedex
France
adnan.yassine@univ-lehavre.fr



## ABSTRACT

In this paper, we present a multiagent model for the Dynamic Vehicle Routing Problem with Time Windows. The system adapts insertion methods to a distributed configuration. The model has two declination: one spatial and one spatiotemporal. The two organization models that we propose rely on two different measures of what the insertion of the current customer would cost to a given vehicle. Our approach provides promising results and provides a new method to tackle the problem, in which the solving process is future-centered. The models developed in this paper offer two solutions with different advantages, which allow a decider to choose one of them following the operational configuration of her real problem. In the case where the transportation operator has a limited vehicles fleet, and where the mobilization of a new vehicle is costly, its system should be grounded on the spatiotemporal model, which mobilizes less vehicles. In contrast, if the costs in term of traveled distance are more critical, it is more interesting to ground its system on the spatial model.



## Keywords

Organization, Multiagent Systems, Routing.


## 1. INTRODUCTION

Several operational distribution problems, such as the deliveries of goods to stores, the routing of school buses, the distribution of newspapers and mail etc. are instantiations of NP-Hard theoretical problems called the Vehicle Routing Problems (VRP). In its original version, a VRP is a multi-vehicle Traveling Salesman Problem: there exists a certain number of nodes to be visited once by a limited number of vehicles. The objective is to find a set of vehicles' routes that minimizes the total distance traveled. Besides their practical usefulness, the VRP and its extensions are challenging optimization problems with an academic stimulating issues. One of the most widely studied variant of the problem is the time (and capacity) constrained version: the Vehicle Routing Problem with Time Windows (VRPTW henceforth), in which the requests to be handled are not simply nodes, but customers. For each customer, the following information are informed: the concerned node, two temporal bounds between which he desires to be visited and a quantity (number of goods to receive, number of persons to transport, etc.). Each vehicle has a limited capacity, which should not be exceeded by the quantities that it transports. The addition of time windows increases the complexity of the problem, since it narrows the space of valid solutions. The VRPTW can be formally stated as follows.

Let $G = (V, E)$ be a graph with node set $V = N \cup 0$ and edge set $E = (ij)|i \in V, j \in V, i \neq j$, $N = 1, 2..., n$ is the customer set with node 0 is the depot. With each node $i \in V$ is associated a customer demand $q_i(q_0 = 0)$, a service time $s_i(s_0 = 0)$, and a hard service-time window $[e_i, l_i]$ i.e. a vehicle must be at $i$ before $l_i$ but can be at $i$ before $e_i$ and must wait until the service starts. For every edge $(i, j) \in A$, a distance $d_{ij} \geq 0$ and a travel time $t_{ij} \geq 0$ are given. Moreover, the fleet of vehicles is homogeneous and every vehicle is initially located and end its route at a central depot. Each customer demand is assumed to be less than the vehicle capacity $Cap$. The objective is to find an optimal set of routes (with the minimal cost) such that:

1. All routes start and end at the depot;
2. each customer in $N$ is visited exactly once within its time window;
3. the total of customer demands for each route cannot exceed the vehicle capacity $Cap$.

The performance criteria are in general (following this order):

1. The number of vehicles used,
2. the total distance traveled,
3. the total waiting time.

Since the problem is NP-hard, exact approaches are only of theoretical interest, and heuristics are performed in order to find good solutions, not necessarily optimal, within reasonable computational times. The VRP and the VRPTW can be divided into two sets [18]: static problems and dynamic problems. The distinction between these two categories relies traditionally on the knowledge (static problem) or the ignorance (dynamic problem) before the start of the solving process of all the customers that have to be visited. The operational problems are rarely fully static and we can reasonably say that today a static system cannot meet the mobility needs of the users. Indeed, operational vehicle routing problems are rarely fully static. In operational settings, and even if the whole number of customers to be served is known, there is still some elements that makes the problem dynamic. These elements include breakdowns, delays, noshows, etc. It is thus always useful to consider a problem that is not fully static.

We rely on the multiagent paradigm for solving the dynamic VRPTW. An agent is a software system, that is situated in some environment and that is able to apply autonomous actions to satisfy its goals [27], and a MAS is a network of loosely coupled agents, which interact to solve problems that overpass the capacities or the knowledges of each one [25]. A multiagent modeling of the dynamic VRPTW is relevant for the following reasons. First, since it's a hard problem, choosing a design allowing for the distribution of computation can be a solution to propose short answer times to customers requests. Second, with the technological developments, it is reasonable to consider vehicles with onboard calculation capabilities. In this context, the problem is, *de facto*, distributed and necessitates an adapted modeling to take profit of the onboard equipments of the vehicles. Finally, the consideration of a multiagent point of view allows to envision new measures, new heuristics, not envisaged by centralized approaches.

In this paper, we propose a distributed version of the "insertion heuristics". Insertion heuristics is a method which consists in inserting the customers following their appearance order in the routes of the vehicles. The vehicle chosen to insert the considered customer is the one that has to make the minimal detour to visit him. Several multiagent works in the literature have been proposed to distribute insertion heuristics, but very few propose new measures of the insertion cost of a customer in the route of a vehicle, as an alternative to the traditional measure of its incurred detour. In the present work, we do propose two new measures, in the context of two new self-organization models. They are based on a space and on a space-time representation of the *Vehicle* agents' action zones. The objective is to allow the MAS to self-adapt exhibiting an equilibrated distribution of his *Vehicle* agents, and to decrease this way the number of vehicles mobilized to serve the customers.

The remainder of this paper is structured as follows. In section 2, we discuss previous proposals for the dynamic VRPTW w.r.t our approach. In the sections 3 and 4, we detail the two models and the use of new measures for the insertion decisions of the vehicles.We report on our experimental results in Section 6 and then Conclude with a few remarks.

## 2. RELATED WORK

As we said in the introduction, exact approaches cannot meet operational settings, and upon the relatively small set of benchmarking problems of [24] - 56 problems of 100 Euclidean customers[1] each, only 45 have a known optimal solution up until today [21]. However, interested readers of optimization approaches can refer to, e.g. [16] for a survey.

In fact, most of the proposed solution methods are heuristic or metaheuristic methods, provide good results in non-exponential times, and which have presented good results with benchmark problems. For instance, large-neighborhood local search [1, 22], iterative local search [15, 14], multi-start local search [19], simulated annealing [2], evolutive strategies [20, 11] and ant colonies [7]. These approaches present the best performances with static problems (where the set of transport requests is known *a priori*). For an extensive survey of the literature for the VRPTW approaches, the reader is invited to refer to, e.g. [10, 3].

Generally speaking, most of the works dealing with the dynamic VRPTW are more or less direct adaptations of static methods. For instance, the large-neighborhood local search is adapted to a dynamic context in [8]. In [13], the authors propose to adapt the genetic algorithms to deal with the dynamic VRPTW. The proposed algorithm starts by creating a population of initial solutions and tries continually to improve their quality. When a new customer reveals, he is inserted in all current solutions in the position minimizing the additional cost. Upon the static methods, insertion heuristics are the most widely adapted in a dynamic environment (e.g. [6, 12, 4]). Insertion heuristics are, in their original version, greedy algorithms, in the sense that the decision to insert a given customer in the route of a vehicle is irrevocable. They are also combined with meta-heuristics to improve the quality of the solutions. In [30], the authors propose an approach for the dynamic VRP, in which a central solver made of reactors manage the events coming up in the network. When a customer reveals, he is inserted in the route of a vehicle as for insertion heuristics. After each insertion, an optimization procedure is launched trying to reduce the number of used vehicles and the total traveled distance. The procedure is repeated until the current solution doesn't get better anymore. The customers are handled sequentially following a decreasing priority order, which is function of their respective distance and the decreasing order of their opening time windows.

The advantage of using insertion heuristics is that they are intuitive and fast. However, when they are applied in a dynamic context, their solving process is said to be myopic. Indeed, the system doesn't know which customers will appear once it has assigned the known customers to the vehicles. And even if we could have an optimal assignment and scheduling of the known customers, a new coming customer could make the old assignment sub-optimal, which would - in the worst case - necessitate a whole recomputation of all the routes.

Most of the multiagent approaches for the dynamic VRPTW are grounded, at least partially, on insertion heuristics. In [26] and in [17], the authors propose a multiagent architecture to solve a VRP and a multi-depot VRP for the first and a dial-a-ride problem for the second. The principle is the same: distribute an insertion heuristic, followed by a post-optimization step. In [26], the customers are handled sequentially, broadcasted to all the vehicles, which in turn propose insertion offers and the best proposal is retained by the customer. In the second step, the vehicles exchange customers to improve their solutions, each vehicle knowing the other agents of the system. Since vehicles are running in parallel, the authors envision to apply different heuristics for each vehicle, without changing the architecture. In-Time [17] is a system composed of *Customer* agents and *Vehicle* agents. The *Customer* agent announces himself and all the *Vehicle* agents calculate his insertion cost in their

[1]Euclidean customers have cartesian coordinates, and the distance and the le travel times between each pair of customers are calculated following the Euclidean metric.

routes. Again, the *Customer* agent selects the cheapest offer. The authors propose a distributed local search method to improve the solutions. Indeed, they allow a customer to ask stochastically to cancel his current assignment and to de reannounce himself to the system, with the objective of having a better deal with another vehicle. MARS [5] models a cooperative scheduling in a maritime shipping company in the form of a multiagent system. The solution to the global scheduling problem emerges from the local decisions. The system uses an extension of the Contract Net Protocol (CNP) [23] and shows that it can be used for having good initial solutions to complex problems of tasks assignment. The MAS profits from an *a priori* structuring of the agents, since each vehicle is associated with a particular society and can handle the only customers of this society.

From a protocol and an architecture point of view, our system sticks with the systems we have just described, since we propose a distributed version of insertion heuristics. However, in these proposals, none have focused on the redefinition of the insertion cost of a customer. The traditional insertion cost of a customer in the route of a vehicle is based on the incurred detour of the vehicle. We propose a new insertion cost measure, focused on the space-time coverage of the vehicles, which aims at counterbalancing the myopia of the traditional measures, by privileging an insertion process that is future-centered.

## 3. SPATIAL MODEL

The optimization of the conventional criteria of the VRPTW (number of vehicles and total distance) leads to the appearance of uncovered areas because of their low density. In fact, the fact that we deal with a dynamic and nondeterministic problem can lead to the appearance of two different but non independent phenomena. The first is the concentration of vehicles in some zones which are more attractive and may lead to the second phenomena, which is the lack of service elsewhere. The idea behind our self-organization models is that when the positioning of vehicles is made such as to cover as much territory as possible, the risk of customers whose demand is unsatisfied, and the obligation to mobilize new vehicles to serve them, decreases. The choices we make to solve this problem is to use the multiagent paradigm coupled with the insertion heuristics. In this context, we have only one lever to change the system's behavior, which is the way in which the *Vehicle* agents calculate the insertion cost of a customer. These calculation methods are two dimensional: spatial and spatiotemporal. The two self-organization models that we propose have the objective of minimizing the number of used vehicles, while keeping the use of a "pure" insertion heuristics, i.e. without any further improvements or post-optimization.

Our systems are composed of a dynamic set of agents which interact to solve the dynamic VRPTW. A solution consists of a series of vehicles routes, each route consists of a sequence of customers with their associated visit time. We define two categories of agents. *Customer* agents, which represent users of the system (persons or goods) and *Vehicle* agents. We assume that there is an access point to the system (Web server, GUI, simulator, etc.) which verifies the correctness of customers requests (existing node, valid time windows, etc.) before to create the corresponding *Customer* agents. Once created, a *Customer* agent announces itself to all the *Vehicle* agents of the MAS. Each *Vehicle* agent sends an offer to the *Customer* agent with a corresponding insertion cost. The *Customer* agent chooses the *Vehicle* agent with the lower cost. Finally, the chosen *Vehicle* agent inserts the customer in its route.

Following the description above, the *Customer* agent chooses between several *Vehicle* agents the one with the minimal proposed insertion cost. The systems that are based on this heuristic use generally the measure of Solomon [24] as an insertion cost. This measure consists in inserting the customer which has the minimal impact on the general cost of the vehicle (which is generally function of the vehicle's incurred detour). This measure is simple and the most intuitive but has a serious drawback, since inserting the current customer might make lots of future customers' insertions infeasible, with the current number of vehicles. Its problem is that it generates vehicles' plans that are very constrained in time and space, i.e. plans that offer a few possibilities of insertion between each pair of adjacent planned customers. As a consequence, the appearance of new customers might oblige the system to create new vehicles to serve them. Through the modeling of *Vehicle* agents' action zones, we propose a new way to compute the customer's insertion cost in the route of a vehicle, and a new choice criterion between vehicles for the insertion of a given customer. We propose a new method that allows the system to choose the *Vehicle* agent "which decrease in the probability to participate in future insertions is minimal", to serve the new customer. The logic of our models is different from the traditional models, which focus on the increase of the traveled distance, neglecting the impact of the current insertion decision on future insertion possibilities.

The objective of the spatial self-organization model is to allow the specialization of the system's vehicle to zones while maintaining an optimal coverage of the network (cf.Figure 1). Thus, we define action zones on the transportation network, to which the vehicles are attached. The attachment of vehicles to their zones is not encoded in the vehicle behavior, but it has an effect on how they calculate their customers insertion costs. This computation should ensure that a *Vehicle* agent plans its route so that it's incentive to stay in its zone. The definition of geographical zones of vehicles is treated as a partitioning graph problem and is left out of the scope of this paper. We suppose that the definition of these zones is a system parameter, which is the responsibility of an expert. Each zone is defined by a set of nodes and a barycentre.

Definition 1. *Spatial Action Zone*
*Let $G = (N, A)$ be a graph with a set of nodes $N = \{(n_i)\}, i = \{0, \ldots, m\}$ (node $n_0$ is the depot) and a set of arcs $A = \{(n_i, n_j) | n_i \in N, n_j \in N, n_i \neq n_j\}$. Let the costs matrix $C = \{(C_{ij})\}$ of size $m \times m$ (the arc $(n_i, n_j)$ has a distance of $C_{ij}$). We define the zone $\zeta = (N_\zeta, A_\zeta)$ as a subgraph of $G$.*

Definition 2. *barycentre of a Zone*
*The barycentre of zone $\zeta$ is a node $n^{*\zeta}$ of $N_\zeta$ that minimize $\sum_{y \in N_\zeta} d_{n^{*\zeta}, y}$.*

Each zone is defined by a barycentre and a set of nodes (cf.Figure 2). The barycentre of a zone corresponds to the node which is the closest to all other nodes in the zone. At any point in time, each *Vehicle* agent has a distance from its action zone. This distance depends the customers inserted into its route. It is computed such as to include a penalty $\beta$

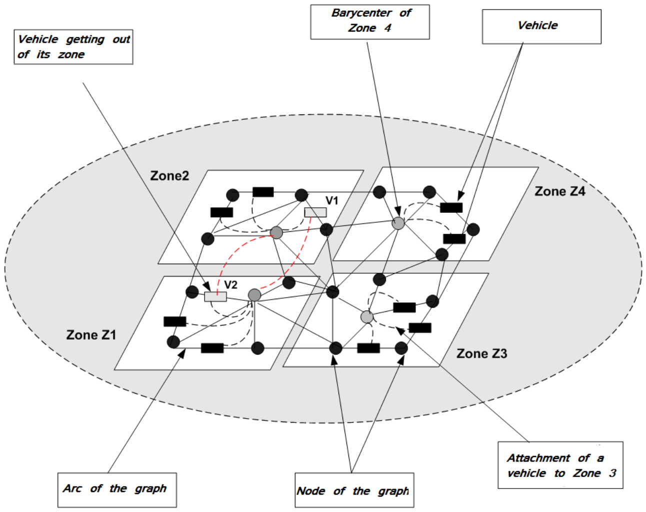


**Figure 1: Specialization and Attraction Zones**

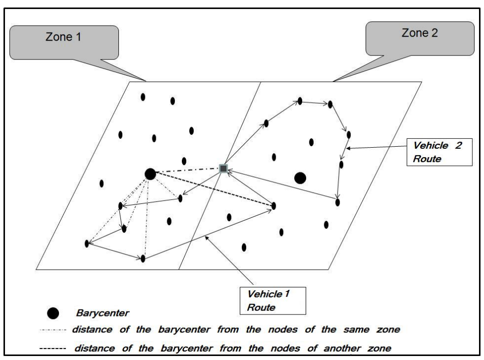


**Figure 2: Spatial Action Zones**

to the *Vehicle* agent distance if it integrates nodes outside its zone. Indeed, if the node is inside the vehicle's zone, its distance from the barycentre of the action zone remains unchanged. Otherwise, its distance is multiplied by a factor $\beta$ which is a system parameter.

DEFINITION 3. *Vehicle Distance from its Zone*
*The distance of a vehicle $v$ from its zone $\zeta_v$ at a given moment is equal to the average distance of the nodes in its route from the barycentre of $\zeta_v$:*

$$d_{v,\zeta_v} = \frac{\sum_{n^v \in Nodes(v)} d_{n^v,n^*\zeta}}{card(Nodes(v))}$$

*with*

$$\forall c \in N, d_{n^v,c} = \begin{cases} d_{n^v,c} & \text{if } n^v \in z_v \\ \beta \times d_{n^v,c} & \text{else} \end{cases}$$

*$Nodes(v)$ represents the nodes of the* Vehicle *agent's route and $card(Nodes(v))$ is the number of nodes in $Nodes(v)$. Finally, $\beta$ is the penalty imposed to the vehicle distance, if its route integrates nodes which are outside $v$'s zone.*

The offer that a *Vehicle* agent proposes to a customer for its insertion is equal to the old distance of the vehicle from its zone minus its new one, if it had to insert the customer. The bigger $\beta$ is, the more the vehicles are organized so that they stay in their zones. The definition of geographical zones of vehicles is treated as a partitioning graph problem and is left out of the scope of this paper. We suppose that the definition of these zones is a system parameter, which is the responsibility of an expert.

# 4. SPATIOTEMPORAL MODEL

Even if it allows a better spatial coverage of the network, the spatial self-organization model has two major drawbacks. First, it assumes *a priori* geographical segmentation. With the absence of data on previous customers demands, this task requires a great calibration effort to specify the most efficient zones' segmentation. Second, it doesn't incorporate the temporal dimension of the problem, since a vehicle might not be able to serve a customer even if it is located in its zone, because of the time constraints. In the following, we propose to integrate the temporal dimension in the *Vehicle* agents' action zones and to eliminate any *a priori* definition of these zones.

In the heuristics and multiagent methods of the literature, the hierarchical objective of minimizing the number of mobilized vehicles is considered in priority w.r.t the distance traveled by all the vehicles. The vast majority of the literature heuristics are as a consequence based on a two-phase approach: the minimization of the number of vehicles, followed by the minimization of the traveled distance [21]. The model that we propose in this section has the objective of minimizing the number of used vehicles, while keeping the use of a "pure" insertion heuristics, i.e. without any further improvements.

To this end, our model allows *Vehicle* agents to cover a maximal space-time zone of the transportation network, avoiding this way the mobilization of a new vehicle if a new customer appears in an uncovered zone [28]. A space-time pair $\langle i, t \rangle$ - with $i$ a node and $t$ a time - is said to be "covered" by a *Vehicle* agent $v$ if $v$ can be in $i$ at $t$. In the context of the dynamic VRPTW, maximizing the space-time coverage of *Vehicle* agents results in giving the maximum chance to satisfy the demand of a future (unknown) customer. The logic of this measure is different from the traditional measures', which focus on the increase of the traveled distance, neglecting the impact of the current insertion decision on future insertion possibilities.

Following the description of the previous section, the *Dispatcher* agent chooses between several *Vehicle* agents the one with the minimal proposed insertion cost. The systems that are based on this heuristic use generally the measure of Solomon [24] as an insertion cost. This measure consists in inserting the customer which has the minimal impact on the general cost of the vehicle (which is generally function of the vehicle's incurred detour). This measure is simple and the most intuitive but has a serious drawback, since the insertion of the current customer might result in making the insertion of a great number of future customers infeasible, with the current number of vehicles. Its problem is that it generates vehicles' plans that are very constrained in time and space, i.e. plans that offer a few possibilities of insertion between each pair of adjacent planned customers. As a consequence, the appearance of new customers risks to oblige the system to create a new vehicle to serve them. Through the modeling of *Vehicle* agents' Action Zones, we propose a new way to compute the customer's insertion cost in the route of a vehicle, and a new choice criterion between vehicles for the insertion of a given customer. We propose a computation which objective is to choose, provided a newcomer customer, the *Vehicle* agent "which decrease in the probability to participate in future insertions is minimal". We use that variation of *Vehicle* agents' Action Zone as an insertion cost for the insertion of a given customer in its route.

## 4.1 Environment Modeling

The space-time Action Zone of a *Vehicle* agent is composed of a subset of the network nodes, together with the times that are associated to them. We model the MAS environment in the form of a space-time network, inferred from the network graph. Each node of the graph becomes a pair $\langle space, time \rangle$, which represents the "state" of the node in a discrete time period. The space-time network is composed of several subgraphs, where each subgraph is a copy of the static graph, and corresponds to the state of the graph in a certain period of time (cf. Figure 3). We index the nodes of a subgraph as follows: $\langle 0, t \rangle, \ldots, \langle N, t \rangle$, with $t \in \{1, ..., h\}$, with $0, \ldots, N$ are the nodes of the network and $h$ the number of considered discrete periods. The total number of nodes in the space-time network is equal to $h \times N$. The edges linking the nodes of a subgraph are those of the static graph, and the costs are the travel times as described in the introduction ($t_{ij}$).

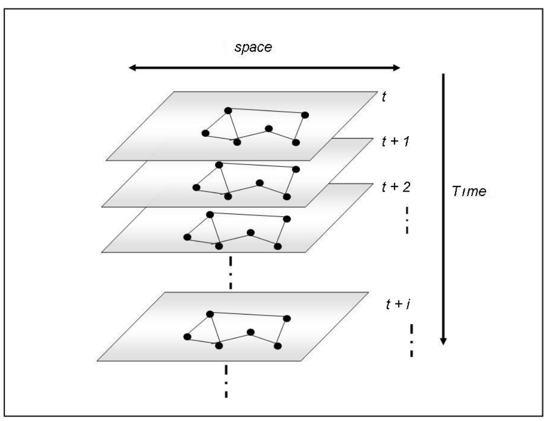


**Figure 3: Space-Time Network**

## 4.2 Intuition of the Action Zones

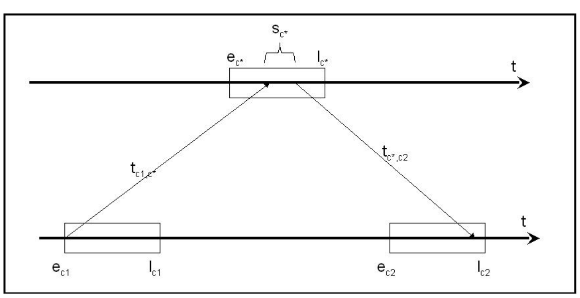


**Figure 4: Feasible insertion**

Consider a *Vehicle* agent $v$ that has an empty route. In order for this agent to be able to insert a new customer $c$ - described by: $n$ a node, $[e, l]$ a time window, $s$ a service time, and $q$ a quantity - $l$ has to be big enough to allow $v$ to be in $n$ without violating his time constraints. More precisely, the current time $t$, plus the travel time between the depot and $n$ has to be less or equal to $l$ (cf. Figure 4). Starting from this observation, we define the Action Zone of a *Vehicle* agent as the potential customers that satisfy this constraint. To do so, we define the Action Zone of a *Vehicle* agent as the set of pairs $\langle n, t \rangle$ of the space-time network that remain valid given his current route ($n$ can be visited by the vehicle at $t$). The Action Zone of a *Vehicle* agent with an empty route is illustrated by the triangular shadow in the Figure 5 (it is actually a conic shadow in a three-dimensional space).

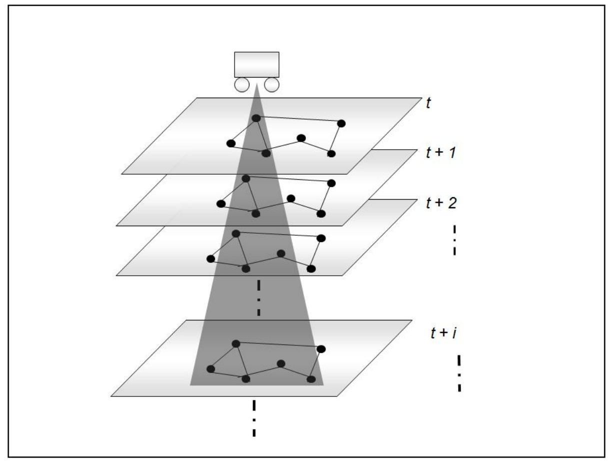


**Figure 5: Initial Space-Time Action Zone**

When a *Vehicle* agent inserts a customer in his route, his Action Zone is recomputed, since some $\langle node, time \rangle$ pairs become not valid because of his insertion. In the Figure 6, a new customer is inserted in the route of the vehicle. The Action Zone of the *Vehicle* agent after inserting the customer is represented by the interior of the contour of the bold lines, which represent the space-time nodes which remain accessible after the insertion of the customer (the computation of the new Action Zone is explained later).

The associated cost to an offer from a *Vehicle* agent $v$ for the insertion of a *Customer* agent $c$ corresponds to the hypothetical decrease of the Action Zone of $v$ following the insertion of $c$ in his route.

The idea is that the chosen *Vehicle* for the insertion of a customer is the one that looses the minimal chance to be candidate for the insertion of future customers. Thus, the criterion that is maximized by the society of *Vehicle* agents is the sum of their Action Zones, i.e. the capacity that the MAS has to react to the appearance of *Customer* agents, without mobilizing new vehicles.

To illustrate the Action Zones and their dynamics, we present the version of the measure that is related to an Euclidean problem, i.e. where travel times are computed following the Euclidean metric. The following paragraphs detail the measure as well as its dynamics.

## 4.3 The Computation of Action Zones

In the Euclidean case, the transportation network is a

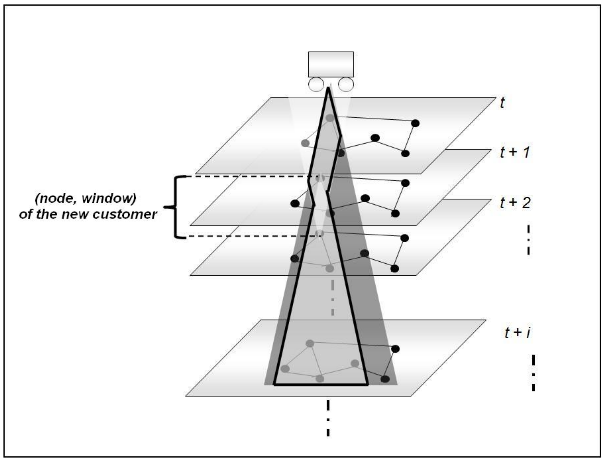


**Figure 6: Action Zone after the Insertion of a Customer**

plane, and the travel times between two points $i$ (described by $(x_i, y_i)$) and $j$ (described by $(x_j, y_j)$) is equal to

$$\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

Therefore, if a vehicle is in $i$ at the moment $t$, he cannot be in $j$ earlier than $t_i + \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$.

We can compute at any time, from the current position of a vehicle, the set of triples $(x, y, t)$ where he can be in the future. Indeed, considering a plane with an X-axis in $[x_{min}, x_{max}]$ and a Y-axis in $[y_{min}, y_{max}]$, the set of space-time positions is the set of points in the cube delimited by $[x_{min}, x_{max}]$,$[y_{min}, y_{max}]$ and $[e_0, l_0]$ (recall that $e_0$ and $l_0$ are the scheduling horizon and are the minimal and maximal values for the time windows). Consider a vehicle in the depot $(x_0, y_0)$ at $t_0$. The set of points $(x, y, t)$ that are accessible by this vehicle are described by the following inequality:

$$\sqrt{(x - x_0)^2 + (y - y_0)^2} \leq (t - t_0)$$

The $(x, y, t)$ satisfying this inequality are those that are positioned inside the cone $\mathcal{C}$ of vertex $(x_0, y_0, t_0)$ and with the equation $\sqrt{(x - x_0)^2 + (y - y_0)^2} = (t - t_0)$ (c.f Figure 7). This cone represents the Action Zone of a *Vehicle* agent, with an empty route, in the Euclidean case. It represents all the possible space-time positions that this *Vehicle* agent is able to have in the future.

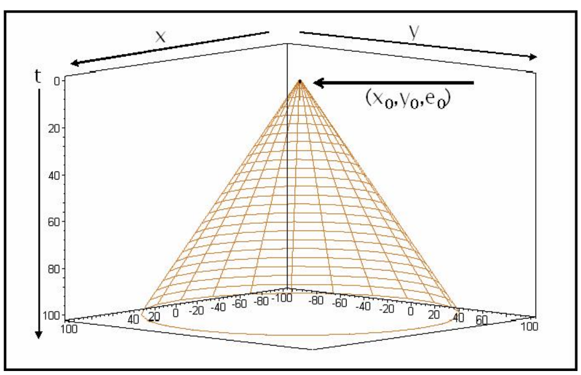


**Figure 7: Initial Action Zone**

We use the Action Zone of the *Vehicle* agents when a *Customer* agent has to choose between several *Vehicle* agents for his insertion. We have to be able to compare the Action Zones of different *Vehicle* agents. To do so, we propose to quantify it, by computing the volume of the cone $\mathcal{C}$ representing the future possible positions of the vehicle:

$$Volume(\mathcal{C}) = \frac{1}{3} \times \pi \times (l_0 - e_0)^3$$

This is the quantification of the initial Action Zone of any new *Vehicle* agent joining the MAS. When a new *Customer* agent appears, a *Vehicle* agent computes his new Action Zone, the cost that he proposes to the *Dispatcher* agent is the difference between his old Action Zone and his new one. The new Action Zone computation is detailed in the following paragraph.

## 4.4 Dynamics of the Action Zones

Consider a customer $c_2$ (of coordinates $(x_2, y_2)$ and with a time window $[e_2, l_2]$) that joins the system, and suppose that $v$ is temporarily the only available *Vehicle* agent of the system and has an empty route. The agent $v$ has to deduce his new space-time action zone, i.e. the space-time nodes that he can still reach without violating the time constraints of $c_2$. The new action zone answers the following questions: "if $v$ had to be in $(x_2, y_2)$ at $l_2$, where would he have been before? And if he had to be there at $e_2$ where would he be after $e_2 + s_2$ ?". The triples $(x, y, t)$ where the *Vehicle* agent can be before visiting $c_2$ are described by the inequality $[a]$, and the triples $(x, y, t)$ where he can be after visiting $c_2$ are describe by the inequality $[b]$.

$$\sqrt{(x - x_2)^2 + (y - y_2)^2} \leq (l_2 - (t)) \qquad [a]$$

$$\sqrt{(x - x_2)^2 + (y - y_2)^2} \leq (t - (e_2 + s_2)) \qquad [b]$$

The new Action Zone is illustrated by the Figure 8: the new measure consists in the intersection of the initial cone $\mathcal{C}$ with the union of the two new cones described by the inequalities $[a]$ and $[b]$ (denoted respectively by $\mathcal{C}_1$ and $\mathcal{C}_2$). The new measure of the Action Zone is equal to the volume of the intersection of $\mathcal{C}$ with the union of $\mathcal{C}_1$ and $\mathcal{C}_2$. The complete computation of the volume of the intersection of these two cones is reported in the Appendix A of [29].

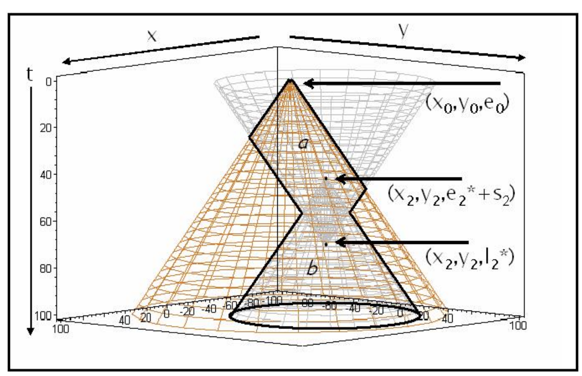


**Figure 8: Space-Time Action Zone after the insertion of $c_2$**

The cost of the insertion of a customer in the route of a vehicle is equal to the measure associated with the old Action Zone of the vehicle minus the measure of the new Action Zone, after the insertion of the customer. The quantity measured represents the space-time positions that the vehicle cannot have anymore, if he had to insert this customer in his route. The retained *Vehicle* agent to visit a given customer is the one for which the insertion of the customer causes less loss in his space-time Action Zone. This corresponds to choosing the vehicle that looses the minimal possibilities to be candidate for future customers.

### 4.5 Coordination of Action Zones

The objective of the self-organization model is to allow a better space and space-time coverage of the transportation network. This improvement is materialized by a minimal mobilization of vehicles in front of the appearance of new customers. With the mechanism described until now, every Vehicle agent tries to maximize its own action zone independently from the other agents of the MAS. However, it would be more interesting to incite the agents society in its whole to cover the network in the most efficient way. More precisely, the fact that a vehicle loses space-time nodes that it is the only one to cover should be more costly than to lose nodes that are covered by other agents.

To this end, to every node of the space-time network, we start by associating the list of vehicles covering it. Then, to every creation of a new vehicle agent, the set of space-time nodes that are part of its action zone is computed. The vehicle proceeds then with the notification of these nodes that they are part of its action zone. ces nœuds qu'ils font partie de sa *zone d'action*. Every node updates its list of vehicles that are covering it at each notification from a Vehicle agent. Similarly, when the action zone of a Vehicle agent loses a node, the node is notified and its vehicles list updated.

Now, when the insertion cost of a customer is computed, avery Vehicle agent starts by calculating the space-time nodes that it would lose if it happens to insert the new customer. Then, it interrogates each of these nodes about the "price to pay" if it happens to not cover them anymore. This price is inversely proportional to the number of vehicles covering this node. More precisely, the price to pay is equal to

$$\frac{1}{card(v_{\langle n,t\rangle})}$$

with $v_{\langle n,t\rangle}$ denoting the Vehicle agents covering the space-time node $\langle n,t\rangle$.

This way, the space-time network being the only entity knowing the action zones of all the Vehicle agents (thanks to the lists of vehicles associated with the nodes), it associates more or less penalty to the decisions of non-coverage of the network by the vehicles as time progresses. Thus, the Vehicle agents are incited to cover the whole network in a coordinated way, improving by doing so the reactivity of the MAS.

## 5. SIMULATION TOOL

In this section, we briefly introduce the tool that we propose for the scenarios simulation of the dynamic VRPTW. Except for dedicated projects and commercial applications, the systems proposing a platform for the simulation of vehicle routing systems are rare. We choose to develop such an application for several reasons. First, this allows us to have a pragmatic vision of the execution environment of our proposals. Then, such an application inists on the finality of our proposals, which is to develop a decision support system for transport operators. As will be illustrated hereafter, the operator is offered an interface with the state of its fleet and the ongoing customers. These indicators allow her to perform some afjustments when needed. Eventually, the operator will have the possibility to choose between the three models that we propose the most suitable one, provided her operational settings. Finally, a Web application is also proposed for the customers, to demonstrate the deployment scenario that we envision for our system, from the customer's viewpoint. Here follow some screenshots of the simulation tool.

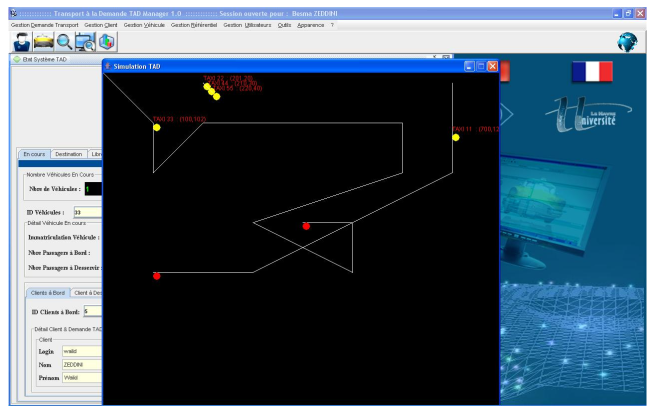

**Figure 9: Vehicle Plans**

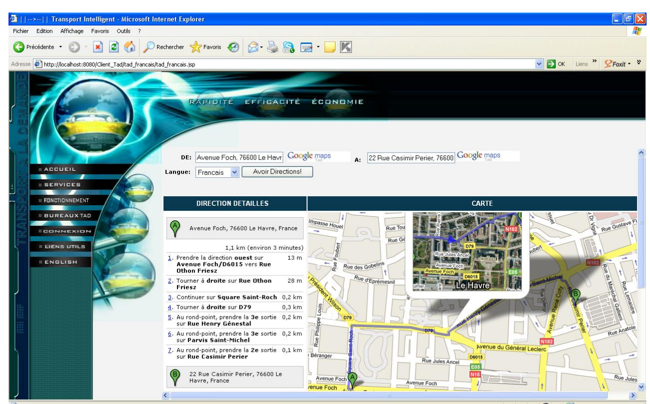


**Figure 10: Customers' Itinerary**

## 6. RESULTS

Marius M. Solomon [24] has created a set of different static problems for the VRPTW. It is now admitted that these problems are challenging and diverse enough to compare with enough confidence the different proposed methods. A proof for that claim is that there is no unique heuristic that provides the best results for each one of these problems at the same time. In Solomon's benchmarks, six different

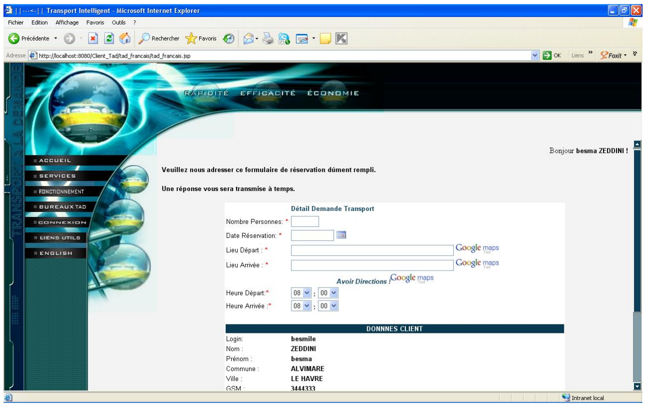


**Figure 11: Booking Form**

| | Δ Distance | | Δ Space-Time | | Δ Space | |
|---|---|---|---|---|---|---|
| Problem | \|Fleet\| | Dist | \|Fleet\| | Dist | \|Fleet\| | Dist |
| R1 (25 c) | 64 | 6372 | 53 | 6561 | 58 | 5732 |
| C1 (25 c) | 34 | 3167 | 31 | 3152 | 32 | 3014 |
| R1 (50 c) | 107 | 12036 | 92 | 12089 | 101 | 11307 |
| C1 (50 c) | 60 | 6712 | 53 | 7093 | 58 | 6682 |
| R1 (100 c) | 181 | 17907 | 150 | 17348 | 164 | 16680 |
| C1 (100 c) | 121 | 16011 | 108 | 16512 | 113 | 15206 |

**Table 1: Results summary**

sets of problems have been defined: C1, C2, R1, R2, RC1 and RC2. The customers are geographically uniformly distributed in the problems of type R, clustered in the problems of type C, and a mix of customers uniformly distributed and clustered is used in the problems of type RC. The problems of type 1 have narrow time windows (very few customers can coexist in the same vehicle's route) and the problems of type 2 have wide time windows. Finally, a constant service time is associated with each customer, which is equal to 10 in the problems of type R and RC, and to 90 in the problems of type C. We choose to use Solomon benchmarks, while following the modification proposed by [9] to make the problem dynamic. We have implemented three MAS with almost the same behavior, the only difference concerns the measure used by *Vehicle* agents to compute the insertion cost of a customer. For the first implemented MAS, it relies on the Solomon measure (noted Δ Distance), on the space-time model for the second (noted Δ Space-Time). We choose to run our experiments with the problems of class R and C, of type 1, which are the instances that are very constrained in time (narrow time windows).

Table 1 summarizes the results where we consider successively 25, 50 and 100 customers. The results show, with the two classes of problems, that the use of the space-time model mobilizes less vehicles than the spatial model, which in turn behaves better than the traditional measure, whatever the number of considered customers. These results validate the intuition of the models that consists of maximizing the future insertion possibilities for a *Vehicle* agent. Once this result validated, it is interesting to check the results with respect to the total distance traveled by all the vehicles. With respect to this criterion, the space model behaves better than the two others, while the behavior of the space-time model is less efficient, since it gives better results for the problems C1 with 25 customers and R1 with 100 customers, but is dominated by the traditional measure for the others. The fact remains that our results for both models provide better results than the traditional heuristic, provided the primary objective of the problem, which is to minimize the number of vehicles mobilized by the system.

## 7. CONCLUSION

In this paper, we have proposed two agent-oriented self-organization models for the dynamic VRPTW based on the agents' action zones. The action zones of the *Vehicle* agents reflect their coverage of the transportation network. We use the variation of these action zones as a new metric between *Vehicle* agent to reduce the myopic behavior of traditional metrics. By optimizing the coverage of the environment by the *Vehicle* agents, our model allows the MAS to self-adapt by exhibiting an equilibrated distribution of the vehicles, and to lessen this way the number of vehicles mobilized to serve the customers.

The models developed in this paper offer two solutions with different advantages, which allow a decider to choose the model to use following the operational configuration of the real problem faced. In the case where the transportation operator has a limited vehicles fleet, and where the mobilization of a new vehicle is costly, it is undeniable that the system should be grounded on the space-time model, which mobilizes less vehicles. In contrast, in certain real problems, the operator has a virtually unlimited fleet of vehicles, and the costs in term of traveled distance are more critical. Indeed, certains systems relie on a fleet of vehicles, which besides offer a traditional individual taxi service. In this kind of systems, it is more interesting to ground it system on the spatial model.

Our current work is oriented towards taking into account historic data of customers requests on the network nodes. We use these data as a weighting of the action zones of the *Vehicle* agents that concern the nodes frequently requested, and this to make them converge towards high density zones in the right time. Besides, the assessment the impact of breakdowns, noshows and other dynamic changes in the environment, on the solving process is also an ongoing research.

## 8. REFERENCES


[1] R. Bent and P. Hentenryck. A two-stage hybrid local search for the vehicle routing problem with time windows. *Transportation Science*, 38(4):515Ű530, 2004.

[2] Z. J. Czech and P. Czarnas. A parallel simulated annealing for the vehicle routing problem with time windows. In *Proceedings of the 10th Euromicro Workshop on Parallel, Distributed and Network-based Processing*, pages 376–Ű383, Canary Islands (Spain), 2002.

[3] G. Desaulniers, J. Desrosiers, M. Solomon, F. Soumis, and J.-F. Cordeau. The VRP with time windows. In D. Vigo and P. Toth, editors, *The Vehicle Routing Problem*, SIAM Monographs on Discrete Mathematics and Applications, pages 157–193. SIAM, 2002.

[4] M. Diana. The importance of information flows temporal attributes for the efficient scheduling of dynamic demand responsive transport services. *Journal of advanced Transportation*, 40(1):23–46, 2006.

[5] K. Fischer, J. Muller, M. Pischel, and D. Schier. A model for cooperative transportation scheduling. In V. R. Lesser and L. Gasser, editors, *Proceedings of the First International Conference on Multiagent Systems (ICMAS'95)*, pages 109–116, Menlo park, CA (USA), 1995. AAAI Press / MIT Press.

[6] L. Fu and S. Teply. On-line and off-line routing and scheduling of dial-a-ride paratransit vehicles. In *Computer-Aided Civil and Infrastructure Engineering*, volume 14, pages 309–319. Blackwell Publishers, Oxford (UK), 1999.

[7] L. M. Gambardella, E. D. Taillard, and G. Agazzi. MACS-VRPTW: A multiple ant colony system for vehicle routing problems with time windows. In M. D. D. Corne and F. Glover, editors, *New Ideas in Optimization*, pages 63–76, McGraw-Hill (London), 1999.

[8] M. Gendreau, F. Guertin, J.-Y. Potvin, and R. Séguin. Neighborhood search heuristics for a dynamic vehicle dispatching problem with pick-ups and deliveries. *Transportation Research Part C*, 14:157–174, 2006.

[9] M. Gendreau, F. Guertin, J.-Y. Potvin, and E. D. Taillard. Parallel tabu search for real-time vehicle routing and dispatching. *Transportation Science*, 33(4):381–390, 1999.

[10] B. Golden, S. Raghavan, and E. Wasil. *The vehicle routing problem, latest advances and new challenges*, volume 43 of *Operations research/computer science interfaces*. Springer Verlag, 2008.

[11] J. Homberger and H. Gehring. Two-phase hybrid metaheuristic for the vehicle routing problem with time windows. *European Journal of Operational Research*, 162:220Ű238, 2005.

[12] M. E. Horn. Fleet scheduling and dispatching for demand-responsive passenger services. *Transportation Research C*, 10(1):35–63, 2002.

[13] H. Housroum, T. Hsu, R. Dupas, and G. Goncalves. A hybrid ga approach for solving the dynamic vehicle routing problem with time windows. In I. C. Society, editor, *Proceedings of the IEEE Conference on Information and Communication Technologies: from Theory to Applications*, pages 3347–3352, Damascus, Syria, 2006.

[14] T. Ibaraki, S. Imahori, K. Nonobe, K. Sobue, T. Uno, and M. Yagiura. An iterated local search algorithm for the vehicle routing problem with convex time penalty functions. *Discrete Applied Mathematics*, 156(11):2050Ű2069, 2008.

[15] T. Ibaraki, M. Kubo, T. Masuda, T. Uno, and M. Yagiura. Effective local search algorithms for the vehicle routing problem with general time window constraints. *Transportation Science*, 39(2):206Ű232, 2005.

[16] M. Jepsen, B. Petersen, S. Spoorendonk, and D. Pisinger. Subset-row inequalities applied to the vehicle-routing problem with time windows. *Operations Research*, 56(2):497–511, 2008.

[17] R. Kohout and K. Erol. In-Time agent-based vehicle routing with a stochastic improvement heuristic. In *Proceedings of the sixteenth national conference on Artificial intelligence and the eleventh Innovative applications of artificial intelligence (AAAI'99/IAAI'99)*, pages 864–869, Menlo Park, CA (USA), 1999. AAAI Press.

[18] A. Larsen. *The Dynamic Vehicle Routing Problem.* PhD thesis, University of Denmark, 2000.

[19] A. Lim and X. Zhang. A two-stage heuristic with ejection pools and generalized ejection chains for the vehicle routing problem with time windows. *INFORMS Journal on Computing*, 19(3):443Ű57, 2007.

[20] D. Mester and O. Bräysy. Active guided evolution strategies for large scale vehicle routing problems with time windows. *Computers & Operations Research*, 32(6):1593Ű1614, 2005.

[21] Y. Nagata, O. Bräysy, and W. Dullaert. A penalty-based edge assembly memetic algorithm for the vehicle routing problem with time windows. *Computers & Operations Research*, xx(x), July 2009.

[22] D. Pisinger and S. Ropke. A general heuristic for vehicle routing problems. *Computers & Operations Research*, 34(8):2403Ű2435, 2007.

[23] R. G. Smith. The contract net protocol: High-level communication and control in a distributed problem solver. *IEEE Trans. on Comp.*, C-29(12):1104–1113, December 1980.

[24] M. Solomon. Algorithms for the vehicle routing and scheduling with time window constraints. *Operations Research*, 15:254–265, 1987.

[25] K. P. Sycara. Multiagent Systems. *AI Magazine*, 19(2):79–92, 1998.

[26] S. R. Thangiah, O. Shmygelska, and W. Mennell. An agent architecture for vehicle routing problems. In *Proceedings of the 2001 ACM symposium on Applied computing (SAC '01)*, pages 517–521, New York, NY (USA), 2001. ACM Press.

[27] M. Wooldridge and N. R. Jennings. Intelligent agents: Theory and practice. *Knowledge Engineering Review*, 10(2):115–152, 1995.

[28] M. Zargayouna. Une représentation spatio-temporelle de l'environnement pour le transport à la demande. In *Atelier: Reprśentation et raisonnement sur le temps et l'espace, Plate-forme AFIA 2005*, Nice (France), 2005. in french.

[29] M. Zargayouna. *Modèle et langage de coordination pour les systèmes multi-agents ouverts. Application au problème du transport à la demande*. Phd Thesis, University of Paris-Dauphine, Paris (France), 2007. In french.

[30] K. Q. Zhu and K.-L. Ong. A reactive method for real time dynamic vehicle routing problem. In *12th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'00)*, pages 176–179, Vancouver (Canada), 2000. IEEE Computer Society.

# Traffic Simulations with Emotional Effects: A Proposal for Core Affect Contagion

Mario Paolucci
Laboratory of Agent Based Social Simulation, ISTC/CNR
Via Palestro 32, 00185 Rome, Italy
mario.paolucci@istc.cnr.it

Hiromitsu Hattori
Graduate School of Informatics, Kyoto University
Yoshida-Honmachi, Sakyo-ku, Kyoto, 606-8501, Japan
hatto@i.kyoto-u.ac.jp

## ABSTRACT

In this position paper, we introduce the idea of using emotions as a mediator between high-level planning and reactive behavior for multiagent-based simulation of driving. After reviewing motivations and arguing for the importance of our approach, we focus on a simple case, namely the contagion of core affects. We propose to extend an existing architecture, allowing for selection of driving styles depending on the current core affect. In turn, core affect is influenced by the driver's perception of others driving styles. We conclude by sketching a preliminary model that could be used to describe the result of reciprocal affect influencing in the traffic context.

## Categories and Subject Descriptors

I.2.11 [**Distributed Artificial Intelligence**]: Multiagent systems

## General Terms

Design

## Keywords

multiagent simulation, emotion, traffic simulation, human behavior modeling

## 1. INTRODUCTION

Multiagent-based simulation (MABS) can be used as a technique, alternative to equation based models [6], to recreate "in silico" virtual human societies, and to observe and analyze collective phenomena emerging from the local behaviors of many agents. In order to create realistic MABS, it is required to investigate how to reproduce human's realistic behavior; that is, human behavior modeling is essential research issue. We have worked on behavior modeling exploiting participatory simulation, which is one promising way to learn human behavior in the realistic environment, while most existing studies have used simple or abstract agent models [12, 19]. To date, we have considered behavior models which enables agents to take actions in response to specific physical conditions. For this modeling methodologies, we hypothesize that a human would react with identical behavior in a real context and in an accurate enough simulation of it.

The issue rests upon the definition of what we mean by accurate in this case. One observation deriving from past experiments (Hattori et al., in press) has shown that human behavior may change in ways that are difficult to explain, and not accessible to introspection (the subjects declare simply that they felt like behaving differently). Thus, the definition of the context must be as accurate as possible given the simulation technology.

In this paper, we propose an emotional path to take into account interaction with other agents. The interaction with others was previously neglected (Hattori et al. in press) but it is an obvious candidate in the more precise context specification.

Emotions play crucial roles to represent effects within- and between-individuals [18]. Realistically, it is inevitable to think that humans are occasionally driven by their emotions. Emotions distinctly affect each humans' behaviors. For example, one can immediately think that a nervous emotional state tends to lead safer behaviors while an aggressive emotion may lead riskier behaviors. Even if in the identical physical environment, a human could take a different behavior depending on the current emotion. Thus, we consider emotions as key factors to change behavior styles. Emotions also could work to form mood or atmosphere of people (*i.e.* crowd mind). Assuming evacuees in natural disasters, resulting from the propagation of nervous emotions of few people to others, a tense atmosphere, which may cause a panicking situation, could be formed. We thus consider emotions as key factors to express implicit coordination.

However, in the literature there is not yet a consensus on what dimensions of emotions are. Even less so about emotional influencing and change, or on what it means for an emotional state to evolve in time - the dynamics of emotion. The difference is striking if compared to the substantial agreement on goal-driven limited rationality. Inspired by a model of driving styles elaborated by one of the authors, we try a first step in the direction of considering emotions by limiting ourselves to a very specific case, that of *core affects* and of their transmission through *contagion*. We do not claim that this specific aspect is the most important in general - rather, what we hope is to sparkle an interest in researching what emotion theory could be useful to understand driving behavior.

In the following, we will report our idea to incorporate an intermediate emotional level into behavior models of agents used for multiagent-based urban traffic simulation. This challenge is to achieve considerable extension of agent's behavior model so that we will be able to represent sequential computation of emotional interaction among agents. We think it is one of the promising way to conduct a sensible and realistic representation of urban traffic simulation.

Concretely, we will propose 1) an agent model to process emotion-related data, 2) a mechanism to propagate emotions; that is, we will try to develop technologies to achieve internal and external effect of emotions in MABS.

## 2. IS IT REASONABLE TO INTRODUCE EMOTION IN MODELS OF DRIVING

### 2.1 Issues of driving behavior modeling

Thanks to the availability of realistic simulators, data on micro driver behavior is starting to be available, for example in the form of rule sets prioritization. In general, drivers follow simple rules like "in a curve, release the accelerator" or "if the car slows down, press on the accelerator." These rules are often in conflict, which can be solved by simple prioritization. Hattori et al (2009), in participatory simulation experiment, collect individual sets of preferences for such rules from human drivers, employing a driver simulator [10]. These individual driving styles can then be employed on different simulated tracks.

However, the question on how these styles can be considered stable remains open. For a simulator to work, one could presume they remain the same for a fixed individual. But further evidence is needed to show that, for example, a driver does not change style in reaction to changing context. In this second case, a driver is not characterized by a single behavior but by a collection of them. To select one of them, a decision level must be added to the model, different from the level that manage longer term plans as the track to destination.

In addition, these first measurements are conducted in isolation - there are still no data on the interaction between drivers. Yet, evidently observation and reaction to other cars is a critical factor that could influence style characterization and cause style change.

In the following of this paper, we propose a model of a driver with multiple driving styles, who uses an emotional state (more precisely, a core mood) to select one of them. We argue how this description would help solving, at least partially, both issues, providing a reference frame that naturally manages style changes and accounts for social interaction in the form of emotional contagion.

In the classification by Boero and Squazzoni [4], the driving classification model in [10] can be placed at the "typification" level - in fact it has been validated in a single track run, with the hope it will be shown valid on a general track. The model we propose here is instead a theoretical model, capable to be validated by abstract simulations or by the consequences it can have on typified applications.

### 2.2 Related Works

The importance of emotions as an element of driver's model cannot be underestimated. Their detection and manipulation has already been extensively studied, especially for what regards driving support systems. For example, in Grimm et al. (2007) [8], the influence of voice tones on drivers mood is studied. Emotions are considered a critical area of study for traffic applications, especially with the rise of road rage accidents. In fact, subjects "reported feeling easily provoked when driving." However, their studies disregard what is the most important dimension of emotion: the social dimension. We are instead interested in detection of emotional stated between different drivers. We argue that observation and interpretation of driving styles from other drivers could be the major source of emotional state change.

Imagine a situation where a driver's attention is caught by the car in front of him. That car, for a moment, is the most important object in his attention field. He needs to evaluate quickly it position and speed, in order to adapt his own and avoid contact. All this happens somehow automatically; in fact, adapting to another car's speed could be managed by a low-level, reflexive rule in [10]. But what if the same driver perceived something unusual in the car in front of him. As a first example, he could notice an excessive rate of adaptive steering. This would change his awareness of its object; at this point, the car is not something to follow reflexively. Instead, the exam of its behavior calls for higher cognitive functions - anomaly detection and classification. In the example, taking into account the hour of the day, the illumination of the road, and other parameters, driver A could establish that driver B is likely to be drunk or prone to a sleep accident. This detection would be likely to change his emotional state.

Interaction between cars in a road are not only dyadic. Thus, we should yalso account for group influence on the driver emotional state. This goes from the simple annoyance that can be generated when a car is stuck in a group of slow moving cars (a case in which none of the cars could actually have a representation of the group, only of its own position) to cases in which the reciprocal influencement is actually recognized (imagine a pack of fast driving cars on a highway).

In fact, emotions are exactly what we need to solve these coordination problems. Citing from the Stanford Encyclopedia of Philosophy, "they render salient only a tiny proportion of the available alternatives and of the conceivably relevant facts."

The relevance of emotions in an agent model of traffic has been long recognized; Bazzan et al. are already remarking how, to improve on models of logistic, a suitable cognitive model of the driver agent should be implemented [2]. In that work, the need for modeling a multidimensional state of the driver is clearly stated: "emotions, intentions, beliefs, motives, cultural and social constraints, impulsive actions, and even simply willingness to try" are listed as potential factors. The model they present, however, remains confined to the classic BDI approach, and the social aspect limited to having other agents as an object of reasoning. So, no communication, imitation or contagion are considered.

However, more recent simulative work seem to have abandoned this road. In a recent review, Bazzan (2009), there is no mention of emotions and the focus returns on physical characteristic of the road, like traffic lights [3]. In this context, we have provided arguments to defend the relevance of emotions. While we are aware that the model proposed in this paper leaves large room for improvement, we hope to contribute attracting attention on a critical and neglected point of view.

## 3. DRIVING BEHAVIOR MODELING WITH EMOTION-RELATED FUNCTIONS

### 3.1 Basic Idea

We adopt layered architecture perspective to construct an agent model because this architecture is useful to explicitly modularize functions of agents. For example, three-layered architecture was applied to NASA's robot for space shuttle operations. Furthermore, many teams joined in DARPA Urban Challenge 2007 applied a type of layered architectures consisting of layers for vehicle control, sequencing control operators, and route planning.

In our case, there are potentially three main modules; global route planning, emotion managing, and behavior model (see Figure 2).

### 3.2 Participatory modeling for driving behavior

In the multiagent research area, many researchers have focused on multiagent-based traffic simulations. To date, however, agent modeling with the goal of reproducing human driving behavior has not been the focus of most previous works [1, 5, 9]. Balmer *et al.*, for example, constructed a multiagent traffic simulator where each agent iteratively revises his/her preferences on a traveling route. In this work, the agent model is considerably simplified since only a decision on route setting is made. Halle and Chaib-draa [9] proposed an agent architecture for realizing collaborative driving by a

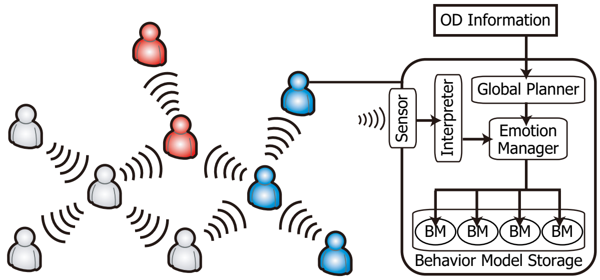


**Figure 1: Overview**

convoy of cars. In their work, however, the individuality of driving style was not considered. In contrast, Paruchuri *et al.* [15] tried to reproduce a variety of driving styles. However, they did not consider realizing human-like driving, but simply introduced three driving styles defined based on three fine-tuning parameters (cautious, normal, and aggressive).

Participatory modeling [13] is a promising technology with which to obtain individual behavior models based on actual human behavior. In participatory modeling, we can elicit a human's individual behavior as well as the reason for them in particular application domains. Such information can be used as prior knowledge to explain a human's individual behavior. For a sequence of human behaviors, we can construct an individual behavior model composed by a set of prior knowledge, each of which can explain one of the local behaviors in the sequence.

During participatory driver agent modeling, we construct driving behavior models from human driving data through the collaboration with human subjects. Using the participatory modeling technique allows us to construct behavior models from not only our (modeler's) knowledge/ability, but the actual behavior of human subjects. The modeling process consists of the following five steps.

1. Collect human driving log data from trials performed on a 3D virtual driving simulator.
2. Together with domain expert, identify individual driving behaviors by the investigation of collected log data.
3. Collect prior knowledge constituting a driving behavior model by interviewing the subjects of the driving simulation
4. Select meaningful prior knowledge and represent it in formal expression
5. Construct a driving behavior model that can explain a human subject's actions by hypothetical reasoning [16]

## 4. A MODEL WITH AN INTERMEDIATE "EMOTION" LEVEL

In this section, we will try to improve the model of traffic driving styles with the introduction of emotions, as discussed in the introduction.

The model presented in [11] described associates to every driver a *style*, that is, a single micro set of movements. In that paper, it was assumed that a subject decides his/her next operation based on the surrounding environment as observed from his/her viewpoint.

The driving model $\mathcal{M}$ was a set of prioritized driving rules $\langle P, \preceq \rangle$, which is a set of driving rules and $\preceq$ represents the priorities of each rule in $P$. $P$ is a subset of $Rules$, the set of rules obtained from all human subjects.

In larger time scales, the assumption of a single set of prioritized rules should be relaxed. A driver won't exhibit the same behavior when she is relaxed at the start of the day, comparing to her style when she comes back home in the evening's traffic jam. For this reason, we introduce in the agent model an intermediate level, driven by emotions, that will manage the choice of micro behaviors. This layer connects the rational level, where plans take shape, with a set of reactive behaviors.

This choice is somehow arbitrary and reflects the point we want to stress in the present work. What would be more accurate is to say that inner elaboration of external context in general could be useful to shift from one style to another; while some contextual assessment can very well be emotional, other ones can be rational, and work the same. However, in the present work we limit ourselves to emotional states, and even to simple one. In this sense, emotions act as a mediator between internal states of the agent and a set of driving styles. Our claim is that these, far from exhaustive, are in fact the most relevant for our purpose, and building the agent model around them can at least be a good start.

Now the question is, what are the interesting emotional states in a model of driving? Surely the most extreme ones are relevant: road rage, drunk driving, sleeping accidents are likely to have heavy personal consequences, and also consequences on a local scale, increasing the likelihood of traffic slowdown and jamming.

What about more subtle states, then? Could a depressed, subdued driver have measurable effects beyond a very local scale? What about an excited, elated style of driving? It is well know that emotions tend to propagate spontaneously: "people often catch each other's emotions" [14]. Can a subtle emotional state propagate to neighbor drivers and finally cause a measurable group effect? To answer this question, we need a model of emotional interpretation and change, situated in the specific context of driving.

In fact, driving is a very specific modality, that precludes access to many of the usual emotional hints (face and body movements). This restriction could make us think that the transmission of emotions is impaired or absent in the context of driving. However, we should also consider that focusing on a task leaves the driver highly vulnerable to peripheral clues, and thus potentially very receptive to the elaboration of emotional states.

To build a tentative model of this phenomenon, we need to answer several questions. How to represent emotions, and which emotions are the most relevant ones for what regards driving. Drivers emotions can be influenced both by objective events (we call this case $event \rightarrow emotion$) and by reaction to other's emotions (we call this case $emotion \leftrightarrow emotion$).

$event \rightarrow emotion$.

An example of an emotional change that is not generated by others' emotions, but from an event, could be an unexpected crossing pedestrian that forces an emergency brake. The emergency situation changes the drivers' emotional state, generating fear, perhaps rage, and raising the arousal level. Of course thinking of a simple event-emotion causation is oversimplifying. Modelization could start the with an evolution formula that takes into account the combined effect of current emotion and the generating event, to produce an updated emotion. In the present work, however, we limit ourselves to the simpler following case.

$emotion \leftrightarrow emotion$.

.The reciprocal influencing of emotions is the only dimension for which we actually propose a model in this work.

Thus, we think we have enough evidence to argue that emotional states should be investigated in an agent based traffic model. This could be done in several ways. One of them is producing a simplified simulation - a demonstrative two-cases scenario, with or without an emotional level. The simulation should point out where the emotional level is likely to make a difference and where it is not.

In the following, we draw the basic elements of such a model. We need a representation of an emotional state, and a mechanism for the contagion/diffusion of emotions. In a (probably incomplete) review of the literature, we found out that existing models seem to focus on describing state, but we found no simulation model that describes influencing and change.

## 4.1 Core Affects: The State

The very nature of the labels we use to define emotions is questioned by both philosophical reflection and neurological data. We can hardly tackle such a matter here. As a first choice, we will go with a classic: we will use a simplified version of the emotion classification in [17], who describe the simple, objectless emotional states called "core affects". A core affect "is assessed by asking how one is feeling right now" (p. 815), in contrast with prototypical emotions. Feeling blue is a core affect, the fear caused by the sight of a charging bear is a prototypical emotion.

Core affects are defined as the combined effect of two base dimensions, activation (from deactivated to activated) and pleasure (from pleasant to unpleasant). For example, a state of high activation and high pleasure can be labeled as elation; a state of low activation and neutral on the pleasure scale could be classified as fatigued or calm. These states could be connected to driving micro behaviors. Consider for example steering frequency. A relaxed (low activation, slightly pleasurable would naturally keep his direction, drives straight with few steering corrections. A nervous (high activation, unpleasant) driver would have much more frequent trajectory corrections. A fatigued (neutral pleasure, very low activation) driver would have difficulties keeping his trajectory straight, steering only when nearly off road.

The examples above show how core affect has a quite straightforward interpretation in terms of driving behavior; but also that these behaviors are easily characterizable by observation. We will return to this point in the next section.

For further details on core affects we refer to [17]; we are aware that more recent papers use three dimensions (e.g., valence, activation and dominance in [8]), that we could adopt in a future work. At the current stage of development, though, two dimensions looks adequate for experimentation. In fact, since the actual states lie on a circumplex (there is no state corresponding to neutral pleasure and medium activation), only one dimension is enough to characterize the points, which lie on a circle.

For our analysis, we will simply use the angle to quantify the core affect state. Approximatively, happiness lies between 0 and 20 degrees, elation between 20 and 40, and so on. We are aware that such a sharp characterization may sound irksome in a field like this one of emotions, where most definitions are vague by nature. But our model is only a first step towards the inclusion of emotions in agents.

Why, then, we need such a sharp identification of a core affect with a single number? We do that because we are aiming to have not just a representation of the emotional state, but a model of emotion contagion. We will see how this adds more complexity in the next section.

## 4.2 Core affects: Change

The introduction of core affects creates a level of complexity in the system, that has several consequences. For the single agent, it is helpful for better planning and to take preventive action. In

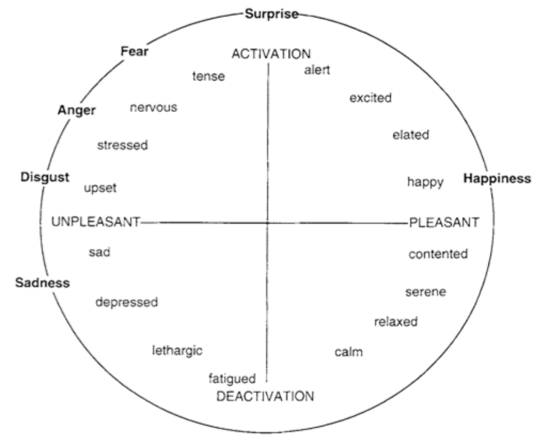


**Figure 2: Affects circumplex. Reproduced from Russell and Feldman Barrett 1999.**

addition, some negative emotions can be contagious (frustration).

Most papers on the themes of emotions and driving (including for example [8]focus on how to recognize the driver's core affect, offering hints and suggestions on how to influence it.

However, the social aspect of these changes is neglected. There is no information on how the observation and (possibly mistaken) recognition of other drivers affect status could influence one's own. In its social aspect, this phenomenon is well suited for investigation through agent-based social simulation.

Seeing the contagion happen at an aggregate level, we notice how the spreading of an emotional state can give rise to a kind of aggregate description of the system, where all the cars that share the same emotional state can be considered as single aggregate agent, thus adding an intermediate level of analysis between the single car and the general aggregates of the system. These intermediate formation can have a relevant effect on the behavior of the system as a whole (imagine a group of slow, frustrated cars that acts like an obstruction in an highway, lowering the overall path times).

To create a complete, simulable model, we must now describe how core affects do change, both in time and as a result of interaction between agents. It is known that the duration of emotional episodes depends on personality, changing from individual to individual. Duration is coherent for individuals, allowing prediction on the intensity of their reactions [7]. In an agent based model, it's not difficult to design a kind of "emotional stickiness" to agents.

About interaction between agents, the current state of the art does not seem to include numerical models. But once we have a simple coordinate system like the one described in the previous section, it is not difficult to transform the statements found in psychological research into numerical change. The first ingredient needed is of course a metric; on the unit circle, we can simply use the absolute angle between two points as a distance[1]. Examples of the statements that may be translated include.

- distant affect repel each other distance in affects pushes arousal up
- similar affects attract each other

[1] Of course, precautions must be used to take into account the specific properties of angles, as the fact that the metric has a maximum distance at $\pi$.

- points in the negative pleasure plane have higher inertia

The validity of these statements should be checked both experimentally and in simulations. In a first attempt, we decide to subdivide the core affects circumplex in coherent subsets $\{a_1...a_n\}$. We presume that core states in the same subset tend to aggregate in a coherent state, while ones in different subsets repulse each other. In the simplest case, with $n = 4$, the subset can be identified with the four quadrants.

At each moment, an agent will have a list of other agents that are in his attention field - in most cases, cars in front of it or visible in its rearview mirrors. A driver agent only attributes a core emotional state to agents in his attention field. Only a subset of these agents - the *salient* ones - will activate his emotional attention and thus contribute to the shift of his own core state. In short, we start from the following assumptions:

- agents have an affect field that consists in the set of other agents that influence their core affect.
- affects in different subsets have different effects; specifically,
  - affects are attracted by other affects in the same subset
  - are repelled by aspects in different subsets.

In practice, the simplest core state change could be modeled as such:

- The center of mass of the other agents in the same quadrant is calculated; the agent is moved towards that point by a quantity weighted by the number of agents in the calculation and by its own propensity to change.
- The center of mass of the other agents in different quadrants is calculated; the agent is moved away from that point by a quantity weighted by the number of agents in the calculation and by its own propensity to change.

We present the corresponding formulas in the appendix.

### 4.3 Core affect: observation of other agents

In a agent world, we would be tempted to allow these emotional levels to communicate directly with each other. After all, it is useful for the common good to have such a direct connection - knowing in advance that the driver in front of me is anxious or drunk will help me making better decisions. And since driving is more of a collaborative activity than of a competitive one, everyone on the road could take advantage of this.

## 5. CONCLUSION

In this paper, we have argued why emotions should be included in any multi-agent based model of traffic. While initial literature seems to be aware of the issue, current agent modeling seems to have moved away from this direction. We have proposed a simple model that leans on a numerical measure of the core affect on a circumplex and proposes a simple algorithm to manage the reciprocal influencing of agents.

The model is possible of being tested, both at the simulation level and in a laboratory test with human subject. Simulation can appraise if and how much the model proposed differs from a model that does not take into account emotions. Laboratory tests can support the choices we have made for emotion (affect) effects and for emotion (affect) change. To our knowledge, this is the first proposal of a traffic simulator that takes into account the interaction of emotional states.

Finally, the results of both experimental lines could be integrated to produce participatory simulation with emotionally plausible simulated agents.

## 6. ACKNOWLEDGMENTS

One of the authors gratefully acknowledges the Short Term Mobility Program of CNR/Italy, that allowed the exchange visit that gave the idea for this work. We thank Prof. Toru Ishida for encouragement and Maria Miceli for the suggestions and, even better, criticisms. Mistakes and ingenuity belong to the authors only. The authors also want to thank the reviewers for their valuable comments.

## 7. REFERENCES

[1] M. Balmer, N. Cetin, K. Nagel, and B. Raney. Towards truly agent-based traffic and mobility simulations. In *Proceedings of the 3rd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-04)*, pages 60–67, 2004.

[2] A. Bazzan, J. Wahle, and F. Klügl. Agents in traffic modelling - from reactive to social behavior. In *Advances in Artificial Intelligence*, number 1701, pages 303–306. Springer, 1999.

[3] A. L. C. Bazzan. Opportunities for multiagent systems and multiagent reinforcement learning in traffic control. *Autonomous Agents and Multiagent Systems*, 18(3):342–375, 2009.

[4] R. Boero and F. Squazzoni. Does empirical embeddedness matter? methodological issues on agent-based models for analytical social science. *Journal of Artificial Societies and Social Simulation*, 8(4):6, 2005.

[5] K. Dresner and P. Stone. Multiagent traffic management: A reservation-based intersection control mechanism. In *Proceedings of the 3rd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-04)*, pages 530–537, 2004.

[6] J. M. Epstein. Agent-based computational models and generative social science. *Complexity*, 4(5):41–60, 1999.

[7] E. Gilboa and W. Revelle. *Emotions: Essays on emotion theory*, chapter Personality and the structure of emotional responses, pages 135–159. Psychology Press, 1994.

[8] M. Grimm, K. Kroschel, H. Harris, C. Nass, B. Schuller, G. Rigoll, and T. Moosmayr. On the necessity and feasibility of detecting a driver's emotional state while driving. In *Affective Computing and Intelligent Interaction*, volume 4738 of *Lecture Notes in Computer Science*, chapter 12, pages 126–138. Springer Berlin Heidelberg, 2007.

[9] S. Halle and B. Chaib-draa. A collaborative driving system based on multiagent modelling and simulations. *Journal of Transportation Research Part C*, 13:320–345, 2005.

[10] H. Hattori, Y. Nakajima, and T. Ishida. Agent modeling with individual human behaviors. In *Proceedings of the 8th International Conference on Autonomous Agents and Multiagent Systems (AAMAS-2009)*, page 1369 襄Ş1370, 2009.

[11] H. Hattori, Y. Nakajima, and T. Ishida. Learning from humans: Agent modeling with individual human behaviors. *IEEE Transactions on Systems, Man, and Cybernetics-PART A: Systems and Humans*, 2010. in press.

[12] T. Moyaux, B. Chaib-draa, and S. D'Amours. Multi-agent simulation of collaborative strategies in a supply chain. In

*Proceedings of the 3rd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-04)*, pages 52–59, 2004.

[13] Y. Murakami, Y. Sugimoto, and T. Ishida. Modeling human behavior for virtual training systems. In *Proceedings of the 4th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-05)*, pages 127–132, 2005.

[14] C. Nass, I. Jonsson, H. Harris, B. Reaves, J. Endo, S. Brave, and L. Takayama. Improving automotive safety by pairing driver emotion and car voice emotion. In *Proceedings of the Human Factors in Computing Systems Conference (CHI-05)*, pages 1973–1976, 2005.

[15] P. Paruchuri, A. R. Pullalarevu, and K. Karlapalem. Multi agent simulation of unorganized traffic. In *Proceedings of the 1st International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-02)*, pages 176–183, 2002.

[16] D. Poole. *The Knowledge Frontier*, chapter Theorist: A logical reasoning system for defaults and diagnosis. Springer-Verlag, 1987.

[17] J. A. Russell and L. F. Barrett. Core affect, prototypical emotional episodes, and other things called emotion: Dissecting the elephant. *Journal of Personality and Social Psychology*, 76(5):805–819, 1999.

[18] R. Trappl, P. Petta, and S. Payr, editors. *Emotions in Humans and Artifacts*. MIT Press, 202.

[19] T. Yamashita, K. Izumi, K. Kurumatani, and H. Nakashima. Smooth traffic flow with a cooperative car navigation system. In *Proceedings of the 4th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-05)*, pages 478–485, 2005.

## 7.1 Appendix

In this appendix, we report on the formulas that we propose for core affect update. We model cars as agents in the set $A = a_1 ... a_n$. Each agent is in an emotional state $e_i$. At every moment of time an agent $a_i$ has a perception range, that we model simply as a set of cars influencing her emotional state: $P_i$, containing $NP_i$ cars. Of these cars, some will be in the same emotional quadrant as the agent ($P_i^s$) and others will be in a different quadrant ($P_i^d$).

Influence from similar and dissimilar agents will be weighted both by individual characteristics (influenceability) and by the sheer number of agents perceived. We give the first two absolute weight, $\alpha_i^s$ and $\alpha_i^d$, respectively, for the (positive) influencing susceptibility to contagion from agents in the same and in a different quadrant (with $\alpha^s + \alpha^d < 1$ at fixed $i$). The second we weight with the simple weighting function $1/(1 + e^{\quad N})$, where $N$ stands for the number of agents in the same or different states, respectively $N_{P^s}$ and $N_{P^d}$, with agent $i$ index implied. The weights become then

$$w^s = \alpha_i^s \frac{1}{1 + e^{\quad N_{P^s}}}$$

and

$$w^d = \alpha_i^d \frac{1}{1 + e^{\quad N_{P^d}}}$$

The average of influences from the same quadrant has the nice property or remaining in the quadrant itself, so that moving towards it can be done with no issues. Instead, the last missing ingredient consists in an interpretation of the repulsion statement - to push the agent away from an emotional state, we simply pull it towards the opposite state, obtained by shifting it of $\pi$ and normalizing. However, this inverted state - let's call it $e^d$, needs not to be in the same quadrant as the current one (though it cannot be in the opposite one). So a little care is to applied in making sure that the formula really pushes away and not forwards the state under exam. Essentially, if the difference between $e_i$ and $e^d$ is more than $\pi$, then we need to use the complement representation of $e^d$, that is, if the current representation is in $[0, 2\pi)$, the value $2\pi \quad e^d$. We call this operator $R_{e_i}$.

Now, with all the ingredients in place, the update formula is quickly written:

$$e_i' = (1 \quad w^s \quad w^d)e_i + w^s \sum_{P^s} \frac{e_{k_i}}{N_{P^s}} + w^d R_{e_i}\left(\sum_{P^d} \frac{e_{k_i}}{N_{P^d}}\right)$$

# A market-based approach to accommodate user preferences in reservation-based traffic management

Matteo Vasirani
Centre for Intelligent Information Technology
University Rey Juan Carlos
Madrid, Spain
matteo.vasirani@urjc.es

Sascha Ossowski
Centre for Intelligent Information Technology
University Rey Juan Carlos
Madrid, Spain
sascha.ossowski@urjc.es

## ABSTRACT

Removing the human driver from the control loop by the use of *autonomous vehicles* and the integration of these with the *traffic management infrastructure* is a challenging long-term vision for the field of Intelligent Transportation Systems (ITS). Setting out from a recently proposed urban infrastructure that allows for autonomous vehicles to individually reserve space and time inside an intersection to safely cross them, multiagent approaches have been applied to simulate and to manage both single intersections as well as networks of intersections, achieving significant improvements in the drivers average travel times. However, these approaches do not take full advantage of the potential of vehicle-centric ITS, as they ignore the fact that different drivers may value their travel times quite differently.

In this paper we combine two different market-based mechanisms, acting at intersection level and at network level, respectively, to accommodate driver preferences in reservation-based urban traffic management. At intersection level, intersection manager agents assign space-time slots through combinatorial auctions, while at network level a pricing scheme, based on general market equilibrium, accounts for an efficient use of the available network resources. Our experiments show that this combined approach on the one hand allows drivers to effectively improve their travel times if they are willing to pay more money for their trip, while on the other hand the negative impact on social welfare (average travel times) is unnoticeable.

## Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Coherence and coordination, intelligent agents, multiagent systems*

## General Terms

Algorithms, Design, Experimentation

## Keywords

Traffic and transportation, market-based mechanisms, combinatorial auctions

## 1. INTRODUCTION

Talk about autonomous vehicles that interact with an intelligent traffic infrastructure always sounds far-fetched, but such a scenario may be closer than we think. Indeed, removing (at least partially) the human driver from the control loop by the use of autonomous vehicles and the integration of these with the intelligent infrastructure is a challenging long-term vision for the field of Intelligent Transportation Systems (ITS). Autonomous vehicles are already a reality: two DARPA Grand Challenge and one DARPA Urban Challenge[1] have been hitherto celebrated, where autonomous vehicles have successfully interacted with both manned and unmanned vehicular traffic in an urban environment. In line with this vision, the IntelliDrive initiative[2] promotes research and development of technologies to directly link road vehicles to their physical surroundings. The advantages of such an integration span from improved road safety to a more efficient operational use of the transportation network.

However, the level of integration will most likely be limited by the individual needs and preferences of the human users of the vehicles, who will have the final say regarding the basic characteristics of their trips. Thus, managing next-generation integrated infrastructures for ITS means regulating a large-scale open distributed system populated by a huge number of autonomous, individually rational agents. This scenario is particularly well-suited for multiagent systems technology in an urban context, because management actions can target vehicles *individually*, instead of whole *flows* of traffic as, for instance, mechanisms based on the coordination of traffic light cycles do [10]. To this respect, infrastructure facilities that allow each autonomous vehicle to reserve time-space-slots at interactions, so as to safely transit through them, have been studied both for single intersections [4] and for networks of intersections [17]. Still, these reservation-based regulation mechanisms do not consider the fact that different drivers have different preferences.

In this paper we present an economically-inspired policy for managing future reservation-based urban traffic management infrastructures that takes into account the drivers' different valuations of their vehicles' travel times. At the intersection level, vehicles compete for the right to cross intersections through combinatorial auctions, while at the network level a pricing scheme, based on general market equilibrium, accounts for an efficient use of network resources (time-space slots at intersections). In Section 2 we briefly outline previous work in the field. Section 3 evaluates our auction-based intersection control mechanism. Section 4 shows how a distributed pricing scheme, by setting the publicly-known reserve prices of the auctions, leads to a combined policy that allows vehicles to travel the faster through the network the more their drivers are willing to pay for the trip, with

[1]http://www.darpa.mil/grandchallenge/index.asp
[2]http://www.intellidriveusa.org

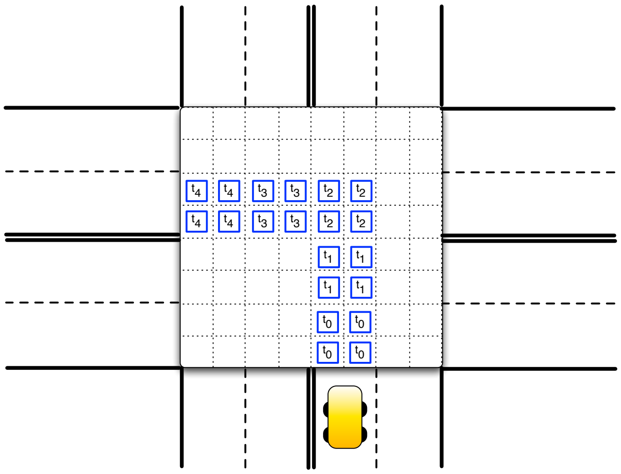

**Figure 1: Bundle of items defined by a reservation request. The vehicle needs the necessary space slots at time $t_0 \dots t_4$ to transit through the intersection.**

no significant social cost in terms of average travel times. Section 5 summarises the lessons that we have learnt.

## 2. PREVIOUS WORK

The applications of multiagent systems technology to the field of traffic and transportation are manifold [1]. In the context of urban traffic management, much work focuses on automation systems embedded in control devices that work at the operational level [19], as well as on distributed control [8] for traditional infrastructures. With the exception of some recent work [4] [16] [17], few authors have paid attention to the potential of a tighter integration of vehicles and control elements in future urban road traffic management infrastructures.

This paper sets out from the work of Dresner and Stone [4], who examine a minimally centralised infrastructure facility that allows for the control of intersections. In their model, an intersection is regulated by an intelligent agent, called *intersection manager*, which assigns reservations of space and time to each autonomous vehicle, operated by a *driver agent*, intending to cross the intersection. When a vehicle is approaching an intersection, the driver agent requests the intersection manager to reserve the necessary time-space slots to safely transit through the intersection. The intersection manager, provided with data such as vehicle ID, vehicle size, arrival time, arrival speed, type of turn, etc., simulates the vehicle's trajectory inside the intersection and informs the driver agent whether or not its request conflicts with the already confirmed reservations. If there is no such conflict, the driver agent stores the reservation details and tries to meet them; otherwise it may try again at a later time. Such an approach has shown, in a simulated environment, several advantages, because it may drastically reduce delays with respect to traffic lights.

```
(request reservation
              :sender   D-4888
            :receiver   IM-05402
            :content(
                        :arrival_time    18:03:15
                        :arrival_speed   33Km/h
                        :lane            1
                        :type_of_turn    STRAIGHT
                        :bid             1.45 €
                    )
 )
```

**Figure 2: Example of a bid in a REQUEST message**

## 3. TRAFFIC CONTROL: BEYOND FCFS

Any traffic control system is driven by the principle of optimising the use of the available resources. In the case of a single reservation-based intersection, this implies that the policy followed by the intersection manager for granting or rejecting the reservation requests should maximise the intersection's throughput. In [18], for instance, Dresner and Stone's *first-come-first-served* (FCFS) policy is compared to several other control regimes inspired by adversarial queuing theory in terms of the vehicles' average delay. However, this metric ignores the fact that in the real world, depending on the context and their personal situation, people value the importance of travel times and delays quite differently. In this section we present a control policy for reservation-based intersections that relies on an auction mechanism, so as to allocate their resources to the agents that value them the most. We specify the auction design space (resources, bidding rules, clearing policy, etc.) and how the original protocol for intersection control proposed by Dresner and Stone is modified. In the following, we use the term *bidder* or *driver agent* to refer to the agent that operates an autonomous vehicle and submits bids to acquire a reservation request.

### 3.1 Auctioned resources

In our scenario, the auctioned good is the use of the space inside the intersection at a given time. An intersection is modelled as a discrete matrix of space slots. Be $\mathcal{S}$ the set of the intersection space slots, $\mathcal{S} = \{s_1, s_2, \dots, s_m\}$. Be $t_{now}$ the actual time, and $\mathcal{T}(t_{now}) = \{t_{now} + \tau, \forall \tau \in \mathbb{N}\}$ the set of (future) time steps. The set of items that a bidder can bid for is the set $\mathcal{I} = \mathcal{S} \times \mathcal{T}(t_{now})$.

Due to the nature of the problem, a bidder is only interested in bundles of items over the set $\mathcal{I}$. In fact, a reservation request implicitly defines which space slots at which time the driver agent needs in order to transit through (see Figure 1). Thus, the items must be necessarily allocated by a combinatorial auction.

### 3.2 Bidding rules

The bidding rules define the form of a valid bid accepted by the auction. Note that the bundle of items the bid refers to is implicitly defined by the reservation request. Given the parameters *arrival time*, *arrival speed*, *lane* and *type of turn*, the auctioneer (i.e., the intersection manager) is able to determine which space slots at which time are needed. The only additional parameter that a driver agent must include in its reservation request is the amount of money that it bids

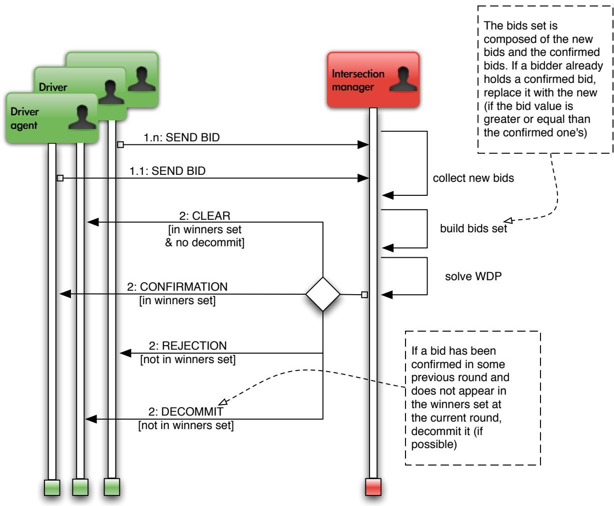


**Figure 3: Auction protocol**

(see figure 2).

In our scenario, a bidder is allowed to withdraw its bid and to submit a new one, if the new bid is greater or equal to the old one. This avoids that a bidder first acquires a reservation with an overpriced bid, and then iteratively tries to resubmit lower bids in order to obtain the same reservation at a lower price.

## 3.3 Auction protocol

The auction proceeds as a continuous alternation of two phases: *bids collection* and *winner determination*. The protocol (see Figure 3) starts with the auctioneer waiting for bids for a certain amount of time. Once the new bids are collected, they will form the *bids set*. Then the auctioneer executes the winner determination algorithm, and the *winners set* is built, containing the bids whose reservation requests are provisionally accepted. The auctioneer sends a CONFIRMATION message to all bidders that submitted the bids contained in the winners set, while a REJECTION message is sent to the bidders that submitted the remaining bids.

Then a new round begins, and the auctioneer collects new incoming bids for a certain amount of time. Once the new bids are collected, the bids set is built as the union of the new bids and the provisionally accepted bids (i.e. the winner of previous bidding rounds)[3]. After having executed the winner determination algorithm, the auctioneer sends a CONFIRMATION message to the bidders whose bid is in the winners set, unless such confirmation has already been sent in a previous round. For all the other bids, the auctioneer sends either a REJECTION message or a DECOMMIT message. The DECOMMIT message is sent to the bidders whose bids have been provisionally accepted in a previous round, but do not belong to the current winners set anymore.

[3]We remark that even a bidder that submitted a winning bid is allowed to resubmit a new bid, which will replace the old one. This is because a driver agent may want to change its provisionally accepted reservation when it realises that it is unable to actually use the reservation due to changing traffic conditions.

**Algorithm 1** Winner determination algorithm

$\mathcal{B} \leftarrow allBids$
$\mathcal{W} \leftarrow \emptyset$
$\mathcal{C} \leftarrow notDecommitBids$
$start \leftarrow currentTime$
**while** $currentTime - start < time\ window$ **do**
  $\mathcal{A} \leftarrow \mathcal{C}$
  **for** $step = 1$ to $|\mathcal{B}|$ **do**
    $random \leftarrow drawUniformDistribution[0-1]$
    **if** $random < wp$ **then**
      $b \leftarrow selectRandomlyFrom\mathcal{B} \setminus \mathcal{A}$
    **else**
      $highest \leftarrow selectHighestFrom\mathcal{B} \setminus \mathcal{A}$
      $secondHighest \leftarrow selectSecondHighestFrom\mathcal{B} \setminus \mathcal{A}$
      **if** $highest.age \geq secondHighest.age$ **then**
        $b \leftarrow highest$
      **else**
        $random \leftarrow drawUniformDistribution[0-1]$
        **if** $random < np$ **then**
          $b \leftarrow secondHighest$
        **else**
          $b \leftarrow highest$
        **end if**
      **end if**
    **end if**
    **if** $\mathcal{N}(b) \bigcap \mathcal{C} = \emptyset$ **then**
      $\mathcal{A} \leftarrow \mathcal{A} \bigcup \{b\} \setminus \mathcal{N}(b)$
      **if** $\mathcal{A}.value > \mathcal{W}.value$ **then**
        $\mathcal{W} \leftarrow \mathcal{A}$
      **end if**
    **end if**
  **end for**
**end while**

This mechanism avoids that a low-valued bid, in the winners set at round $k$, impedes the allocation of the disputed reservation to some high-valued bids, submitted at round $k+n$. A bid can be de-committed as long as the driver agent that submitted the bid can safely decelerate and reach zero speed before the arrival time at the intersection[4]. At the end of any round, the auctioneer sends a CLEAR message to the bidders whose bids are in the winners set and cannot be de-committed. Notice that, in general, for driver agents approaching an intersection it is rational to treat their provisionally accepted bids as if they were cleared, as they can safely decelerate in case of a DECOMMIT.

## 3.4 Winner determination algorithm

Since the auction must be performed in real-time, both the bid collection and the winner determination phases must be time-bounded. This implies that optimal and complete algorithms for the winner determination problem (WDP), as those proposed by Leyton-Brown et al. [11] or by Sandholm [15], are not suited for this kind of auction. An algorithm with *anytime* properties is needed, such as the stochastic local search proposed by Hoos et al. [7] that we have adapted to our scenario in order to manage the de-commitment of bids.

The algorithm starts initialising the set $\mathcal{B}$ with all bids (new ones and confirmed ones). The winners set $\mathcal{W}$ is initially empty, while the set $\mathcal{C}$ at first contains all confirmed

[4]This condition is determined as follows: be $v_a$ the arrival speed, $b$ a deceleration factor, and $t_a$ the arrival time at the intersection. The deceleration equation is defined by $v(t) = v_a - b \cdot t = 0$. Thus, the vehicle can safely reach zero speed before reaching the intersection if $t = v_a/b < t_a$.

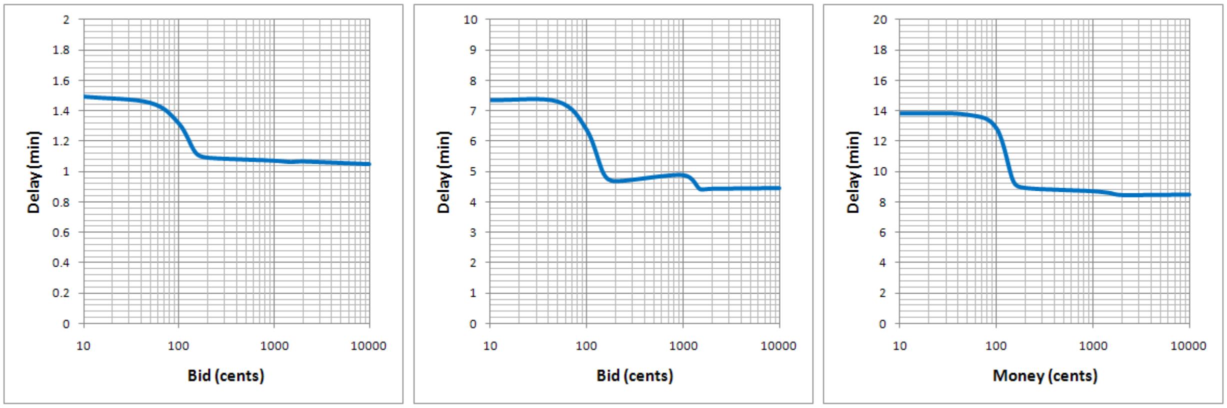


**Figure 4: Bid-delay relation ($\lambda = 10$, $\lambda = 20$, $\lambda = 30$). Please note the different scale of the y-axis in the three plots.**

bids that cannot be de-committed. Once the initialisation has concluded, the algorithm executes the main loop. Within this loop, a stochastic search is performed for a number of steps equal to the number of bids contained in $\mathcal{B}$. The set $\mathcal{A}$, which at every step contains the candidate bids for the winners set, is initialised with the bids $\mathcal{C}$ that cannot be de-committed. Then, with probability $wp$ (walk probability), a random bid is selected from the set of bids that are not actually in the candidate winners set ($\mathcal{B} \setminus \mathcal{A}$). Otherwise, with probability $1 - wp$, the highest and the second highest bids are evaluated. The highest bid is selected if its age (i.e., the number of steps since a bid was last selected to be added to a candidate solution) is greater or equal to the age of the second highest. Otherwise, with probability $np$ (novelty probability), the second highest is selected, and with probability $1 - np$ the highest is selected. Once the bid $b$ to be added to the candidate solution has been selected, the neighbourhood of $b$, $\mathcal{N}(b)$, is evaluated. The neighbourhood of a bid $b$ is defined by the set of bids for bundles that share with $b$ at least one item. If the neighbourhood $\mathcal{N}(b)$ does not contain any bid that cannot be de-committed, the bid $b$ is added to the candidate solution $\mathcal{A}$ and all the neighbours of $b$ are removed from $\mathcal{A}$. Finally, if the value of $\mathcal{A}$ (i.e., the sum of the bids $\in \mathcal{A}$) is greater than the value of the best-so-far winners set, $\mathcal{W}$, the best solution found so far is updated.

## 3.5 Experimental results

To evaluate the auction-based policy, we simulate a single intersection with 4 incoming links of 3 lanes each. We simulate different traffic demands by varying the expected number of vehicles ($\lambda$) that, for every origin-destination pair (i.e., the 4 incoming links), are spawned in an interval of 60 seconds. We spawned vehicles for a total time of 10 minutes. In the following, we refer to the auction-based policy as CA and to the first-come-first-served policy as FCFS.

The main goal of this set of experiments is to confirm that the auction-based policy enforces an inverse relation between money spent by the bidders and their delay. The delay measures the increase in travel time due to the presence of the intersection. It is computed as the difference between the travel time when the intersection is regulated by the intersection manager and the travel time that would

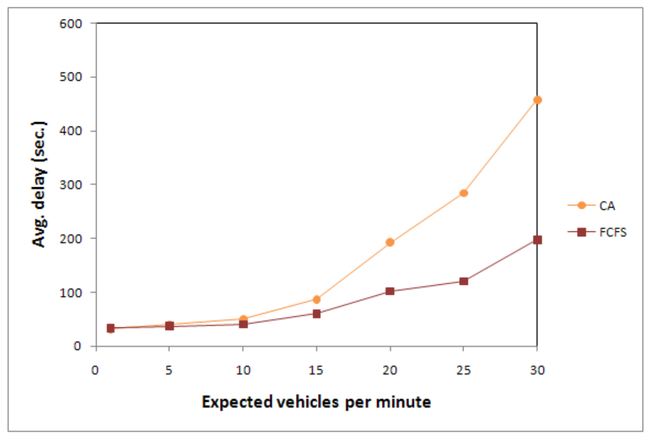


**Figure 5: Average delay**

arise if the vehicle could travel unhindered through the intersection. We generated an artificial population of bidders whose initial endowment is drawn from a normal distribution with mean 100 cents and variance 25 cents, since the willingness to pay of human drivers is usually normally (or log-normally) distributed [6]. In this population, we inserted a set of driver agents, which we use as floating cars to evaluate their delay, endowed with 10, 50, 100, 150, 200, 1000, 1500, 2000 and 10000 cents. We also evaluated the auction-based policy with respect to the average delay of the entire population of driver agents. In all the experiments, we gave the intersection manager 1 second to collect the incoming bids and another second to execute the winner determination algorithm and return a solution. Regarding the parameters of the winner determination algorithm, we set the walk probability $wp$ and the novelty probability $np$ to 0.15 and 0.5, respectively. These values were reported by Hoos et al. [7] to give the best results in similar types of auction (number of bidders, expected size of bundles).

Figure 4 plots the relation between delay and bid value for different values of $\lambda$. There is a sensible decrease of the delay (between 30% and 40%) experienced by the driver agents which bid from 100 to 150 cents with respect to those that

**Algorithm 2** Reserve price update

$t \leftarrow 0$
**for all** $l_h \in \mathcal{L}_j$ **do**
  $p_j^t(l_h) \leftarrow \epsilon$
  $s_j(l_h) \leftarrow initialValue$
**end for**
**while** true **do**
  **for all** $l_h \in \mathcal{L}_j$ **do**
    $d_j^t(l_h) \leftarrow evaluateDemand$
    $z_j^t(l_h) \leftarrow d_j^t(l_h) - s_j(l_h)$
    $p_j(l_h) \leftarrow updatePrice(\epsilon, z_j^t(l_h), s_j(l_h))$
  **end for**
  $t \leftarrow t+1$
**end while**

bid less. Nevertheless, the delay reduction tends to settle for driver agents that bid more than 150 cents. This reflects the fact that as the traffic demand increases, the chance that even driver agents with high bids will not be able to travel through the intersection at the desired speed grows as well. Consider a vehicle with a wealthy driver who is in a hurry, travelling behind a vehicle that does not intend to allocate much money to acquire a reservation, and being too close to the intersection to overtake it. In such a case, even the highest bid would not be effective, because it would be impossible to actually make use of the reservation gained in the auction.

Figure 5 plots the average delay for different traffic demands ($\lambda \in [1, 30]$). When the traffic demand falls between 1 and 15 expected vehicles per minute, the performance of the CA policy and the FCFS policy is approximately the same. Still, when the traffic demand increases ($\lambda \geq 20$), from the point of view of social welfare, the CA policy performs worse than the FCFS, with a noticeable increase of the average delay. Of course, this is not surprising, as the CA policy was designed to grant a reservation to the driver agent that values it most, rather than maximising the number of granted requests.

# 4. TRAFFIC ASSIGNMENT: A GENERAL EQUILIBRIUM PERSPECTIVE

As seen in the experiments of the previous section, for single reservation-based intersections under high demand, the CA policy entails a significant social cost, in terms of a greater average delay for the entire population of driver agents. For this reason, if we focus on a urban road network with multiple intersections, an integrated strategy is needed that combines traffic *control* and *assignment*, i.e. which distributes traffic flows over the network elements in line with their capacities, thus reducing the demand of potentially congested intersections.

## 4.1 Reserve price update strategy

From an economic perspective, an intersection manager is a supplier of reservations, which it then allocates through a combinatorial auction. Thus, it controls the *reserve price* of the auctioned reservations, i.e. the minimum price at which it is willing to sell [20]. Depending on the intersection usage, the intersection manager may apply pricing strategies and modify this reserve price.

Following the market metaphor, our intersection managers compete with each other for driver agents, raising prices in case of increasing demand or lowering them in case of decreasing demand. The pricing strategy is based on the general market equilibrium theory [2] [3]. The adaptive and concurrent pricing strategies applied by the intersection managers are in charge of computing in a distributed way the price vector $\mathbf{p}^*$ that corresponds to the general market equilibrium, a situation where the amount of resources sought by the driver agents is equal to the amount of resources supplied by the network.

Be $\mathcal{L}_j$ the set of incoming links of intersection $j$. For each incoming link $l_h \in \mathcal{L}_j$, the intersection manager defines the following variables:

- Current reserve price $p_j^t(l_h)$: the reserve price applied by the intersection manager $j$ for the auctions that allocate the reservations to the driver agents of the incoming link $l_h$.
- Total demand $d_j^t(l_h \mid p_j^t(l_h))$: represents the total demand at time $t$, i.e., the number of driver agents on link $l_h$ that are bidding for a reservation.
- Supply $s_j(l_h)$: defines the supply of intersection manager $j$ for the incoming link $l_h$, i.e., the number of driver agents that intersection manager $j$ wants to participate in each auction.
- Excess demand $z_j^t(l_h \mid p_j^t(l_h))$: the difference between the total demand at time $t$ and the supply, $z_j^t(l_h \mid p_j^t(l_h)) = d_j^t(l_h \mid p_j^t(l_h)) - s_j(l_h)$. We remark that the excess demand can be negative, when the demand is lower than the supply.

Given the set of all intersection managers that are operating in the market, $\mathcal{J}$, we define the price vector $\mathbf{p}$ as the vector of the reserve prices applied by each intersection manager $j \in \mathcal{J}$. To enforce the attainment of the general equilibrium, each intersection manager applies the reserve price update strategy outlined in Algorithm 2. At time $t$, each intersection manager $j$ computes, independently from each other, the excess demand $z_j^t(l_h \mid p_j^t(l_h))$ and updates the price $p_j^t(l_h)$ using the formula:

$$p_j^{t+1}(l_h) \leftarrow max\left[\epsilon,\ p_j^t(l_h) + p_j^t(l_h) \cdot \frac{z_j^t(l_h \mid p_j^t(l_h))}{s_j(l_h)}\right] \quad (1)$$

where $\epsilon$ is the minimum reserve price and $s_j(l_h)$ is the number of driver agents that intersection manager $j$ wants to participate in each auction. The definition of $\epsilon$ and $s_j(l_h)$ is a design decision that may affect the dynamics of the market: i) $\epsilon$ is the minimum reserve price above which a bidder must bid to get a reservation and ii) $s_j(l_h)$ is the number of vehicles above which the intersection manager considers that there is an excess demand and starts raising prices. Vehicles travelling on network links with low demand shall incur in costs as low as possible, so we chose $\epsilon = 0$. To define the supply $s_j(l_h)$, we rely on the fundamental diagram of traffic flow [12]. Let $\rho^{opt}$ be the density that maximises the traffic flow on link $l_h$. We chose $s_j(l_h) = 0.5 \cdot \rho^{opt} \cdot ||l_h||$, where $||l_h||$ is the length of link $l_h$. In other words, the intersection manager considers that there is an excess demand when the density on link $l_h$ reaches the 50% of the optimal density. We remark that in order to apply the reserve price update strategy, there is no need of communication between the intersection managers, since they are able to compute locally the total demand at time $t$, counting the number of driver agents that are bidding for a reservation.

## 4.2 Driver agent model

Differently from the single intersection scenario evaluated in Section 3, in case of a network of intersections we need a reasonable model for the vehicles' route choice. We assume that driver agents have a model of the road network that enables the computation of the travel time at free flow. Furthermore, we assume that the intersection reserve prices are available to the driver agent, published on some sort of price index board[5]. Each driver agent holds a private valuation of the bids that it is willing to submit to transit through the intersections of its chosen route, defined by the variable $b_i$. Given the monetary constraint, the driver agent selects the most preferred route $r^*$, taking into consideration the estimated travel time associated with the route. More formally, we model a route $r$ as an ordered list of links, $r = [l_1, \ldots, l_M]$, each of them characterised by two attributes, namely travel time at free flow

$$TT^{free}(l_k) = \frac{||l_k||}{v_{max}(l_k)} \tag{2}$$

and reserve price

$$K(l_k) = \begin{cases} p_j^t(l_k) & if\ l_k \in \mathcal{L}_j \\ 0 & otherwise \end{cases} \tag{3}$$

where $||l_k||$ is the length of link $l_k$, $v_{max}(l_k)$ is the maximum allowed speed on link $l_k$, and $p_j^t(l_k)$ is the reserve price set by intersection manager $j$ that governs the intersection which the link $l_k$ is connected to. The sum over all the links of $r$ gives the travel time at free flow of the entire route $r$:

$$TT^{free}(r) = \sum_{k=1}^{M} TT^{free}(l_k) \tag{4}$$

Based on the bids $b_i$ that the driver agent plans to submit, the choice set $\mathcal{R}$ is given by those routes whose intersections have a reserve price lower than the bid $b_i$:

$$\mathcal{R} = \{r_1, \ldots, r_N \mid K(l_k) \leq b_i \ \forall l_k \in r_x\} \tag{5}$$

Once the choice set is built, the driver agent selects the shortest route $r^* = argmin_{r \in \mathcal{R}} TT^{free}(r)$. Since the reserve prices change with time, a driver agent may react to the price fluctuations and rearrange its route on-the-fly.

## 4.3 Experimental results

To evaluate our approach, we use a hybrid mesoscopic-microscopic simulator. The traffic flow on road sections is modelled at mesoscopic level but, as a higher level of detail is required for reservation-based intersections, when a vehicle enters an intersection, its dynamic switches to a microscopic, cellular-based simulation, whose update rules follow the Nagel-Schreckenberg [13] model. Although our work is independent from the underlying road network, we chose a (simplified) topology of the urban road network of the city of Madrid (see Figure 6) for our empirical evaluation rather than an unrealistic, lattice-like, network. Each big dark vertex in Figure 6 that connects three or more links is modelled as a reservation-based intersection. We aimed at recreating a typical morning peak scenario, with more than 11000 vehicles that depart within a time window of 50 minutes from/to 7 destinations outside the city (marked with $O_1$ up to $O_7$ in Figure 6).

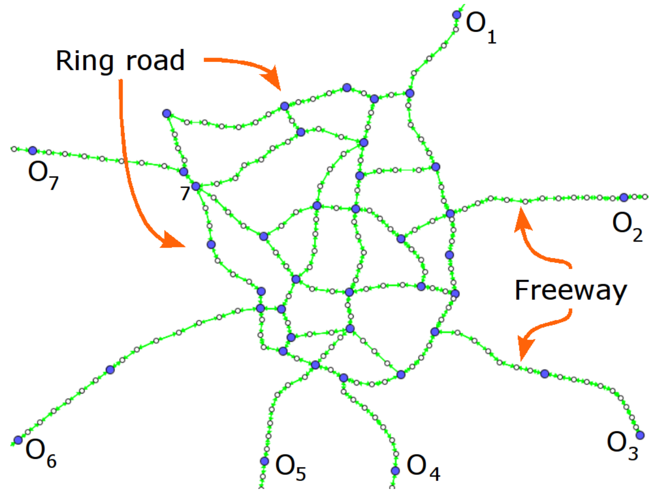


**Figure 6: Simulator of an urban road network**

In our first experiment, the intersection managers apply the reserve price update strategy described in Subsection 4.1, and assign reservations to driver agents using the auction-based policy described in Section 3 (referred to as CA policy). The goal is to verify that our integrated policy effectively guarantees lower delays to driver agents that submit higher bids. For this purpose, we calculate the average increase of the vehicles' travel times, defined as

$$\frac{TT_i^{real} - TT^{lower\ bound}}{TT^{lower\ bound}} \tag{6}$$

being $TT_i^{real}$ the observed travel time for vehicle $i$ from an origin to a destination, and $TT^{lower\ bound}$ the travel time from the same origin to the same destination if the vehicle could cross each intersection unhindered[6]. For simplicity, we refer to the percentage increase of the travel time with the term normalised delay. Figure 7 plots the relation between bid value and normalised delay of the population of driver agents. As in the experiments of Subsection 3.5, it is possible to appreciate an inverse relation between these two quantities. The driver agents that submit bids between 150 and 200 cents reduce the delay of about the 50% with respect to those which bid less than 50 cents.

In a second experiment, we compared the CA policy to networks of intersections governed by the first-come-first-served control policy without assignment (FCFS)[7]. The aim is to evaluate the global performance (in terms of average

[5]This is technically feasible already today: for instance, the NYSE indexes approximately 8500 stocks, whose price variations are spread worldwide almost immediately.

[6]This ratio enables us to aggregate the results of driver agents even though they have different origins and/or destinations.

[7]We assume that in this case the driver agents choose the shortest route from their origin to their destination.

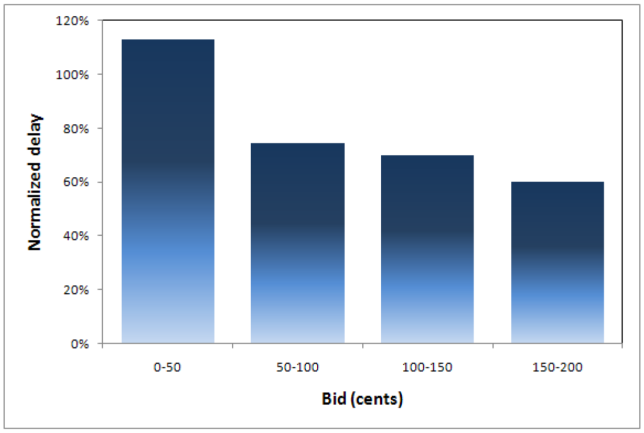


**Figure 7: Relation between normalised delay and bid**

travel time) of the sophisticated CA policy compared to the rather simple FCFS policy, and to detect any potential social cost of CA, similar to the one reported in Section 3. Table 1 shows the average travel time of the driver agents, according to their origin-destination pairs, when the reservations are allocated with the CA policy, and when they are granted with the FCFS policy. With CA, there is a net *reduction* of the average travel time for 29 of 42 origin-destination pairs. This reduction is particularly significant for the "busy" routes that connect $O_5$ and $O_6$ with $O_3$ and $O_4$.

Figure 8 plots the moving average of the travel times in our experiment, as a function of the percentage of completed trips. In the beginning, the average travel time is similar for both control policies, but as the number of driver agents that populate the network (i.e., its load) increases, it grows significantly faster with the FCFS than with the CA policy. This is due to the fact that the higher the network load, the more relevant the different reserve prices: with the CA policy, more drivers choose a route through less expensive intersections, thus leading to less demand at "bottleneck" intersections. There are two consequences: (1) in line with Figure 5, lower demand leads to a lower "social cost" of our auction-based policy with respect to FCFS at intersection level; and (2) a more homogeneous distribution of vehicles over the network leads to a better use of network resources, and thus to lower average travel times. Our experiments show that, in general, the gains obtained by (2) outweigh the overhead introduced by (1) with respect to social welfare (i.e., average travel time).

# 5. CONCLUSIONS

In this paper we have presented an economically inspired policy to accommodate the preferences of users of autonomous vehicles in reservation-based urban traffic management. At intersection level, intersection manager agents assign space-time slots through combinatorial auctions, so as to give priority to vehicles whose drivers are willing to pay more. At network level, a decentralised pricing scheme, targeting the general market equilibrium, (implicitly) coordinates the reserve prices of the intersections' auctions. As a result, we obtain a system in dynamic equilibrium where unused intersections become cheaper while more demanded ones become more expensive, leading to a more efficient use of the network resources. We have shown that vehicles whose drivers are willing to pay more for their reservations are effectively rewarded with lower travel times, while the social cost of our policy when compared to FCFS is reduced and often even compensated by the traffic assignment effects of our policy.

**Table 1: Average travel time (min): CA (upper) vs. FCFS (lower)**

| Origin | Destination $O_1$ | $O_2$ | $O_3$ | $O_4$ | $O_5$ | $O_6$ | $O_7$ |
|---|---|---|---|---|---|---|---|
| $O_1$ | -<br>- | 12.10<br>**11.98** | **13.76**<br>22.88 | **24.00**<br>35.13 | **27.27**<br>43.56 | 22.61<br>**21.35** | **13.82**<br>13.82 |
| $O_2$ | 11.06<br>**10.14** | -<br>- | **10.90**<br>16.50 | **19.01**<br>25.86 | **23.97**<br>31.04 | **25.87**<br>38.09 | 21.00<br>**19.50** |
| $O_3$ | 14.95<br>**13.34** | 12.85<br>**9.75** | -<br>- | **9.18**<br>12.21 | **13.53**<br>17.63 | **19.03**<br>23.68 | **27.90**<br>31.73 |
| $O_4$ | **19.73**<br>26.94 | **18.29**<br>22.58 | **10.01**<br>13.91 | -<br>- | **7.13**<br>10.04 | **13.11**<br>15.73 | 23.30<br>**22.74** |
| $O_5$ | **25.04**<br>32.16 | **20.74**<br>30.61 | **12.07**<br>21.53 | **7.47**<br>8.83 | -<br>- | **10.05**<br>10.77 | 21.36<br>**17.65** |
| $O_6$ | 24.46<br>**22.51** | **27.24**<br>57.00 | **16.33**<br>41.05 | **16.39**<br>24.68 | **10.35**<br>19.02 | -<br>- | 14.16<br>**13.73** |
| $O_7$ | 14.82<br>**14.30** | 23.93<br>**23.25** | **20.09**<br>56.42 | **27.53**<br>34.99 | **16.92**<br>31.23 | 12.37<br>**11.99** | -<br>- |

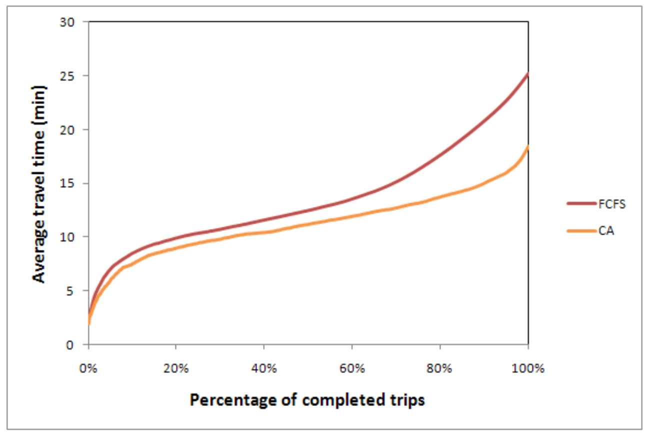


**Figure 8: Moving average of travel time**

Our approach differs from other work on reservation-based traffic management, such as [4] and [17], with regard to its primary objective: instead of trying to improve the travel time of everyone (social welfare), our management policy intends to assign the available resources in the network to the drivers that value them most (i.e., that are willing to pay more for them). A similar objective underlies the work by Schepperle and Bohm [16], although they do not account for networks of intersections. In their work it is the intersection manager that initiates a Vickrey auction, offering the earliest time slot to the first vehicles that are approaching the intersection on each lane.

A natural extension of this work is to include mechanisms that address the queuing phenomenon at intersection level outlined in Section 3. In particular, vehicle-to-vehicle communication [21] could be used to enrich the action space of a driver agent. In this way, a wealthy driver that is in a hurry but is stuck behind a vehicle that is not willing to allocate much money to acquire a reservation could directly subsidise it as in [16], or it could form a coalition with it to submit a joint bid. Research on platoon formation could also be relevant to this respect [5].

Another extension refers to the driver agent models used in the experiments. While we claim that the approach presented in Subsection 4.2 captures reasonably well individually rational behaviour for isolated trips, it is a matter of fact that most people travel repeatedly between specific locations (e.g., commuters). Driver agents could take advantage of this fact just as human drivers do, when they implicitly use historical information and past experiences to update the likelihood of selecting a specific route at a certain time [9]. Furthermore, a more sophisticated driver agent model should go beyond route choice and include, for instance, decisions on departure times [14].

Finally, in future work we will compare our management policy, based on one-to-many (combinatorial) auctions, to other economic models. For example, the market could be regulated by a continuous double auction, where many sellers (i.e., the intersection managers) place their sell-bids, and many buyers (i.e., the driver agents) submit their buy-bids, and the market continuously clears when a match between sell-bids and buy-bids is found. Also bargaining could be easily implemented in our scenario, with driver agents and intersection managers negotiating and agreeing on a mutually acceptable price.

## 6. ACKNOWLEDGMENTS

This research was partially supported by the Spanish Ministry of Science and Innovation through the project "AT" (CONSOLIDER CSD2007-0022, INGENIO 2010).

## 7. REFERENCES


[1] A. Bazzan and F. Klügl *Multi-agent Architectures for Traffic and Transportation Engineering* IGI-Global, 2009.

[2] J. Q. Cheng and M. P. Wellman *The WALRAS Algorithm: A Convergent Distributed Implementation of General Equilibrium Outcomes* Computational Economics, vol. 12, no. 1, pp. 1-24, 1998.

[3] B. Codenotti and S. Pemmaraju and K. Varadarajan *The computation of market equilibria.* SIGACT News, vol. 35, no. 4, pp. 23-37, ACM, 2004.

[4] K. M. Dresner and P. Stone. *A Multiagent Approach to Autonomous Intersection Management.* Journal of Artificial Intelligence Research, no. 31, pp. 591-656, 2008.

[5] S. Halle and B. Chaib-draa. *Collaborative driving system based on multiagent modelling and simulations.* Transportation Research Part C, no. 13, pp. 591-656, 2005.

[6] D. A. Hensher and C. Sullivan. *Willingness to pay for road curviness and road type.* Transportation Research Part D - Transport and Environment, vol. 8, no. 2, pp. 139-155, 2003.

[7] H. H. Hoos and C. Boutilier *Solving Combinatorial Auctions using Stochastic Local Search.* Proceedings of the 17th National Conference on Artificial Intelligence, pp. 22-29, AAAI Press/MIT Press, 2000

[8] R. T. van Katwijk and P. van Koningsbruggen and B. De Schutter and J. Hellendoorn. *A Test Bed for Multi-Agent Control Systems in Road Traffic Management.* Applications of Agent Technology in Traffic and Transportation, pp. 113-131, Birkhäuser, 2005.

[9] F. Klügl and A. Bazzan. *Route decision behaviour in a commuting scenario.* Journal of Artificial Societies and Social Simulation, vol. 7, no. 1, 2004.

[10] L. Kuyer and S. Whiteson and B. Bakker and N. Vlassis. *Multiagent Reinforcement Learning for Urban Traffic Control using Coordination Graphs.* Proceedings of the 19th European Conference on Machine Learning, pp. 656-671, 2008.

[11] K. Leyton-Brown and Y. Shoham and M. Tennenholtz. *An Algorithm for Multi-Unit Combinatorial Auctions.* Proceedings of the 17th National Conference on Artificial Intelligence, pp. 56-61, AAAI Press/MIT Press, 2000.

[12] H. Lieu. *Traffic-flow theory.* Federal Highway Administration, US Dep. of Transportation, 1999.

[13] K. Nagel and M. Schreckenberg. *A Cellular Automaton Model for Freeway Traffic.* J. Phys. I France 2, pp. 2221-2229, 1992.

[14] R. Rossetti and R. Liu. *A Dynamic Network Simulation Model Based on Multi-Agent Systems.* Applications of Agent Technology in Traffic and Transportation, pp. 181-192, Birkhäuser, 2005.

[15] T. Sandholm. *Algorithm for optimal winner determination in combinatorial auctions.* Artificial Intelligence, pp. 1-54, 2002.

[16] H. Schepperle and K. Bohm. *Agent-Based Traffic Control Using Auctions.* Cooperative Information Agents XI, LNCS 4676, pp. 119-133, 2007.

[17] M. Vasirani and S. Ossowski. *A market-inspired approach to reservation-based urban road traffic management.* Proceedings of the 8th International Joint Conference on Autonomous Agents and Multi-Agent Systems, pp. 617-624, 2009.

[18] M. Vasirani and S. Ossowski. *Evaluating Policies for Reservation-Based Intersection Control.* Proceedings of the 14th Portuguese Conference on Artificial Intelligence (EPIA'09), 2009.

[19] M. Wiering. *Multi-agent reinforcement learning for traffic light control* Proceedings of the 17th European Conference on Machine Learning, pp. 1151-1158, 2000.

[20] P. R. Wurman and M. P. Wellman and W. E. Walsh. *A Parametrization of the Auction Design Space.* Games and Economic Behavior, vol. 35, no. 1-2, pp. 304-338, 2001.

[21] X. Yang and J. Liu and F. Zhao and N. H. Vaidya. *A Vehicle-to-Vehicle Communication Protocol for Cooperative Collision Warning.* Proceedings of the 1st International Conference on Mobile and Ubiquitous Systems, pp. 114-123, IEEE Computer Society, 2004.

# A Surprise-based Selective Attention Agent for Travel Information

Luis Macedo
Centre for Informatics and Systems of the University of Coimbra
Polo II, Pinhal de Marrocos
3030 Coimbra, Portugal
macedo@dei.uc.pt

## ABSTRACT

This paper describes an agent that can be integrated in travel information systems so that these provide only the relevant/interesting travel information for travelers, preventing these from a superabundance of information and unnecessary interruptions. To do that the agent includes a surprise-based artificial selective attention mechanism grounded on psychological and neuroscience theories of selective attention and surprise which defend that surprise plays an undeniable role on attention focus. Our claim is that only travel information that diverges from the norm or is unfamiliar to the traveler should be considered relevant and therefore delivered to the traveler. We describe the architecture of the surprise-based selective attention agent and illustrate its critical role in an en-route travel information system.

## Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous

## General Terms

Experimentation

## Keywords

Filtering travel information, Selective attention, Surprise, ATIS, BDI agents

## 1. INTRODUCTION

Typically, travel information breaks down into two categories: static information, which is known in advance and changes infrequently, and real-time, dynamic information, which changes frequently. Static information includes planned construction and maintenance, special events, tolls and payment options, transit schedules and fares, intermodal connections, commercial vehicle regulations, listings of roadside services and attractions, maps and navigational instructions, and historical travel times by location and time of day, day of the week and season. Real-time information includes roadway conditions, including congestion and incident information which change minute-by-minute, alternate routes which can vary depending on the degree of congestion, whether transit vehicles are on schedule, the availability of spaces on parking lots, the identification of the next stop on a train or bus, the location or arrival time of the next train or bus, and travel time to a destination which can also vary depending on the time of day.

Advanced Travel Information Systems (ATIS) are designed to assist travelers in making pre-trip and en-route travel decisions by providing them pre-trip and en-route information. Pre-trip information is to inform travelers of traffic and transit conditions before they select a route, mode, departure time, or decide whether to make a trip. En-route information provides drivers information pertaining to traffic conditions, incidents, construction, transit schedules, weather conditions, hazardous road conditions, and recommended safe speeds while en-route. This information allows the drivers for instance to select the route which is best for them. Information can be provided while en-route by variable message signs, commercial radio, highway advisory radio, personal communication devices (e.g., cellular telephones, Personal Digital Assistants — PDAs, Smartphones) or in-vehicle navigational systems.

With wireless ATIS, the historic distinction between pre-trip and en-route information is starting to blur. Travelers are increasingly able to receive information, often in real or nearly real time, both before and during their trips because of the existence of all those mobile devices. The new wireless and web technologies are used both to gather traffic information (e.g., cell-phone probes, incident reports by cell phone users, GPS (Global Positioning System) / GIS (Geographic Information Systems) tracking for incident management) and disseminate it (e.g., Internet postings of up-to-date transit schedules, advice issued through on-board navigation systems, advisory services delivered through mobile phones, PDAs or Smartphones).

However, while these information systems can undoubtedly help humans perform better in these complex traveling scenarios, if the amount of information achieves a level that is unhandled, instead of being beneficial, it is a problem. Moreover, with the expected increase in the number of these travel information systems, in the number of the information technologies used to disseminate information and the countless kinds of information provided, this may become even worse. Humans will be continuously receiving a superabundance of information which they cannot handle by themselves. Although, evolution already provided humans with the selective attention components that indicate which few aspects of the world are significant to the particular problems at hand, the amount of information received by those selective attention components may be itself a problem and compromise agents' performance. This is even more problematic because most of the time this information is provided in a way that affects especially the high level natural selective attention, which is involved in strategic cognitive choices such as the preference or shift of a task or activity over another. This means that humans might have to interrupt whatever they are doing to deal with the information provided by those information systems. This phenomena is sometimes referred as ”Interruption overload” [31] and is especially problematic (or dangerous) if the human agent is performing critical tasks like driving a car. Actually, there is evidence indicating that those devices are the cause of many vehicle accidents [50, 53].

Given this wealth of information coupled with human real-time multi-task processing constraints, incorporating selective attention mechanisms in devices is a fundamental strategy to any chance of success, since this would decrease the number of interruptions. Moreover, it is contended that while many traveler information systems are innovative and make use of cutting edge technologies, they lack real machine intelligence and therefore may be limited in their ability to service the traveling public over the long-run. On the one hand, a wave of technological developments, in particular the increasing deployment of GIS and, on the other hand, the introduction and rapid market penetration of mobile devices such as cell phones boosted the development of ATIS towards what has been termed Intelligent Traveler Information Systems (ITIS) [1], in which artificial intelligence techniques are drawn upon to create systems capable of providing travelers with more personalized planning assistance.

Selective attention, the capability exhibited by humans for selecting the relevant portions of information from the environment, has been thoroughly researched over the last 100 years in psychology and more recently in neuroscience (e.g., [14, 54]). It is thought to be necessary because there are too many things in the environment to perceive and respond to at once. However, at present there is no general theory of selective attention. Instead there are specific theories for specific tasks, tasks such as orienting, visual search, filtering, multiple action monitoring (dual task), and multiple object tracking.

It is generally agreed that surprise and curiosity/interest play an essential role in selective attention [5, 4, 3, 2, 6, 14, 29, 32, 43, 42]. In fact situations that include novelty, incongruity, unpredictability, surprise, uncertainty, change, challenging and complexity certainly demands greater attention than a stimulus distinguished by none of these properties. Moreover, these properties are also those assigned to situations that cause curiosity [5, 4, 3, 2, 6, 29]. .

The computational models of surprise proposed by Itti and Baldi [13, 12, 34] quantify low-level surprise visual stimuli, and at this point does not account for high-level or cognitive beliefs of human observers. Both approaches focus on the role of surprise in visual attention (the perception of objects, movements, or scenes), and both are mainly concerned with the detection of unexpected events and the computation of surprise intensity. For example, central to Itti and Baldi's surprise model is the proposal to compute surprise intensity as the distance (measured by the Kullback-Leibler divergence) between the prior probability distribution over a set of hypotheses and the posterior distribution resulting from the Bayesian updating of the prior distribution on the basis of new information.

A similar approach has already been proposed by Schmidhuber in the context of reinforcement learning and neural nets. Schmidhuber [43, 42] used artificial curiosity as a reward that enables an artificial agent to acquire quickly learning examples from the environment during its exploratory activity. Oudeyer [33] used artificial curiosity as an intrinsic motivation for improving the learning progress of a developmental robot. Both computational models of curiosity subsume, to some extent, models of surprise in that curiosity intensity relies on error prediction. However, much like Itti and Baldi, and Peters' computational models of surprise, Schmidhuber and Oudeyer's computational models of curiosity are applied only to low level or raw sensorial data. Although some surprise theorists (e.g., [45]) have claimed that surprise can also be elicited at ”lower” levels of representation than the propositional level, specifically by perceptual mismatch, it is doubtful whether perceptual mismatch per se causes the experience of surprise in humans [30]. As argued by Losee [19], there are more complex kinds of information such as beliefs. According to cognitive theories, these mental states are actually the most important information inputs for those cognitive processes such as surprise and curiosity. Therefore, good models of surprise and curiosity should take this higher level kind of information into account.

Opposed to these approaches relying on low-level, raw information, Macedo, Reisenzein and Cardoso [21, 28] and Lorini and Castelfranchi [18, 17, 16] proposed, independently, computational models of surprise that are based on the mechanism that compares newly acquired beliefs to pre-existing beliefs. Both models of artificial surprise were influenced by psychological theories of surprise (e.g., [29]), and both seek to capture essential aspects of human surprise (see [27] for a comparison of both models). In agreement with most theories of human surprise, both models of artificial surprise conceptualize surprise as a fundamentally expectation- or belief-based cognitive phenomenon, that is, as a reaction to the disconfirmation of expectations or, more generally, beliefs. Furthermore, in both models, beliefs are understood as propositional attitudes (e.g., [44]), and a quantitative belief concept (subjective probability) is used. Both artificial surprise models draw a distinction between two main kinds of expectations or beliefs whose disconfirmation causes surprise (see also [32]): Active versus passive expectations. Although Macedo and Cardoso initially used the same surprise intensity function, according to which the intensity of surprise about an event is proportional to its unexpectedness, Macedo, Reisenzein and Cardoso subsequently opted for a ”contrast model” of surprise intensity. This model assumes that the intensity of surprise about an event reflects its probability difference to the contextually most expected event (see also, [51]).

The model of surprise developed by Macedo, Reisenzein and Cardoso is combined with another for curiosity to drive the exploratory behaviour of a Belief-Desire-Intention (BDI) artificial agent. Macedo [20] stated a clear distinction between surprise and curiosity, although according to Meyer et al surprise elicits curiosity. However, the actual Macedo's computational model of curiosity is based solely on the idea that novelty and uncertainty (measured by entropy) elicit curiosity/interest (e.g., [5, 20]). According to psychological theories of curiosity [4, 49, 48, 47], this model as well as those of Shmidhuber or Oudeyer are incomplete in that they don't take into account other variables such as complexity.

In spite of the importance of selective attention in travel information systems such as in driving [50, 53], to our knowledge, only [11] applied a surprise-based mechanism for filtering information. However, the computational model of surprise has no apparent relation to human surprise, which we think it is very important in that we are trying to substitute human attention.

In this paper we describe the integration into the ATIS of such artificial attention mechanism focusing on its surprise-based component, at least at the level of personal devices so that only relevant travel information for the task their human masters are carrying out is selected and communicated to them. Our approach relies on the psychological and neuroscience studies about selective attention, whose main aspects were already considered in the computational models of surprise and curiosity proposed by Macedo [20]. In fact, those models already capture the variables of unexpectedness, unpredictability, novelty, and uncertainty. Specifically, we adopt, adapt and improve those computational models of surprise and curiosity developed by Macedo and Cardoso [20, 25, 24, 23, 22, 21, 26, 27] and, in addition, include also an utility metric, so that only the information that is both curious and useful is selected and transmitted to the human travelers. In order to assess the effectiveness of the surprise-based selective mechanism, we compare the selections made by the devices and by humans under similar circumstances.

The next section describes the computational model of surprise-based selective attention and outlines the architecture of the agent in which it is integrated. This will be followed by presenting the application of this kind of agents in travel systems. We then describe an exploratory study about the contribute of the surprise-based selective attention agent to solve the travel information overload of its master. Finally, after a short discussion, some conclusions are presented and suggestions for further work are made.

## 2. A COGNITIVE COMPUTATIONAL MODEL OF SURPRISE-BASED SELECTIVE ATTENTION

Selective attention may be defined as the cognitive process of selective allocation of processing resources (focus of the senses, etc.) on relevant, important or interesting information of the (external or internal) environment while ignoring other less relevant information. The issue is how to measure the relevance of information. What makes something interesting? In cognitive science, attentional focus is linked with expectation generation and failure, i.e., with surprise [32]. Therefore, it is reasonable to consider that any model of selective attention should rely on a cognitive model of surprise. However, surprise is not enough. Happiness/pleasantness may also play also a fundamental role on attention [49, 48, 47]. According to cognitive theories of emotion and specifically to belief-desire theories of emotion [39], happiness is directly related to congruence and relatedness between new information and the intentions or the motives/desires of a human agent. For this reason, the system must also incorporate a measure of the expected reward or utility of the information for a specific human agent, based on her/him particular intentions and desires at hand. Other variables such as novelty (different, unfamiliar), complexity (hard to process, challenging, mysterious), uncertainty, coping potential [5, 4, 3, 2, 6, 49, 48, 47] (according to previous studies, there is evidence indicating that these variables elicit curiosity/interest [5, 4, 3, 2, 6, 14, 49, 48, 47]), might also been taken into account.

In order to accomplish all those requirements, we developed an architecture for a personalized, selective attention mechanism (see Figure 1). We assume this mechanism is incorporated in an agent which interacts with the external world receiving from it information through the senses and outputs actions through their effectors. We also assume the agent is a BDI agent [37, 36, 7], exhibiting a knowledge or belief container, a module of feelings, as well as intentions and desires. In addition, we also assume the agent contains other resources for the purpose of reasoning, decision-making and communication. The first of the steps is concerned with getting percepts (module 1 in Figure 1). The second is the computation of the current world state (module 2 in Figure 1). This is performed by generating expectations or assumptions for the gaps of the environment information provided by the sensors based on the knowledge stored in memory. We assume that each input information resulting from this process goes through several sub-selective attention devices, each one evaluating information according to a certain dimension such as surprise (module 4 in Figure 1), novelty (module 5 in Figure 1), uncertainty (module 6 in Figure 1), complexity (module 7 in Figure 1), coping potential (module 8 in Figure 1), and pleasantness (i.e., utility or congruence to agent's goals and desires — happiness; relatedness to agent's goals and desires) (module 9 in Figure 1) taking into account some knowledge container (memory — preexisting information, that should reflect the human information) (module 10 in Figure 1), and the intentions and desires (motives — module 12 in Figure 1). The values of surprise, curiosity (includes novelty and uncertainty), happiness, etc. are computed by the feeling module (module 11 in Figure 1). There is a decision-making module (module 13 in Figure 1) that takes the values computed by those sub-selective attention modules into account and computes an overall relevance/interesting value for each input information. Then, this module of decision-making selects the higher relevant information and allocates appropriately resources (reasoning, processing, displaying, communication resources, etc.) (module 14 in Figure 1) to deal with it. In this sense, the selective attention mechanism is on the basis of other cognitive abilities of the agent in that it decides in which information those other cognitive abilities should focus.

In this paper we will focus on the surprise-based selective attention mechanism. We claim that any computational model of selective attention should capture a cognitive model of surprise. We will describe in more detail the surprise-based selective attention module as well as all those secondary modules that surrounds (serves) it.

The process of making the right decision depends heavily on a good model of the environment that surrounds agents. This is also true for deciding in which information should the agent focus. Unfortunately, the real world is not crystal clear to agents. Agents almost never have access to the whole environment, mainly because of the incompleteness and incorrectness of their perceptual and understanding components. In fact, it is too much work to obtain all the information from a complex and dynamic world, and it is quite likely that the accessible information suffers distortions. Nevertheless, since the success of agents depends heavily on the completeness of the information of the state of the world, they have to pursue alternatives to construct good models of the world even (and especially) when this is uncertain. According to psychologists, cognitive scientists, and ethologists [15, 35], humans and, in general, animals attempt to overcome this limitation through the generation of assumptions or expectations to fill in gaps in the present observational information. Note, however, that not all those expectations are made explicit. However, the reasoning of the agent may be improved if its model of the world also contains a good model of the future worlds. In this case, the process cannot be confined to filling in gaps in the information provided by perception because there is no information at all of those worlds. In order to overcome this limitation, agents also exhibit the ability to make predictions about future states of the world, taking the present world and inference processes into account (module 2 in Figure 1). When the missing information, either of the present state of the world or of the future states of the world, becomes known to the agent, there may be an inconsistency or conflict between it and the assumptions or expectations that the agent has. As defended by Reisenzein [38], Gärdenfors [10], Ortony and Partridge [32], etc., the result of this inconsistency gives rise to surprise which in our model of selective attention and according to previous studies plays a central role in selective attention. It also gives rise to the process of updating beliefs, called belief revision (e.g., [9]).

Following the pluralist view of motivation (e.g.: [40]) , the module of basic desires (basic motivations/motives) (module 12 in Figure 1) contains a set of basic desires that drive the behaviour of the agent by guiding the agent to reduce or to maximize a particular feeling [24]. In this paper we focus on agents that exhibit the basic desire of surprise that directs the agent to feel surprise, i.e., to satisfy that basic desire the agent selects focusing attention on aspects of the world that make it feel surprise.

The module of feelings (module 11 in Figure 1) receives information about a state of the environment and outputs the intensities of feelings. Following Clore [8], we include in this module affective, cognitive, and bodily feelings. The latter two categories are merged to form the category of non affective feelings. This means that this module is much broader than a module of emotion that could be considered. Feelings are of primary relevance to influence the behavior of an agent, because computing their intensity the agent measures the degree to which the basic desires are fulfilled. In this paper, we highlight the feeling of surprise. We adopted Macedo, Cardoso and Reisenzein computational model of surprise [21, 28]. In contrary to other computational models such as Itti and Baldi's which are appropriate solely to the lower level of selective attention required in raw sensorial attention, this computational model was empirically tested against human surprise ratings and fits well human surprise and therefore it is appropriate for reasoning about non-raw data such as high level, cognitive beliefs and knowledge. It will ensure that given some information and the agent's belief store, only that information that is unexpected or unpredictable will be object of alert. Note, however, that Lorini and Castelfranchi's surprise model is also appropriate to be incorporated in this agent's architecture. Macedo, Cardoso and Reisenzein computational model of surprise suggests that the intensity of surprise about an event $E_g$, from a set of mutually exclusive events $E_1, E_2, \ldots, E_m$, is a nonlinear function of the difference, or contrast, between its probability and the probability of the highest expected event $E_h$ in the set of mutually exclusive events $E_1, E_2, \ldots, E_m$.

*Definition 1.* Let $E = E_1, E_2, \ldots, E_m$ be a set of mutually exclusive events. Let $E_h$ be the highest expected event from $E$. The intensity of surprise about an event $E_g$ from $E$ is given by:

$$Surprise(E_g) = \log(1 + P(E_h) - P(E_g)) \qquad (1)$$

The probability difference between $P(E_h)$ and $P(E_g)$ can be interpreted as the amount by which the probability of $E_g$ would have to be increased for $E_g$ to become unsurprising. The formula implies that, in each set of mutually exclusive events, there is always at least one event whose occurrence is unsurprising, namely, $E_h$.

The memory of the agent (module 10 in Figure 1) stores information (beliefs) about the world. This information includes the configuration of the surrounding world such as the position of the entities (objects and other animated agents) that inhabit it, the description of these entities themselves, and the descriptions of plans executed by those entities. The information is stored in several memory components. There is a metric (grid-based) map to spatially model the surrounding physical environment of the agent. Descriptions of entities (physical structure and function) and plans are stored both in the episodic memory and in the semantic memory (see [20] for more details).

# 3. SELECTIVE ATTENTION TO TRAVEL INFORMATION

An ATIS database provides information of various types to the travelers. On the other hand, data is continuously collected from sources such as travelers, traffic sensors, and weather service. Selective attention agents may be integrated at two levels (see Figure 2): in personal devices to act as personal assistant selective attention agents, and in the ATIS database itself. In personal devices the goal of the selective attention agents is to avoid unnecessary interruptions to their users by enabling that only interesting information is provided to them. In the ATIS database the goal is to ensure that irrelevant information is not stored in the ATIS database.

Let us illustrate these two roles of selective attention agents, in this case based solely on surprise. Suppose that a traveler has the following expectations for the traffic conditions of a certain road, for a certain time: 1% of probability of "good traffic conditions" (event $E_1$), 9% of probability of "moderate traffic congestion" (event $E_2$), and 90% of probability of "excessive traffic congestion" (event $E_3$). Should the traveler be alerted if the real time traffic conditions are bad? Suppose also his/her surprise-based, selective attention, personal assistant agent is setup with the same set of expectations. This surprise-based selective attention agent customized to produce an alert only when the surprise value of information is above the 90% level wouldn't provide that information to the traveler. Actually, according to Equation 1, the surprise value of $E_3$="excessive traffic" is:

$$\begin{aligned} Surprise(E_3) &= \log(1 + P(E_3) - P(E_3)) \\ &= \log(1 + 0.9 - 0.9) = 0 \end{aligned}$$

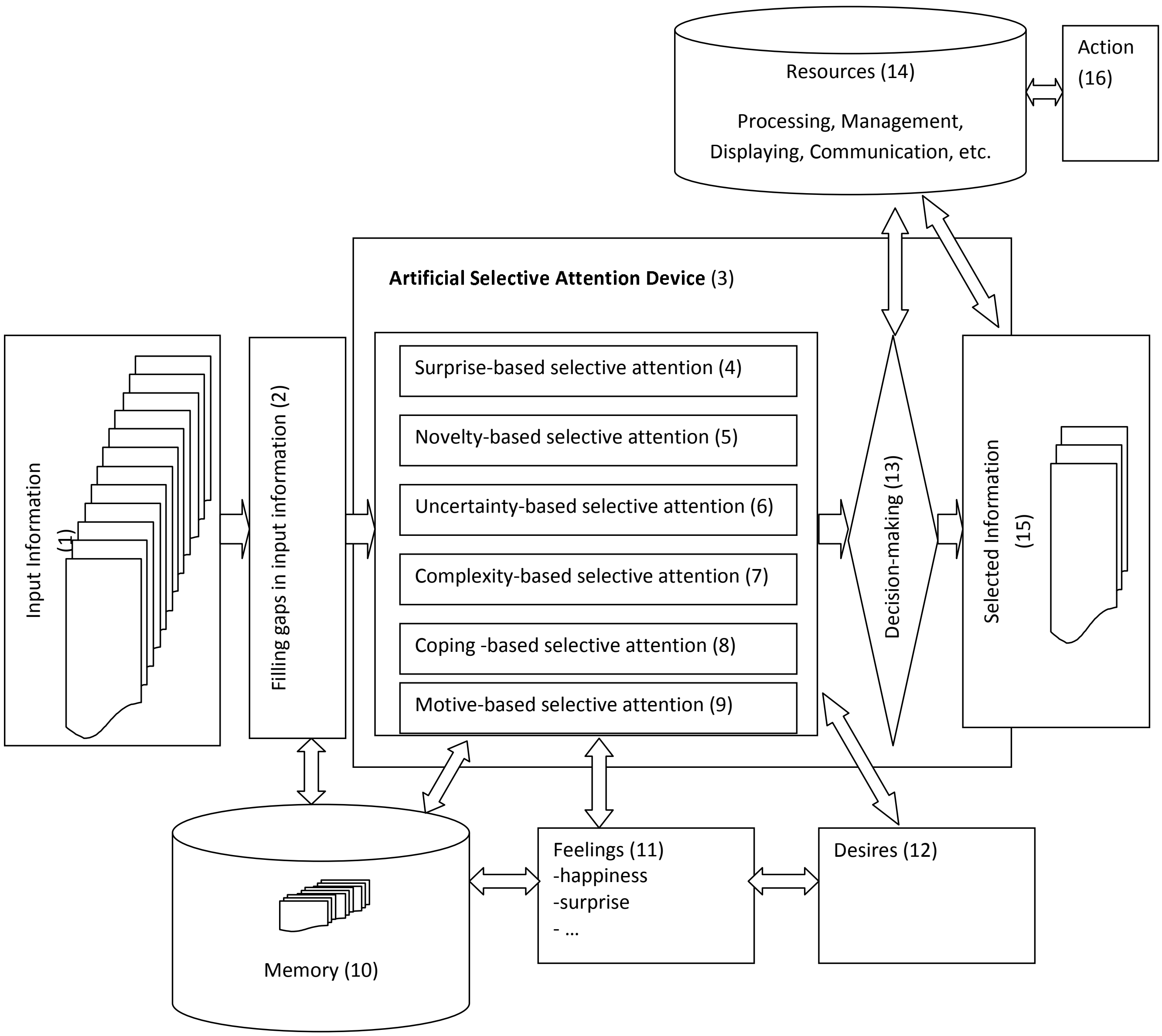




Figure 1: Architecture of an artificial selective attention agent.

Notice that in this case the event $E_3$="excessive traffic congestion" is the one with the highest probability in the set of mutually exclusive events. If the traveler beliefs strongly in that why should he/she be notified?! If the real time traffic condition is $E_2$="moderate traffic congestion", the surprise value now is:

$$\begin{aligned} Surprise(E_2) &= \log(1 + P(E_3) - P(E_2)) \\ &= \log(1 + 0.9 - 0.09) = 0.855 \end{aligned}$$

which is still below the level of triggering an alert (90%) and therefore no alert is produced. However, if the real time traffic condition is $E_1$="good traffic condition", the surprise value now is:

$$\begin{aligned} Surprise(E_1) &= \log(1 + P(E_3) - P(E_1)) \\ &= \log(1 + 0.9 - 0.01) = 0.918 \end{aligned}$$

which is enough to trigger an alert. Notice that the level of triggering an alert is customized and therefore the accuracy of the selective attention agent depends on it. For instance, if the level was 85% the agent would produce an alert in the second situation which for some people might be a reasonable choice. This example is about traffic conditions information, but it is worth of notice that it might be applied also to any other kind of travel information such as GPS traces, points of interest, weather conditions and road conditions.

Using a surprise-based selective agent in the ATIS, only the collected information that is above a specified level of surprise for the surprise-based selective attention agent in the ATIS would be considered relevant to be added to the ATIS database. Consider that this database contains the information that the traffic conditions of a certain road at a certain time are 1% of the times good, 9% moderate and 90% bad. Suppose also that only information with a surprise value above 90% is allowed to be added to the database. If someone submits the information that the traffic conditions are moderate or bad, this information would not be added to the database. However, if someone submits the information that the traffic condition is good, this would be worth of addition to the database, because it is a less familiar situation.

## 4. EXPERIMENT

We did an exploratory study in order to compare the relevance value computed by the selective attention agent and the relevance value computed by humans. While the relevance value rated by humans is of subjective nature, the relevance value computed by the artificial selective attention agents is based rigorously on expectations computed from statistical data collected from previous traffic situations in the past 30 days for a certain place, all at the same time of the day. The artificial agent used Equation 1 to compute the relevance value which in this case is confined to surprise. We select a street from a city (Bissaya Barreto Avenue, in Coimbra, Portugal) and configured a selective attention agent to provide real time information about the traffic conditions in that street to 5 volunteer travelers whose path include that street. We collect the relevance the travelers assign to the information the agent delivered during 10 days at the same time (9h:00m) and always concerning the same street. The real time traffic conditions of the 10 days of the experiment are presented in Table 1. In addition, after the trip, the information the agent didn't delivered because its surprise value was below the triggering level of alert was shown to the travelers and these were asked to rate the relevance they would assign that information if it was delivered.

**Table 1: Traffic conditions of the 10 days of the experiment.**

| Day | Traffic condition |
|---|---|
| 1 | Good |
| 2 | Excessive |
| 3 | Excessive |
| 4 | Good |
| 5 | Excessive |
| 6 | Moderate |
| 7 | Good |
| 8 | Excessive |
| 9 | Moderate |
| 10 | Excessive |

Figure 3 shows the comparison of the relevance value rated by humans with those computed by the agent based solely on surprise. As it can be seen, the correlation is very high (0.99), but still there are some discrepancies. For instance, we noticed that humans assign total relevance for information with a surprise value above about 80%. Moreover, we noticed that some situations in which the agent didn't delivered information, the humans rated a low (but different from 0) relevance value for that information. However, they didn't consider that it would be worth of delivery. Although not shown in the chart, the experiment shows that using the 90% level of triggering an alert the agent failed twice (day 6 and day 9) according to the traveler opinions. In those two cases, they say that the agent should have provided information. However, when we decrease the triggering level to 80% the performance of the agent was very good with no incorrect decisions.

## 5. DISCUSSION AND CONCLUSIONS

We presented an approach to deal with travel information overload relying on a surprise-based artificial selective attention mechanism that may be integrated in travel information technologies. The exploratory experimental results indicate that the mechanism performs well, contributing to the decrease of interruptions when driving. However, the performance of the selective attention mechanism depends on several factors such as the reference class [41] considered to computed the expectations and the triggering level of alert. With respect to the reference class, it is worth of notice that the artificial selective attention agent computes the degree of belief based on a frequentist approach to probability, contrasting with the humans's subjective expectations. For instance, to compute the probability of bad traffic conditions for a certain place, the agent might take several options such as taking all the traffic history of that place into account for the computation of the probability, or restricting these data to those situations that happened at that place at a certain season, day of the week or even specifically time of the day.

As demonstrated in the previous two sections, the trig-

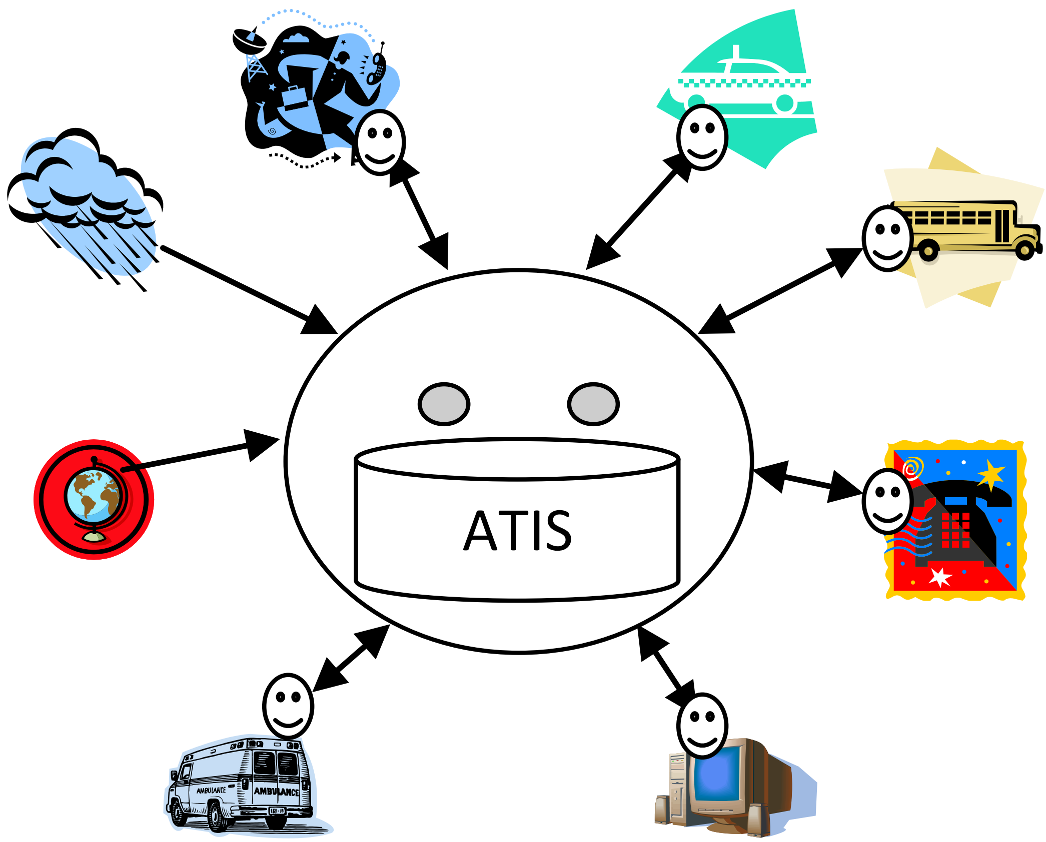


**Figure 2: Selective attention agents, information sources and information dissemination of an ATIS; the smiley faces represent the selective attention agents. The ATIS itself is embedded into a selective attention agent.**

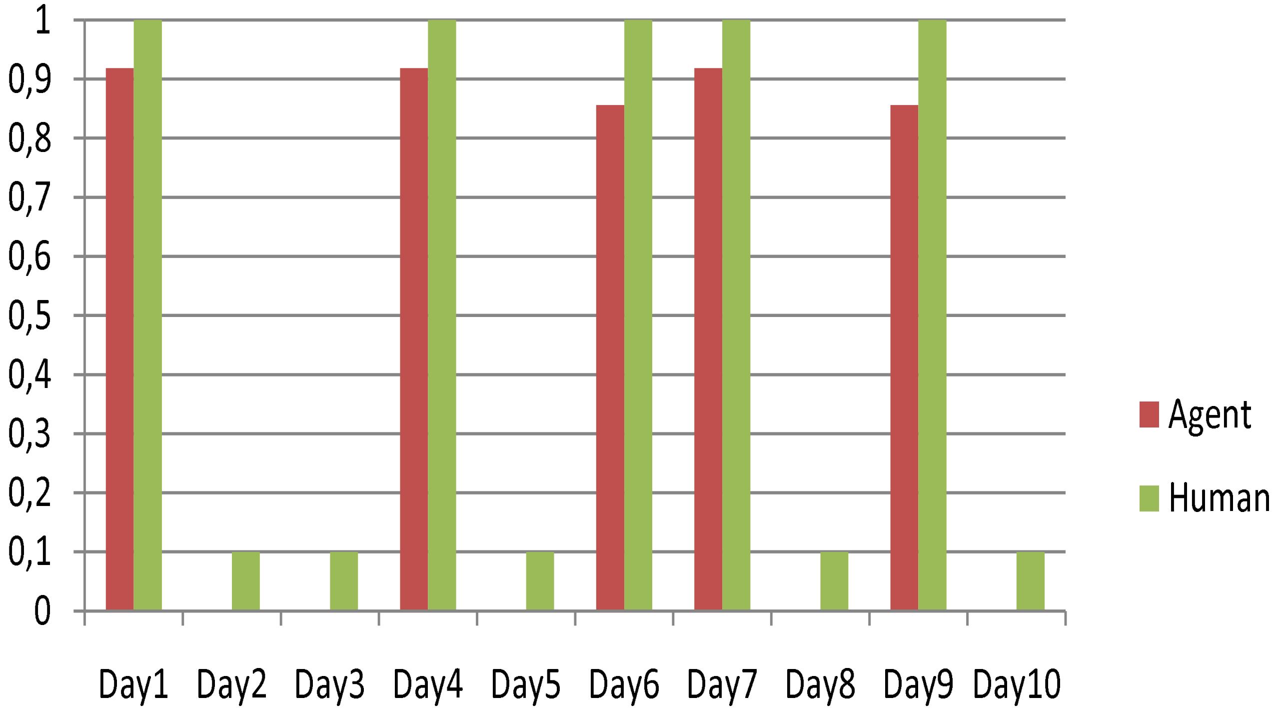


**Figure 3: Comparison between the relevance rated by humans and the surprise-based relevance computed by the selective attention agent.**

gering level of alert has direct influence on the performance of the agent. However, if the agent is to act as a personal agent the triggering level may change from human to human. Therefore, a pilot experiment should be carried out to determine the the most correct triggering level on average for the travelers.

The experiment that we carried out needs further improvements. In order to assess the significance of the results, the number of travelers and the number of locations (streets, roundabouts, etc) involved in the experiment should be increased. Furthermore, in order to generalize the evidence for the role of the surprise-based selective attention mechanism on travel information, several kinds of travel information should be considered in the experiments such as information about roadside services and attractions, maps and navigational instructions, roadway conditions (including congestion, incidents, construction and other hazardous road conditions), weather conditions, alternate routes which can vary depending on the degree of roadway conditions, whether transit vehicles are on schedule, the availability of spaces on parking lots, the identification of the next stop on a train or bus, the location or arrival time of the next train or bus, and travel time to a destination.

As mentioned above the surprise-based selective attention mechanism allows the agent to alert for travel information that is unexpected. Although the variables of novelty, unexpectedness, complexity and uncertainty are quite related, and sometimes used as synonyms in the literature, we defend that they are different and therefore having different and indispensable roles in the selective attention mechanism, and to some extent complementing each other. While novelty means new information, uncertainty means that new information will probably be acquired. Information is a decrease in uncertainty which, according to information theory, is measured by entropy. New information is surprising, but there might exist information that, although it is not novel, it is surprising. It is also worth of notice that the definition of surprise adopted in this paper is different from the notion of *surprisal* from information theory [46, 52]. To illustrate the difference between the surprise-based and these related selective attention mechanisms consider the following example. Suppose that an agent has the following expectations for the traffic conditions of a certain road, for a certain time: 1% of probability of "good traffic conditions" (event $E_1$), 9% of probability of "moderate traffic congestion" (event $E_2$), and 90% of probability of "excessive traffic congestion" (event $E_3$). In this case, if the agent receives the information that the traffic conditions of that road, at that time, are good, this information is not new since it already happened in the past according to the agent's memory. However, this information is surprising (actually very surprising: 91.8% of surprise), but different from its surprisal value (6.64).

As mentioned above a selective attention agent may be located either at the personal devices or at the ATIS itself. It is worth of notice that when it is located at the personal devices, even though the agent may not communicate to their master pieces of information below the triggering level of alert, this information is stored in their memory so that it can be taken into account to update the probability distributions in memory. This way, this information influences the computation of expectations in the future and therefore the future selections. On the contrary, the selective attention agent located at the ATIS filters the information below a specified level of relevance, preventing its storage.

Another important issue that influences the performance of the agent is whether to consider expectations of the humans or those computed from statistical data. The experiment was done with the latter method. However, for the purpose of assessing the performance of the selective attention agent it might be more appropriate to give the artificial agents the same expectations of humans so that they act under the same conditions. Nevertheless, in terms of practical application it makes more sense to make the agent compute its own expectations.

The experiment described in this paper makes the simplification that the events are all equally related to one's intentions in that all the travelers include in their trajectories the street that was chosen for the study. However, in reality things doesn't happen this way. The following example illustrates this point. Suppose someone is driving and intend to go to a certain place. Suppose that he/she is informed by his/her traffic information system that there is a traffic congestion in a street which is part of the path that he/she is going to follow. Suppose also that he/she receives a similar information but with respect to another street which is not included in his/her trajectory. He/she would be more attracted by the former information than the latter. Assume this two pieces of information are not new, not surprising (those streets are usually congested) or equally surprising. The major difference between these pieces of information is that the former is related to his/her intentions/goals and the latter is not. Therefore, in addition to those very related sub-selective attention mechanisms based on surprise, novelty, complexity and entropy, the pleasantness-based selective attention mechanism plays a central role in selective attention.

In spite of these illustrative examples explaining the difference between the roles of all those sub-selective attention mechanisms, further experiments should be carried out to assess the contribute of all of them to the the overall selective attention mechanism. A factorial experiment in which the several sub-selective attention mechanisms are the factors (the independent variables) should be done.

# 6. REFERENCES


[1] J. Adler and V. Blue. Toward the design of intelligent traveler information systems. *Transportation Research Journal Part C*, 6/3:157–172, 1998.

[2] D. Berlyne. Novelty and curiosity as determinants of exploratory behavior. *British Journal of Psychology*, 41(1):68–80, 1950.

[3] D. Berlyne. The arousal and satiation of perceptual curiosity in the rat. *Journal of Comparative and Physiological Psychology*, 48:238–246, 1955.

[4] D. Berlyne. *Conflict, arousal and curiosity.* McGraw-Hill, New York, 1960.

[5] D. Berlyne. Arousal and reinforcement. In D. Levine, editor, *Nebraska Symposium on Motivation*, volume 15, pages 1–110. University of Nebraska Press, Lincoln, Nebraska, 1967.

[6] D. E. Berlyne. *Aesthetics and psychobiology.* Appleton-Century-Crofts, New York, 1971.

[7] M. Bratman, D. Israel, and M. Pollack. Plans and resource-bounded practical reasoning. *Computational Intelligence*, 4(4):349–355, 1988.

[8] G. Clore. Cognitive phenomenology: Feelings and the construction of judgment. In L. Martin and A. Tesser, editors, *The Construction of Social Judgments*, pages 133–163. Lawrence Erlbaum Associates, Hillsdale, NJ, 1992.

[9] P. Gärdenfors. Belief revision: An introduction. In P. Gärdenfors, editor, *Belief Revision*, pages 1–20. Cambridge University Press, Cambridge, UK, 1992.

[10] P. Gärdenfors. The role of expectations in reasoning. In M. Masuch and L. Polos, editors, *Knowledge Representation and Reasoning Under Uncertainty*, pages 1–16. Springer-Verlag, Berlin, 1994.

[11] E. Horvitz, J. Apacible, R. Sarin, and L. Liao. Prediction, expectation, and surprise: Methods, designs, and study of a deployed traffic forecasting service. In *Proceedings of the Conference on Uncertainty and Artificial Intelligence 2005.* AUAI Press, 2005.

[12] L. Itti and P. Baldi. A surprising theory of attention. In *Proceedings of IEEE Workshop on Applied Imagery and Pattern Recognition.* 2004.

[13] L. Itti and P. Baldi. Bayesian surprise attracts human attention. *Advances in Neural Information Processing Systems (NIPS 2005)*, 19:1–8, 2006.

[14] D. Kahneman. *Attention and effort.* Prentice-Hall, Englewood Cliffs, NJ, 1973.

[15] C. Kline. *Observation-based expectation generation and response for behavior-based artificial creatures.* Msc thesis, Massachusetts Institute of Technology, 1999.

[16] E. Lorini and C. Castelfranchi. The role of epistemic actions in expectations. In *Proceedings of the Second Workshop of Anticipatory Behavior in Adaptive Learning Systems*, pages 62–71. Los Angels, USA, 2004.

[17] E. Lorini and C. Castelfranchi. The unexpected aspects of surprise. *International Journal of Pattern Recognition and Artificial Intelligence*, 20(6):817–835, 2006.

[18] E. Lorini and C. Castelfranchi. The cognitive structure of surprise: looking for basic principles. *Topoi: An International Review of Philosophy*, 26(1):133–149, 2007.

[19] R. Losee. A discipline independent definition of information. *J. of the American Society for Information Science*, 48(3):254–269, 1997.

[20] L. Macedo. *The Exploration of Unknown Environments by Affective Agents.* Phd thesis, University of Coimbra, 2007.

[21] L. Macedo and A. Cardoso. Modelling forms of surprise in an artificial agent. In J. Moore and K. Stenning, editors, *Proceedings of the 23rd Annual Conference of the Cognitive Science Society*, pages 588–593. Erlbaum, Edinburgh, Scotland, UK, 2001.

[22] L. Macedo and A. Cardoso. Assessing creativity: the importance of unexpected novelty. In *Proceedings of the ECAI'02 Workshop on Creative Systems: Aproaches to Creativity in AI and Cognitive Science,*, pages 31–37. University Claude Bernard - Lyon, Lyon, France, 2002.

[23] L. Macedo and A. Cardoso. A model for generating expectations: the bridge between memory and surprise. In C. Bento, A. Cardoso, and J. Gero,

editors, *Proceedings of the 3rd Workshop on Creative Systems: Approaches to Creativity in AI and Cognitive Science, International Joint Conference on Artificial Intelligence*, pages 3–11. IJCAI03, Acapulco, Mexico, 2003.

[24] L. Macedo and A. Cardoso. Exploration of unknown environments with motivational agents. In N. Jennings and M. Tambe, editors, *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 328 – 335. IEEE Computer Society, New York, 2004.

[25] L. Macedo and A. Cardoso. The role of surprise, curiosity and hunger on the exploration of unknown environments. In *Proceedings of the 12th Portuguese Conference on Artificial Intelligence*. Springer, Covilhã, Portugal, 2005.

[26] L. Macedo, A. Cardoso, and R. Reisenzein. A surprise-based agent. In R. Trappl, editor, *Proceedings of the 18th European Meeting on Cybernetics and Systems Research*, pages 583–588. Austrian Society for Cybernetic Studies, Vienna, Austria, 2006.

[27] L. Macedo, A. Cardoso, R. Reisenzein, E. Lorini, and C. Castelfranchi. Artificial surprise. In J. Vallverdú and D. Casacuberta, editors, *Handbook of Research on Synthetic Emotions and Sociable Robotics: New Applications in Affective Computing and Artificial Intelligence*, pages 267–291. IGI Global, Hershey: USA, 2009.

[28] L. Macedo, R. Reisenzein, and A. Cardoso. Modeling forms of surprise in artificial agents: empirical and theoretical study of surprise functions. In K. Forbus, D. Gentner, and T. Regier, editors, *Proceedings of the 26th Annual Conference of the Cognitive Science Society*, pages 873–878. Lawrence Erlbaum Associates, Inc., Chicago, Illinois, USA, 2004.

[29] W. . U. Meyer, R. Reisenzein, and A. Schützwohl. Towards a process analysis of emotions: The case of surprise. *Motivation and Emotion*, 21:251–274, 1997.

[30] M. Niepel. Independent manipulation of stimulus change and unexpectedness dissociates indices of the orienting response. *Psychophysiology*, 38:84–91, 2001.

[31] N. O'Connell. Interruption overload. *Strategic Direction*, 24(10):3–5, 2008.

[32] A. Ortony and D. Partridge. Surprisingness and expectation failure: what's the difference? In *Proceedings of the 10th International Joint Conference on Artificial Intelligence*, pages 106–8. Morgan Kaufmann, Milan, Italy, 1987.

[33] P. Oudeyer, F. Kaplan, and V. Hafner. Intrinsic motivation systems for autonomous mental development. *IEEE Transactions on Evolutionary Computation*, 11(2):265–286, 2007.

[34] M. Peters. Towards artificial forms of intelligence, creativity, and surprise. In *Proceedings of the Twentieth Annual Conference of the Cognitive Science Society*, pages 836–841. Erlbaum, Madison, Wisconsin, USA, 1998.

[35] J. Piaget. *The origins of intelligence in children.* International Universities Press, New York, 1952.

[36] A. Rao and M. Georgeff. Modeling rational agents within a bdi-architecture. In R. Fikes and E. Sandewall, editors, *Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning*, pages 473–484. Morgan Kaufmann Publishers, Inc., Cambridge, MA, USA, 1991.

[37] A. Rao and M. Georgeff. Bdi agents: from theory to practice. In *Proceedings of the First International Conference on Multiagent Systems*, pages 312–319. MIT Press, San Francisco, CA, USA, 1995.

[38] R. Reisenzein. The subjective experience of surprise. In H. Bless and J. Forgas, editors, *The message within: The role of subjective experience in social cognition and behavior.* Psychology Press, Philadelphia, PA, 2000.

[39] R. Reisenzein. Emotions as metarepresentational states of mind: Naturalizing the belief-desire theory of emotion. *Cognitive Systems Research*, 9, 2008.

[40] S. Reiss. *Who am I? The 16 basic desires that motivate our actions and define our personalities.* Berkley Books, New York, 2000.

[41] S. Russell and P. Norvig. *Artificial intelligence - a modern approach.* Prentice Hall, Englewood Cliffs, NJ, 1995.

[42] J. Schmidhuber. Curious model-building control systems. In *Proceedings of the International Conference on Neural Networks*, volume 2, pages 1458–1463. IEEE, Singapore, 1991.

[43] J. Schmidhuber. Developmental robotics, optimal artificial curiosity, creativity, music, and the fine arts. *Connection Science*, 18:173–187, 2006.

[44] J. Searle. *Intentionality.* Cambridge University Press, Cambridge, 1983.

[45] A. F. Shand. *The foundations of character.* Macmillan, London, 1914.

[46] C. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423 and 623–656, 1948.

[47] P. J. Silvia. What is interesting? exploring the appraisal structure of interest. *Emotion*, 5:89–102, 2005.

[48] P. J. Silvia. *Exploring the psychology of interest.* Oxford University Press, New York, 2006.

[49] P. J. Silvia. Interest - the curious emotion. *Current Directions in Psychological Science*, 17:57–60, 2008.

[50] D. Strayer, F. Drews, and W. Johnston. Cell phone-induced failures of visual attention during simulated driving. *Journal of Experimental Psychology: Applied*, 9(1):23–32, 2003.

[51] K. H. Teigen and G. B. Keren. Surprises: Low probabilities or high contrasts? *Cognition*, 87:55–71, 2003.

[52] M. Tribus. *Thermostatics and thermodynamics.* van Nostrand, Princeton, NJ, 1961.

[53] L. M. Trick, J. T. Enns, J. Mills, and J. Vavrik. Paying attention behind the wheel: A framework for studying the role of selective attention in driving. *Theoretical Issues in Ergonomic Science*, 5(5):385–424, 2004.

[54] R. D. Wright and L. M. Ward. *Orienting of Attention.* Oxford University Press, Oxford, UK, 2008.

# Language for Implementing Multiagent Transportation Applications

Mahdi Zargayouna
INRETS Institute,
Gretia Laboratory
Building “Descartes II”
2 rue de la Butte Verte
93160 Noisy le Grand Cedex
France
zargayouna@inrets.fr

Flavien Balbo
University of Paris Dauphine,
CNRS-Lamsade Laboratory
Place Maréchal de Lattre de Tassigny
75775 Paris Cedex 16
France
balbo@lamsade.dauphine.fr

Gérard Scemama
INRETS Institute,
Gretia Laboratory
Building “Descartes II”
2 rue de la Butte Verte
93160 Noisy le Grand Cedex
France
scemama@inrets.fr

## ABSTRACT

The multiagent paradigm is well suited to designing transportation applications. However, when implementing these applications in highly dynamic environments, it is very costly to use languages relying on channel-based communication. Data driven coordination languages rely on a shared space in which agents add, read and retrieve data. They are intuitively relevant for distributed transportation applications, where different actors evolve in a highly dynamic and very constrained environment. However, existing coordination languages can hardly be used for transportation applications, because they fail to express agents complex interaction needs and to ensure secure data exchanges. Indeed, in transportation applications, the interaction needs of the agents are driven by their current context and by *ambient* conditions, and information security is usually important. In this article, we propose a data driven coordination language tackling these issues, and we define a programming language on top of Java allowing to use the language syntax while executing it in Java. We illustrate our proposal with two applications: a traveler information system and a demand-responsive transport service.

## Keywords

Data Driven Languages, Applications, Security

## 1. INTRODUCTION

Transportation applications rely more and more on networks of mobile devices such as vehicle and mobile ad-hoc networks. These applications are increasingly based on entities which interaction follows a dynamically reconfigurable communication scheme. These relatively new classes of applications are calling for new models and languages adequate for their design, management and implementation. Indeed, developing such logically distributed applications, without knowing either the overall structure of the system is a challenge and needs specific models and languages to simplify their implementation.

Based on our previous developments (e.g. [1, 18]), we identify three recurrent issues when dealing with transportation applications. The first is related to knowledge management. In many problems such as urban network regulation [1], the knowledge is incomplete or is only known by human experts. The knowledge issue is also related to the coordination of many information sources as in the traffic regulation problem [9]. The multiagent paradigm is well suited to deal with this issue. This paradigm gives the necessary abstraction level to take into account incomplete knowledge. For Parunak [10], “Agent-based modelling is most appropriate for domains characterized by a high degree of localization and distribution”.

The second issue is the dynamics of the real environment, which impacts the quality of the information, makes direct communication difficult and costly and/or implies that mobile entities appear and disappear. The data driven coordination models are well suited to deal with this dynamics. The principle is to use a shared dataspace and to let agents opportunistically get the information they need, without having to maintain any knowledge about the other agents of the system.

The last issue is the security issue, since users usually need to avoid being tracked or localized by other agents or authorities. For example, in the design of location-based applications, the user has to be sure that his needs remain personal and can not be used by other people.

Lacios[1] is a data driven coordination language designed to take into account these three issues. It builds on top of data driven languages enabling the programmer to design and implement secure multiagent systems for transportation applications. To illustrate our proposal and the syntax of Lacios, a transportation example application is used throughout the paper. In this example, human travelers are in a train station in which schedules, booking, payment services and information sources coexist. Two agent types are considered in here: Traveler agents represent travelers wishing to make a journey and Train agents represent trains, and generate information concerning future departures, arrivals, delays, etc. All these agents interact by exchanging data via a shared space in the same way as for all data driven coordination models.

This paper is organized as follows. Section 2 presents the existing solutions to these three issues. Section 3 gives an overview of the language that we are proposing. Section 4 describes Lacios and details its syntax. In section 5, we detail the security management in Lacios. Section 6 presents the programming language Java-Lacios. In section 7, we present the traveler information system based on Lacios

[1]Language for Agent Contextual Interaction in Open Systems

and the coordination environment for a demand-responsive transport service. Finally, section 8 concludes the paper.

## 2. STATE OF THE ART

Existing multiagent programming languages (such as Agent0 [13], AgentSpeak [12], dMars [3] and Claim [14], to cite only a few) gives the necessary abstraction level in the implementation of the agents. However, the fact that they rely on a channel-based communication makes it hard to design and implement open systems in which agents join and leave the system freely. Instead, data driven coordination languages, with the pioneer language Linda [6] and its extensions such as IBM's TSpaces® [16] and Sun's JavaSpaces® [5], provide the possibility for new agents to join the system and, since all the agents have a common interlocutor (the shared space), they don't have to manage an up-to-date address book of the other agents of the system. Agents communicate by exchanging tuples via an abstraction of an associative shared memory called the *tuplespace*. A tuplespace is a multiset of tuples (tuples duplication is allowed) and is accessed associatively (by contents) rather than by address. Every tuple is a sequence of one or more typed values. Communication in Linda is said to be *generative*: an agent *generates* a tuple and its life cycle is independent of the agent that created it. The tuplespace is manipulated by three atomic primitives: *out* to add a tuple, *rd* to read and *in* to take a tuple (read it and remove it from the tuplespace). The parameter of *out* is a fully instantiated tuple (sequence of values), and the parameter of an *in* or a *rd* primitive is a template: a tuple with potentially one or more formal fields (variables). A tuple and a template match if they have the same arity and if every field in the tuple is either equal to the corresponding value or of the same type of the corresponding variable in the template. The primitives *in* and *rd* are blocking: if no tuple matches their parameter template, the caller agent is suspended until a matching tuple is present in the tuplespace. An additional primitive, $eval(P)$, launches a new agent that will run in parallel with the caller.

Nevertheless, systems adhering to data driven coordination models have two main limitations concerning the language expressiveness and the security management. Agents in the coordination languages literature are stateless processes, the language describe what they *do*, but not what they *are*. In the absence of agents' states, agents interaction cannot be conditioned by their own current state or context. The interaction conditions are also poor (limited to the type, position and value of the components of the tuples) and unable to express agents' complex concepts and interaction needs. Besides, in the literature the tuple data structure is enriched on the implementation level (object oriented [5], relational [16], logic [8]) but not on the model level. The result is that the matching mechanism used remains based on templates, which limits the expressive power offered to agents to express their interactional needs. For example, the poor expressiveness of Linda-like languages materializes in the cases where an agent desires to condition its interaction by the status of data that are different from the data it wants to access. In a traveler information system for instance, a traveler agent could be interested by bus tickets if the weather is sunny, and underground tickets if it is rainy. The data it wants to access are tickets, but it is conditioned by other data (the weather conditions).

When used in untrusted environments, systems adhering to data driven coordination models encounter several security threats due to their information sharing. The authors in [11] classify security threats into: i) threat on confidentiality; ii) threat on integrity; iii) threat on availability; iv) threat on authenticity. For instance, consider a Traveler agent in a traveler information system, that adds a message to the shared data space, claiming that this message comes from a Train agent and informing about a delay. This information may mislead the other Traveler agents, which will miss the train if they get this information and act in consequence. The confidentiality threats are related to the interception by an agent of another agent's confidential information or message, and the threats on availability concern the deletion of agent's information or message by another one. For instance, consider a Traveler agent $t$ that tries to read or take a message from a Train agent to another Traveler agent with an information about the ticket price it is proposing. The agent $t$ has to be prevented from reading (confidentiality) and taking (availability) this message. Finally, the integrity threats are related to the modification by other agents of another agents' messages or information.

In the literature, several modifications have been made to Linda-like languages to tackle the security threats. In [4], the authors classify secure data driven languages into entity driven and knowledge driven languages. The idea behind the knowledge driven approach is that tuple spaces, tuples or single data fields are decorated with additional information. Agents can access the resources only in the case they prove their knowledge of this information. In the case of the entity driven approach, additional information associated to resources lists the agents that are allowed to access the resources. We would like to equip data driven coordination languages with a security mechanism that allows for the protection of the exchanged data in a fine-grained way. Our proposal can be classified as an entity driven approach. However, instead of listing the agents that can access a datum, we propose a language in which these agents are described *symbolically*, i.e. their properties are defined without pointing them namely. Indeed, because of this listing requirement, the authors in [4] state that the knowledge driven approach is more suitable for open systems; In this sense, we propose an entity driven approach that is suitable for open systems. We want to let agents specify, when they add a datum to the data space, the conditions under which it can be read or taken by others. We also would like the designer of the system to specify the conditions under which an agent can or cannot add a certain datum to the space, following the application logic. Equipping data driven coordination languages is the objective pursued in this article. To this end, we perform several modifications of the shared space model, and propose a new language, called Lacios, which is the linguistic embodiment of the modified model.

## 3. OVERVIEW

A MAS written in Lacios is defined by a dynamic set of *agents* interacting with an *environment*, which is composed of a dynamic set of *objects*. A MAS written in Lacios is an open system in two ways. As for every data driven language, agents in Lacios can join and leave the system freely. In addition, external - non modeled - systems and users can interact with the MAS. As we will define it later, users (e.g. travelers) interact with the MAS by instantiating the values of certain variables in the code of the agents that represent them in the system. External systems (e.g. trains) can interact with the MAS by instantiating variables with values as well. They can also execute agents that interact with the MAS Environment via a local agent. The figure 1 illustrates the MAS architecture. The modeled MAS executes on a host, where (local) agents add, read and take objects to/from the MAS environment. Every agent is either independent (like agent 1), or representing a non-modeled system/user in the MAS (like agents 2, 3 and 4).

The agents that are defined in a LACIOS program are usually the *local* agents. The users, external agents and external systems that are represented by an agent in the MAS are not modeled, only their actions are observed in the MAS, through the nondeterministic behavior of the local agent. Distant agents (like $agent'_2$) provide their new state to the local agent whenever it changes. An agent in LACIOS is then an entity, that has a state, a local memory and a nondeterministic behavior. As we will define it later, the whole behavior of the agent is not defined in LACIOS: an agent can have a complex behavior, by using additional operators, besides the standard operations defined in LACIOS.

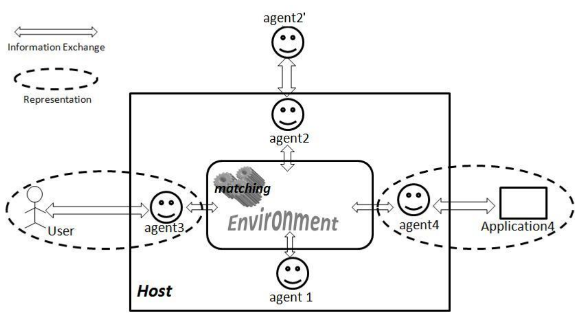


**Figure 1: LACIOS Architecture**

Since agents in LACIOS don't interact directly, but via the environment, our definition of an agent is close to the general definition given by [15]:

DEFINITION 1 (AGENT). *An agent is a computer system capable of autonomous action in some environment in order to meet its design objectives.*

From the security point of view, LACIOS has two objectives: i) to support a global control by the environment of objects' insertion by the agents in order to ensure that the new objects are not fraudulent (authenticity and availability), ii) to support a local control by the agents that can specify *who* can access the object that they add to the environment in order to ensure their privacy (authenticity, confidentiality, availability and integrity). To do so, agents have to have a state defining who they are. This is the first modification we perform to the original model: the consideration of an abstraction of agents' states in the form of data at language level. These states are defined as a set of *property←value* pairs (e.g. $\{identifier \leftarrow 10, position \leftarrow node_1\}$). Agents' states in LACIOS are data representing the state of the agents that are accessed by the environment only for matching and security purposes (they are not directly accessible by the other agents). The second modification is the proposal of a rich interaction mechanism, using operators and variables, which allow to express the two levels of security management: local and global control. The next section is dedicated to the language specification.

# 4. THE COORDINATION LANGUAGE LACIOS

## 4.1 Overview

LACIOS is a coordination language that extends Linda for the design and implementation of MAS, defined in a suitable way for transportation applications. For the specification of agent behavior, we adopt four primitives inspired by Linda and a set of operators borrowed from Milner's CCS [7]. A MAS written in LACIOS is defined by a dynamic set of *agents* interacting with an *environment* - denoted $\Omega_{ENV}$, which is composed of a dynamic set of *objects*. Agents can *perceive* (read only) and/or *retrieve* (read and take) objects from the environment. Agents are defined by a behavior (a process), a state and a local memory in which they store the data they perceive or retrieve from the environment.

The distinguishing features of LACIOS that we focus on in this paper can be summarized as follows. First, an agent can publish its state, update it and use it to condition its interaction with the environment. Second, the data structure (for exchanged data and for agents' states) is based on typed property-value pairs. Finally, an agent can use complex conditions (using operators and functions) on its own state and on other shared objects in order to access the environment, with a single instruction.

We have proposed an operational semantics, unambiguously defining the behavior of a MAS written in LACIOS, which can be found in [17].

## 4.2 Syntax

### 4.2.1 Data Structure

For LACIOS, we define a standard information system data structure: every datum in the system has a *description*, i.e. a set of *property←value* pairs, and all the properties of the language are typed. We define in the following the notions of a type, a property and a description.

DEFINITION 2 (TYPES). *The types of the language are defined as* $type_1, \ldots, type_{nbt}$. *Every* $type_i$ *is a set such that* $\forall (i,j) \in \{1,\ldots,nbt\}^2, i \neq j, type_i \cap type_j = \{\text{nil}\}$

REMARK 1. *We assume the existence of the* boolean *type in the language, i.e.* $\exists i \in \{1,\ldots,nbt\}, type_i = \{\text{true}, \text{false}, \text{nil}\}$

DEFINITION 3 (PROPERTY). $\mathcal{N}$ *is the property space, it is a countable set of properties. A property* $\pi \in \mathcal{N}$ *is defined by a type* $type(\pi) \in \{type_1, \ldots, type_{nbt}\}$.

A description is composed of properties and their corresponding values.

DEFINITION 4 (DESCRIPTIONS). *DS is the set of descriptions. A description is a function that maps properties to values, i.e.* $d \equiv \{\pi \leftarrow v_\pi \mid v_\pi \in type(\pi)\}_{\pi \in \mathcal{N}}$. *The mapping is omitted when* $v_\pi = \text{nil}$. *We use* $d(\pi)$ *in order to access the value* $v_\pi$. *For every description, the set of properties* $\{\pi \mid d(\pi) \neq \text{nil}\}$ *is finite.*

A property evaluated to nil is considered undefined. In LACIOS, every description is associated with an *entity*. An entity can be an *object* or an *agent*. An object is defined by its description ($\mathcal{O}$ is the set of objects), while an agent is defined by a description and a behavior ($\mathcal{A}$ is the set of agents).

DEFINITION 5 (ENTITIES). *$\Omega = \mathcal{A} \cup \mathcal{O}$ is the set of entities of the MAS. Each entity $\omega \in \Omega$ has a description as defined above denoted by $d_\omega$. The value of the property $\pi$ of the entity $\omega$ is denoted by $d_\omega(\pi)$.*

REMARK 2. *We assume the existence of the type* reference *in* LACIOS*, a value of the type* reference *designates an entity in $\Omega$, i.e. $\exists i \in \{1, \dots, nbt\}$, $type_i = \Omega \cup \{\text{nil}\}$.*

For instance, let $o_1$ be an object, $d_{o_1}$ could be defined as follows: $\{id \leftarrow \text{“}o1\text{”}, destination \leftarrow \text{“}Montreal\text{”}, from \leftarrow \text{“}Paris\text{”}\}$. In this example, $d_{o_1}(from)$ is equal to “$Paris$”.

### 4.2.2 Expressions

Expressions are built with values, properties and operators. We define an operator as follows.

DEFINITION 6 (OPERATORS). *Each operator op of the language is defined by:*

- *(i) $arity(op)$ The number of parameters of the operator,*
- *(ii) $par(op) : \{1, \dots, arity(op)\} \rightarrow \{1, \dots, nbt\}$, $par(op)(i)$ gives the index of the type of the $i^{th}$ parameter of the operator op,*
- *(iii) $ret(op) \in \{1, \dots, nbt\}$, the index of the type of the value resulting from the evaluation of op.*

For instance, let $type_1 \equiv boolean$. The operator `and` is defined as follows:
$arity(\texttt{and}) = 2$, $par(\texttt{and})(1) = par(\texttt{and})(2) = 1$ and $ret(\texttt{and}) = 1$.

Expressions are used by agents to describe the data they manipulate, either locally or to interact with the environment. An expression may simply be a value, an operator, or a property. If an expression is a property, it refers to a property of the agent that is evaluating it. For instance, when *destination* appears in the behavior of the agent $a$, it designates the destination of $a$. If a property *neighbor* of the agent $a$ is of type *reference*, *neighbor.destination* designates the destination of the *neighbor* (an entity) of $a$.

DEFINITION 7 (EXPRESSIONS). *Exp is the set of expressions. An expression $e \in Exp$ is generated via the grammar of table 1.*

We can now associate an expression with a property instead of a value in a description. The result is a *symbolic description* which is transformed into a description when its associated expressions are evaluated.

DEFINITION 8 (SYMBOLIC DESCRIPTIONS). *SDS is the set of symbolic descriptions. A symbolic description is a description that maps properties $\pi$ to expressions $e_\pi$, i.e. $sds \equiv \{\pi \leftarrow e_\pi \mid type(e_\pi) = type(\pi)\}_{\pi \in \mathcal{N}}$.*

| | |
|---|---|
| $e ::= \text{nil}$ | |
| $\mid v$ | , with $v \in \mathcal{T} \backslash \text{nil}$ |
| $\mid \pi$ | , with $\pi \in \mathcal{N}$ |
| $\mid op(e, \dots, e)$ | , with $op$ an operator of the language, and nil doesn't appear in any $e$ |
| $\mid \pi.e$ | , with $\pi \in \mathcal{N}$ and $type(\pi) = \Omega$ |

**Table 1: Syntax of an expression**

## 4.3 Matching

Since we consider a data structure richer than tuples, we also use a matching mechanism richer than templates. To do so, we enhance the expressions' syntax with *entity variables*, which designates objects not known by the agent, but will be discovered during the matching process and will be replaced by objects from the environment before their evaluation. Here follows the definition of a variable.

DEFINITION 9 (VARIABLES). *$\mathcal{X}$ is the set of variables. A variable $x \in \mathcal{X}$ is defined by its type $type(x) \in \{type_1, \dots, type_{nbt}\}$.*

The syntax of an expression becomes:

$$e ::= \cdots \mid x.e \text{ with } x \in \mathcal{X} \wedge type(x) = \Omega$$

For instance, consider the following boolean expression $e$: $t.destination = \text{“}London\text{”} \wedge t.ind_{price} \leq budget$. In this expression, $t$ designates an object, unknown for the moment, where $t$ has to have as *destination* “London” and an $ind_{price}$ (standing for individual price) less than the *budget* of the agent for the expression to be evaluated to true, in which case the agent executing *look* with $e$ as a parameter will perceive or retrieve the object.

In the previous example, a single object from the environment (unified with $t$) is needed for the expression to be evaluated. However, the introduction of variables allows for a richer matching. The matching in LACIOS materializes what we call a *contextual interaction*, which is the type of interactions that uses the state of the agent and the state of several objects in the environment to access a set of objects, instead of only one like in Linda templates.

To illustrate the need for this type of interactions, consider the following scenario. Let's say that train agents add objects representing tickets in the environment, designating remaining empty seats. These objects are of two types. The first type is “individual ticket” representing a single empty seat in the train. It's described by the following properties: *id* designating the identifier of the train, *destination* designating his destination and $ind_{price}$ designating the price of the ticket. The second type is “group ticket” representing a group of empty seats in the train (we consider that group prices are applicable starting from three tickets bought together). Group tickets are cheaper than individual tickets and are described by the following properties: *id* designating the identifier of the train, *destination* designating it destination and $group_{price}$ designating the price of the ticket. However, in order to buy such a ticket, a single traveler agent cannot ask for it alone, and has to find two other agents that have the same destination and enough money to buy the group ticket together. Without contextual interaction, an agent has to first look for a ticket with a $group_{price}$ that is lower than his own budget before to search individually for two agents with enough budget and the same destination as his. Meanwhile, the ticket could have been bought by another agent, or when the agent finds a first traveler and looks for a second, the first could've bought another ticket. A contextual interaction allows for an all-or-nothing request, including his own state and several objects from the environment. In a single pace, the agent would find the ticket and both agents that would share the ticket price with him. The corresponding expression to this example is the following:

$$(t.destination = destination) \wedge (budget \geq t.group_{price}) \wedge$$
$$(x.destination = destination) \wedge (x.budget \geq t.group_{price}) \wedge$$
$$(y.destination = destination) \wedge (y.budget \geq t.group_{price})$$

The variable $t$ designates the ticket, $x$ and $y$ represent the two needed agents. This expression needs three objects from the environment in order to be evaluated: a ticket and two agents' descriptions. Either the agent will find the three needed objects and buy the ticket or it won't, in which case it won't launch any further useless interaction.

We can now provide the complete definition of the primitive *look*. We choose to use a single primitive to access the environment (instead of *in* and *rd*). The primitive $look(sds_p, sds_r, e)$, with $sds_p$ and $sds_r$ symbolic descriptions, allows both object perception and retrieval (perception and removal from $\Omega_{ENV}$). It blocks until a set of objects $C$ becomes present in $\Omega_{ENV}$ such that the expression $e$ is evaluated to true. When an agent executes $look(sds_p, sds_r, e)$, the set of objects of the environment $C$ is selected for matching with $e$ (an objet for each context variable), and $e$ has to be evaluated to true for the consequence of the action to be effective. The objects associated with the variables in $sds_p$ are perceived and those associated with the variables in $sds_r$ are retrieved. For instance, the following instruction: $look(\{ticket \leftarrow t\}, \{paper \leftarrow p\}, t.destination =$ "*London*" $\wedge\ t.price \leq budget \wedge p.decision =$ "*accepted*") looks for two objects that will be unified with $t$ and $p$. The object associated with $t$ will be perceived while the object associated with $p$ will be retrieved. After the execution of this instruction, the two objects will be present in the local memory of the caller agent, who will have two additional properties of type *reference*: *ticket* that refers to the object associated with the variable $t$ and *paper* that refers to the object associated with $p$.

## 4.4 Interaction with External Systems/Users

Consider an agent having two properties *destination* and *budget* that are unknown before execution. The values of these properties come from an external system (e.g. a Web server, a GUI, etc). Here is the description of this agent which properties will be defined during execution resulting from their instantiation by an external system: $\{budget \leftarrow b, destination \leftarrow d\}$, where $b$ and $d$ are variables. When writing his/her program, the programmer doesn't have to care about the provenance of this information, only the action of the external system will be observed on execution time, i.e. the assignment of values to the variables, while the external system itself is not modeled.

We enhance the syntax of an expression with free variables as follows:

$$e ::= \cdots \mid x \text{ with } x \in \mathcal{X}$$

The introduction of the variables for the interaction with an external system is interesting insofar as it clearly separates the coordination aspect - what the MAS does - from the interaction with an external system aspect - the context in which the MAS is running. Thus, in the description $\{budget \leftarrow b, destination \leftarrow d\}$, regardless of which system is instantiating the variables $b$ and $d$, the definition of the description and the behavior of the agent remain unchanged.

# 5. SECURITY MANAGEMENT

We have decided to maintain global sharing of the data between all the agents, and not to isolate them in private environments, thus following the original Linda model. However, this choice leads to the same security problems encountered with the original model. More precisely, fraudulent data insertion and retrieval could occur and the agents and the system designer cannot prevent them. In LACIOS, the agents are responsible of the objects that they put in the environment. In order to avoid fraudulent use of these objects, the language supports two control levels, a global level for the designer of the system to control the insertion of objects and a local level for the owner of the objects to control how their object will be used. The global level guards every object insertion in the environment. It verifies wether the agent's properties gives it the right to insert the object. This check would notably guarantee the authenticity of the emitted messages, by checking that the agent is not trying to forge a message. The local level guards every object perception or retrieval. It verifies that the object's owner allows the agent trying to read or take the object to do so. These checks guarantee the confidentiality, the availability and the integrity of the exchanged data. It is noteworthy that LACIOS manages security following the original Linda model, i.e. associatively. The communication remains anonymous and the security is managed symbolically, together with keeping a complete sharing of the data between the agents of the system.

## 5.1 Global Control

The designer of the system knows the conditions under which certain insertions of objects are fraudulent and we provide him/her with a global control of objects insertions by the agents. A threat to authenticity (when an agent tries to forge a message for example) is an example of such a fraudulent insertions. More generally, objects added to the environment might corrupt the coherence of the data according to the application logic (resulting in two agents with the same *position*, or with a new bid that is lower than the current one, etc.).

Let us consider for instance, the following action:

$add(\{from \leftarrow companion_2.id, to \leftarrow companion_1.id, subject \leftarrow$ "*coalition*"$\})$

This action is fraudulent, since the agent tries to send a message to its first companion with a different *id* than its own ($d(from) \neq id$).

This first class of threats (which includes the authenticity threat) concerns the security rules that have to be checked when an *add* is executed. To overcome threats resulting from the fraudulent insertion of objects in the environment, the system designer identifies the critical situations and specifies each one using a security rule $s$ ($s \in \mathcal{S}, \mathcal{S} \subseteq Exp$ is the set of security rules of the system). An expression $s$ in $\mathcal{S}$ is a boolean expression in which the designer specifies the conditions on the state of the agent executing *add* and the conditions on the description of the object that it adds. To do so, we add a specific key word *that* in the syntax of an expression to designate, in a security rule, the object added by the agent.

$$e ::= \cdots \mid that.e$$

For instance, here is the expression preventing an agent from adding an object that has a property *from* that is different from its own: $s \equiv that.from = id$, where *id* designates the identifier of the agent executing the *add* and *that* designates the object added by the agent. When an agent $a$ executes $add(\{from \leftarrow companion_2.id, to \leftarrow companion_1.id, subject \leftarrow \text{“}coalition\text{”}\})$, the security rule specified by the designer is evaluated to false, because $d(from) \neq d_a(id)$, and the operation is canceled.

## 5.2 Local Control

The agents of the system know best the conditions under which the perception or retrieval of an object they add is fraudulent, and we provide them with local control to manage the observability of their own objects. A confidentiality threat (e.g. the interception by an agent of another's confidential information or message), or a threat to availability (e.g. the deletion of the agent's information or message by another agent) are examples of such fraudulent access. We propose to allow agents to define the observability rules - on perception and on retrieval - and to let the environment check that these conditions are respected.

This is done by enabling an agent, when it adds an object, to manage its observability, i.e. to identify the situations where the perception or retrieval of the inserted object is prohibited. To do so, the syntax of the primitive *add* is replaced as follows.

$$\mu ::= \cdots \mid add(sds, e_p, e_r)$$

where $e_p$ and $e_r$ are boolean expressions. The expression $e_p$ specifies the conditions that an agent has to satisfy to have the right to perceive the object described by *sds*, and $e_r$ defines the conditions that an agent has to satisfy to have the right to retrieve it. When an agent executes $look(sds_p, sds_r, e)$, for each object $o \in C$ (the set of objects selected for matching from the environment) that is unified with a variable in $sds_p$, the expression $e_p$ associated to $o$ has to be evaluated to true, and for each object $o$ unified with a variable in $sds_r$, the expression $e_r$ associated with $o$ has to be evaluated to true. Otherwise, the action *look* cannot be executed with this set of objects. When the agent doesn't want to restrain the perception or the retrieval of the object described by *sds*, it assigns true to $e_p$ or $e_r$, respectively. For instance, let agent $a$ (let's say that $a$'s $companion.id = 5$) wants to prevent the message it has addressed to its *companion* to be retrieved by others, and to be perceived by any agent but itself (the key word *that* has the same semantics here, i.e. it designates the inserted object): $add(\{from \leftarrow id, to \leftarrow companion.id, subject \leftarrow \text{“}coalition\text{”}\}, id = that.from, id = that.to)$

Consider an agent $b$ with $d_b(id) = 10$ that executes $look(\{receiver \leftarrow r\}, \{message \leftarrow m\}, m.to = r.id \wedge r.destination = destination)$

The agent $b$ is trying to retrieve a message (object unified with $m$) and to perceive the object representing the agent to which $m$ is addressed (object unified with $r$), if its destination is equal to its own. Thanks to the conditions associated with the added object, $b$ won't be able to perceive $a$'s message. Concretely, any matching that is trying to unify $m$ with $a$'s message is prohibited by the environment and is not considered.

Aside from authenticity, confidentiality and availability, a fourth threat for information security is integrity. The integrity threats are related to the modification by other agents of another agents' messages or information. However, since the *update* primitive is local in Lacios, no agent can modify others' information. The only way to do it - as it is the case for all data driven languages - is to take an object and to create a new one with different information. Taking an object is already guarded by security rules against interception. As a consequence, there is no need for a specific mechanism for the threats on integrity in Lacios.

A fifth threat on information security is traditionally listed, which is the non-repudiation threat, implying that one party of a transaction cannot deny having received a transaction nor can the other party deny having sent a transaction. Due to the generative communication *à la* Linda, we cannot map a certain message to its sender or receiver. None of the data driven coordination languages tackle this issue. Nevertheless, when implemented, a system adhering to our model can use a log with such a mapping, that can be read by agents, to solve this problem. Provided that the authenticity is guaranteed with our global control mechanism, the content of this log is certified correct.

Note that, in the development of the security management defined above, we only take into account the security between local agents and the environment. By doing so, we make two assumptions. On the one side, the *spawn* of an agent representing an external system, user or agent, has to be fulfilled following a security protocol to ensure that this is indeed the agent with the claimed identifier. On the other side, we prohibit local agents from trying to change their identifiers with an *update* throughout the execution of the system. We enforce this prohibition by stating a single value in the definition domain of the concerned property; as a consequence, changing these properties would provoke a semantic error.

## 5.3 Specification of Agent Behavior

This paragraph provides the complete definition of an open MAS written in Lacios, starting with the complete definition of the primitives for Lacios.

$\mu ::= add(sds, e_p, e_r) \mid look(sds_p, sds_r, e) \mid update(sds) \mid spawn(P, sds)$

We are now ready to define processes, which define agent behavior. The primitive $spawn(P, sds)$ launches a new agent that behaves like the process $P$ and whose description is the result of the evaluation of *sds* (its transformation to a description *ds*). Below is the definition of a process, which defines agents' behaviors.

Definition 10 (Process). *Given a set of process identifiers $\{K_i\}_{i \in I}$, a process definition is of the form: $\forall i \in I, K_i \stackrel{def}{=} P_i$, where every $P_i$ is generated via the grammar in Table 2.*

Processes, ranged over by $P, Q, \ldots$ represent the programs of the MAS, and the behavior of its agents. A program can

| | |
|---|---|
| $P ::= \mathbf{0}$ | (null process) |
| $\mid \mu.P$ | (action prefixing) |
| $\mid b\lfloor P \rfloor + b\lfloor P \rfloor$ | (choice) |
| where $b$ is a boolean expression | |
| $\mid P \| P$ | (parallel) |
| $\mid K_j$, for a certain $j \in I$ | (invocation) |
| $\mid \nu X(P)$ | (variables linking) |
| $\mu ::= spawn(P, sds) \mid add(sds, e_p, e_r) \mid$ $\mid update(sds)$ | $look(sds_p, sds_r, e)$ |
| with $e$, $e$ and $e_r$ expressions, $sds, sds_p$ descriptions | and $sds_r$ symbolic |

**Table 2: Process syntax**

be a terminated process $\mathbf{0}$ (usually omitted). It can also be a choice expression between programs $b\lfloor P \rfloor + b\lfloor P \rfloor$, where each $P$ is guarded by the evaluation of a boolean expression: when $b$ is evaluated to true, the program $P$ is executed. A program can also be a parallel composition of programs $P \| Q$, i.e. $P$ and $Q$ are executed in parallel. A program can be an invocation of another process whose identifier is the constant $K_j$, and which behaves like the process defined by $K_j$. A program may be a process prefixed by an action $\mu.P$. Actions are the language primitives, as defined earlier. The operator $\nu$ is introduced to link free variables in $P$. The process $\nu X(P)$ introduces nondeterminism in the agents' behaviors. Indeed, behaves like $P$ where every free variable (in $X$) is nondeterministically linked with a value in its type.

A coordinated MAS is then defined as follows.

Definition 11 (Coordinated MAS). $CS = \langle \Omega, d, \Omega_{ENV}, \mathcal{S} \rangle$

- *$\Omega = \mathcal{A} \uplus \mathcal{O}$ is the set of entities, composed of $\mathcal{A}$ the set of agents and $\mathcal{O}$ the set of objects,*
  - *$\mathcal{A} \subseteq \Omega$ is the set of agents.*
    - *$\Omega_a$ is the private memory of agent $a$, $\Omega_a \subseteq \mathcal{O} \cup \{a\}$, i.e. the agent has access to its own description,*
    - *$proc(a)$ is the process defining the behavior of $a$.*
  - *$\mathcal{O} \subseteq \Omega$ is the set of objects.*
    - *$e_p(o)$ returns the predicate specifying the perception conditions of $o$, i.e. which agents can perceive $o$.*
    - *$e_r(o)$ returns the predicate specifying the retrieval conditions of $o$,*
- *$d : \Omega \rightarrow (\mathcal{N} \rightarrow \mathcal{T})$ is the description function of the MAS, each $d(\omega)$ is an entity description as described before (denoted by $d_\omega$ as well),*
- *$\Omega_{ENV} \subseteq \mathcal{O}$ is the set of objects that are in the environment,*
- *$\mathcal{S} \subset Exp$ is the set of predicates specifying the conditions that have to be verified, in order for an add to be executed.*

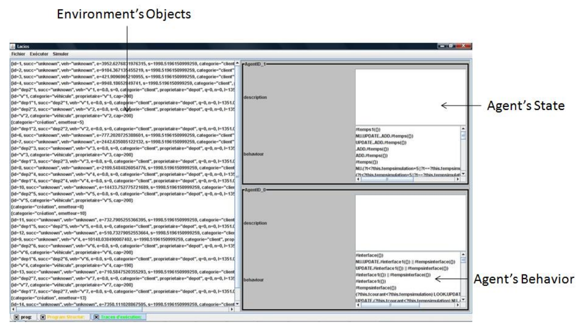


**Figure 2: Visualization of agents behaviors**

# 6. THE PROGRAMMING LANGUAGE JAVA-LACIOS

We have defined a language that, following its operational semantics (cf. [17]), could be implemented in any host language. The usual procedure in order to implement a coordination language is to provide libraries in a host programming language that can be used by any system wishing to adhere the coordination model (e.g. Klava, which is associated to Klaim [2]). However, to take full advantage of the language semantics, it is more useful not to require the programmer himself/herself to respect the semantics in each system that he/she implements. This is possible by providing him/her with a tool allowing to write a program in Lacios's syntax, and to generate a system ready to be executed, with the guarantee that it respects the language semantics. In particular, we want to use the operators of prefixing, choice and parallel composition when defining the agents' behaviors. Java has been chosen as a target programming language in which a compiled Lacios program is translated, because of the relative simplicity of Thread management, as well as the easy creation of parsers thanks to the parser generator JavaCC [2].

A Java-Lacios program is a file where both the behaviors and the initial state of the coordinated MAS are described. A coordinated MAS is defined by the set of initial agents, spawned when the program starts, together with the security rules $\mathcal{S}$. Programmers write their scripts which are parsed and compiled, generating a Java program. We have proposed a GUI for Java-Lacios, which displays the ongoing execution, the current objects in the environment, the current agents' behaviors that are executed, etc. The Fig. 2 illustrates the execution with a Demand-Responsive Transport Service that we have implemented and that will be described in the following section. It is also possible, before the execution, to visualize the graphs (labeled transition systems) related to the agents' behaviors as shown in Fig. 3.

The translation of a Lacios program to a Java program is quite straightforward, but in order to respect the semantics of the language, a few details are noteworthy. On the one hand, the concurrent access to the environment objects necessitates a synchronization of the *add* and *look* calls. However, since an *add* call is non-blocking (following the original Linda model), an agent calling *add* has not to be blocked until the environment releases the lock. To this end, we have defined a buffer to which agents can add objects with-

[2] http://javacc.dev.java.net/

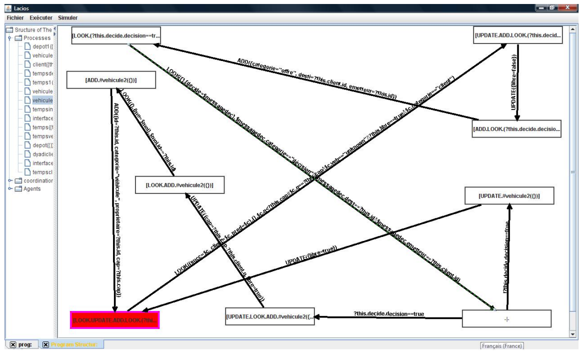

**Figure 3: Labeled Transition System of a LACIOS agent**

out blocking, while emptying the buffer is synchronized with the *look* calls. On the other hand, when a *look* is called, the environment is locked while it is still looking for a matching, to guarantee that an agent does not access the environment in an incoherent state, and to be sure that a same object is not retrieved by more than one agent. If no matching is found, the calling process of the agent is blocked. The blocked processes are notified when an object is added to the environment. In this case, the notified process looks for a matching with the only newly added objects.

Finally, an *update* modifies the agent's properties *locally*, but it however influences its interaction with the environment. Indeed, if a *look* is currently executing, the matching have to be attempted with the current properties of the agent. When the properties of an agent change, and when they concern properties for which an ongoing *look* has attempted to match, the execution of the *look* is executed again, and the pending *look* requests are notified since they might be concerned by the newly changed properties as well.

# 7. APPLICATIONS

## 7.1 An Environment-Centered MAS for Traveler Information Systems

In this section, we describe an application based on Lacios. We modeled and implemented a traveler information server [18], called ATIS[3]. The server purpose is to inform online travelers about the status of a part of the transport network that concerns them. Every traveler is mobile and has a specific objective during his connection on the server. Transport Web services are represented with agents in the server and their expertise domains constitute their properties. These transport service providers can give information in response to a request, or they may proactively send information concerning disturbances, accidents, events, etc. The problem in this kind of applications concerns the information flows that are dynamic and asynchronous. Indeed, each information source is hypothetically relevant. An agent cannot know *a priori* which information might interest him, since this depends on his own context, which changes during execution.

### 7.1.1 Context

Using Lacios allows us to design an information server parameterized by its users. Indeed, if the information systems are adapted to the satisfaction of punctual needs (request/response), the management of the information followup assumes additional processing. These processing are difficult because the information sources are distributed and the management of the followup assumes a comparison of the available information.

We have defined two categories of agents, the first concerns the agents representing the users (that we call PTA for Personal Travel Agent) while the second concerns the agents representing the services (that we call Service Agent).

### 7.1.2 Technical Description

We have implemented a Web server for traveler information, where each Web service has a representant in the server, which is responsible of the convey of messages from the server to the port of the Web service and conversely. The exchange of messages between the server and the services are SOAP [4] messages and the asynchronous communication is fulfilled via the JAXM API [5] for the Web services supporting SOAP, and a FTP server otherwise, used as a sort of mailbox. These details are obviously transparent for the agents evolving in the environment i.e. they interact exactly the same way whatever the transport protocol that is used. Every user is physically mobile and connects to the server via a Mobile Personal Transport Assistant (MPTA) and has during his connection a PTA agent representing him inside the server, which is his interlocutor during his session. The interaction of the MPTA with the PTA agents representing them is a sequence of HTTP requests/responses.

### 7.1.3 Execution Scenario

The context of the example is the following: inside the system, there is an agent representing a trip planning service and an agent representing a traffic service responsible of the emission of messages related to incidents, traffic jams, etc. These agents are persistent, since they are constantly in relation with the system providing the service. On the contrary, PTA agents representing the MPTA in the system are volatile, created on the connection of a user and erased at the end of his session i.e. when he arrives at destination. We have developed a trip planning service which role is to, first receive the trip planning demand (in the form of a SOAP message), then calculates the plan, wraps it in a SOAP message then sends it back to the local agent representing him in ATIS.

Every stop of the network is described by a line number *line* to which it belongs, and a number *number* reflecting his position on the line. A user $u$ is also described by his current position in the network (the properties *line* and *number*). In a basic execution scenario, $u$ has a path to follow during his trip i.e. a sequence of tuples $\{(line, number_{source}, number_{destination})_i \mid i \in I\}$, with $I$ the number of transport means used by the traveler. Every tuple represents a part of the trip, without transfer. To receive his plan, the MPTA connects to the information server, and the agent $u$ representing him is created. Then, the user is asked to specify his departure as well as his destination. Once this informa-

[3]Agent Traveler Information Server

[4]Simple Object Access Protocol, `http://www.w3.org/TR/soap/`

[5]Java API for XML messaging, `http://java.sun.com/webservices/jaxm/`

tion entered, $u$ adds his planning demand in the environment. A demand is an object described by his properties: $from, subject$, etc. Afterwards, $u$ keeps on listening to messages that are addressed to him, this way: $look(\emptyset, \{message \leftarrow x\}, x.tor = id)$. The agent representing the trip planning service is listening to messages asking for a plan: $look(\emptyset, \{request \leftarrow x\}, x.subject = \text{“}plan\text{”})$. As soon as he receives the message, he creates a SOAP message addressed to the trip planning Web service and awaits for the response. When he receives the answer, a message is added to the environment addressed to $u$ with the received plan as body: $add(\{from \leftarrow id,$ $to \leftarrow request.from,\ body \leftarrow plan\},\ id = that.from,\ id = that.to))$. The agent $u$, when he receives the message, analyzes it and displays the result on the user's MPTA. Then, the agent $u$ restrains his interaction to the messages concerning events coming up on his way. To do so, he executes the following action:

$look(\emptyset, \{event \leftarrow x\}, \{x.subject = \text{“}alert\text{”}\}, x.line = line \wedge$ $x.number \geq number \wedge x.number \leq line)$

The agent $u$ is interested by the alerts concerning his transport plan, which is expressed by the preceding *look* action. Let us assume that the agent representing the alert service adds an alert message concerning an accident on the way of $u$ resulting on a serious delay for him. The traveler, via his representing agent $u$, is notified concerning this alert event. Since the properties *line* and *number* are updated at each move of $u$ (each time he moves from stop to stop), the segment concerned by the alert messages gets gradually reduced until the end of the trip. The use of the environment, the constant update of the properties of the PTA agents, together with the use of *look* actions allowed us to maintain a constant awareness of the traveler about problems occurring during his trip, without relying on continuous requests to the server.

## 7.2 Coordination Environment for Demand-Responsive Transporton Services

We have proposed a demand-responsive transport service (DRTS) as a MAS in which the agents' activities are coordinated through the environment, based on Lacios.

### 7.2.1 Demand-Responsive Transport Services

A DRTS is a system designed to answer online customers that desire to be transported from one point in the network to another. Customers specify a time window associated with each point (departure and arrival) inside which they want to be visited. In our application the environment, via the use of Lacios, structures the MAS components temporally and spatially, so that the interaction between agents is driven by their perception of it. The interaction between customers and vehicles is driven by their space-time positions, and the environment is modeled accordingly. A main issue in this application is the dynamics of the environment because when modeled as an MAS, DRTS are open MAS, where agents (e.g. customers and vehicles) join and leave the system freely. In such a dynamic environment, limiting communications is very important, since broadcasting all the available information is very costly. We use an implicit model on which Lacios is grounded, in which communication is decoupled in space and time, so as to offset the loss in information in dynamic environments.

### 7.2.2 System Description

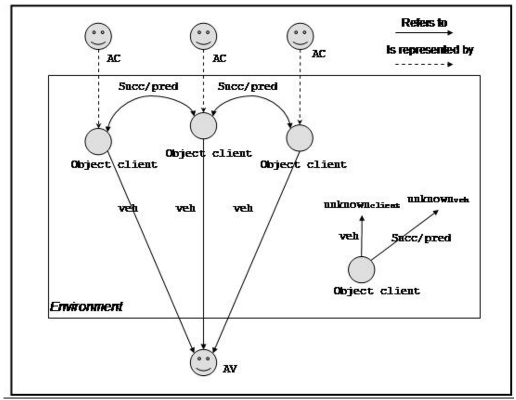


**Figure 4: Symbolic descriptions**

We have designed a distributed model for a DRTS, in which two agent categories are defined: Vehicle Agents (VA) and Customer Agents (CA). Both VAs and CAs are generated dynamically: a new CA is associated to each new customer connected to the system, and a new VA is associated to each new vehicle creation (which occurs when no available vehicle can serve a new customer). The descriptions in this system are related to VAs and CAs (see figure 4). A VA is described symbolically by his current position and his remaining available seats. A CA is described by his departure and arrival nodes, his time windows, the vehicle *veh*, and his successor (property *succ*) and predecessor (*pred*) customer in the route of *veh*. A CA that doesn't belong to any VA route has a property *veh* equal to $unknown_{veh}$, and a property *succ* and *pred* equal to $unknown_{cust}$.

In our application, the boolean expressions contained in the *look* actions are defined by VAs so that they will perceive only those customers they can insert in their route. The description of a new CA (with unknown *veh*, *succ* and *pred*) is perceived when: (i) there are two nodes in the route of the same vehicle between which its departure node can be inserted without violating any of the time windows of the customers in that route, (ii) if there are remaining available seats when this insertion occurs, (iii) if there are two successor nodes in the route of this same vehicle between which the arrival node of the customer can be inserted, without violating any time window of the route. As a consequence, the use of Lacios allows a new CA to discover the VAs that could be interested by its insertion, without knowing them in advance, while at the same time limiting communication in the system to only those agents that can reach an agreement (an insertion in the route of a vehicle). It is worth noting that the VAs that don't perceive a CA can use their time to be candidates for the insertion of other customers.

The protocol followed in the MAS is a negotiation mechanism between CAs and VAs. When a new customer connects to the system, a CA is created (*spawn*), and is perceived by the available VAs with their current *look* actions (that is, which are not already involved in the insertion of another customer). The syntax of expressions that we introduced for Lacios allows for the translation of the mathematical constraints on the insertion of a customer (vehicle capacity and space-time feasibility) onto Lacios expressions, which is not possible for traditional Linda-like languages. Each VA computes an insertion price for the insertion of this customer, and proposes it to the CA (*add*), which will choose the VA proposing the lowest price.

The interested reader about the complete definition of the agents actions and their expression parameters is invited to refer to [17].

In this application, the use of LACIOS structured agent interaction and coordination, and made it more efficient to interact in a dynamic environment where agents appear and disappear without maintaining knowledge about the others and where communications can be disturbed and costly.

# 8. CONCLUSION

When designing transportation systems, we have observed several recurrent issues that call for new models and languages for their design, management and implementation. Although multiagent languages facilitate the design of such applications, they are not efficient with respect to communication costs in open and highly dynamic applications. In this paper, we suggest that using the environment as a common coordination medium is an efficient solution. We have defined the language LACIOS, which is a Linda-like language with distinguishing features making it easier to design and implement open highly dynamic multiagent transportation applications.

The multiagent environment might be distributed over several hosts, but until now, this was done in an *ad hoc* way aiming at reducing communication cots between the different hosts. We are currently working on automatic distribution of the environment, following the specification of entities' properties, and based on their clustering in the form of Galois lattices.

# 9. REFERENCES


[1] F. Balbo and S. Pinson. Using intelligent agents for transportation regulation support system design. *Transportation Research Part C: Emerging Technologies*, 18(1):140 – 156, 2010.

[2] R. De Nicola, G. L. Ferrari, and R. Pugliese. Klaim: A Kernel Language for Agents Interaction and Mobility. *IEEE Transactions on Software Engineering*, 24(5):315–330, 1998.

[3] M. d'Inverno, D. Kinny, M. Luck, and M. Wooldridge. A formal specification of dmars. In *Proceedings of ATAL'97*, pages 155–176, London (UK), 1998. Springer.

[4] R. Focardi, R. Lucchi, and G. Zavattaro. Secure shared data-space coordination languages: A process algebraic survey. *Sci. Comput. Program.*, 63(1):3–15, 2006.

[5] E. Freeman, K. Arnold, and S. Hupfer. *JavaSpaces: Principles, Patterns, and Practice*. Addison-Wesley Longman Ltd., Essex (UK), 1999. 304 pages.

[6] D. Gelernter. Generative communication in linda. *ACM Transactions on Programming Languages and Systems*, 7(1):80–112, 1985.

[7] R. Milner. *Communication and Concurrency*. Prentice-Hall, 1989. 272 pages.

[8] A. Omicini and F. Zambonelli. Coordination for Internet application development. *Autonomous Agents and Multi-Agent Systems*, 2(3):251–269, 1999.

[9] S. Ossowski, J. Z. Hernández, M.-V. Belmonte, A. Fernández, A. García-Serrano, J.-L. P. de-la Cruz, J.-M. Serrano, and F. Triguero. Decision support for traffic management based on organisational and communicative multiagent abstractions. *Transportation Research Part C: Emerging Technologies*, 13(4):272 – 298, 2005.

[10] V. D. Parunak, R. Savit, and R. L. Riolo. Agent-based modeling vs. equation-based modeling: A case study and users' guide. In *MABS*, pages 10–25, 1998.

[11] C. P. Pfleeger and S. L. Pfleeger. *Security in Computing*. Prentice Hall Professional Technical Reference, 2002.

[12] A. S. Rao. AgentSpeak(L): BDI agents speak out in a logical computable language. In R. V. Hoe, editor, *Seventh European Workshop on Modelling Autonomous Agents in a Multi-Agent World*, Eindhoven, The Netherlands, 1996.

[13] Y. Shoham. AGENT0: A simple agent language and its interpreter. In *Proceedings of the Ninth National Conference on Artificial Intelligence*, pages 704–709. AAAI Press, Menlo Park, CA, 1991.

[14] A. Suna. *CLAIM & SyMPA : An environment for programming intelligent and mobile agents*. PhD dissertation, University of Paris VI, 2005. In french.

[15] M. Wooldridge and N. R. Jennings. Intelligent agents: Theory and practice. *Knowledge Engineering Review*, 10(2):115–152, 1995.

[16] P. Wyckoff, S. McLaughry, T. Lehman, and D. Ford. Tspaces. *IBM Systems Journal*, 37(3):454–474, 1998.

[17] M. Zargayouna. *Modèle et langage de coordination pour les systèmes multi-agents ouverts*. PhD thesis, University of Paris-Dauphine, Paris (France), Dec. 2007. In french.

[18] M. Zargayouna, F. Balbo, and J. Saunier. Agent information server: a middleware for traveler information. In O. Dikenelli, M.-P. Gleizes, and A. Ricci, editors, *Engineering Societies in the Agents World VI*, volume 3963 of *LNAI*, pages 3–16. Springer-Verlag, 2006.