Automatic Creation of Contextual Knowledge in Simulated Agents

Hans K. Fernlund and Avelino J. Gonzalez

Intelligent Systems Laboratory
School of Electrical Engineering and Computer Science
University of Central Florida
Orlando, FL 32816-2450
hfe@du.se, gonzalez@pegasus.cc.ucf.edu

Abstract

Modeling human behavior can be complicated and expensive. To be able to reduce costs, new methodologies and tools must be developed that automate the creation of human behavior models. In this paper we describe one way to accomplish this through the use of Genetic Programming in conjunction with Context-Based Reasoning (CxBR). Context-Based Reasoning is based on the concept that humans think and act in terms of contexts. Genetic Programming (GP) addresses computer programs that evolve new, better programs by themselves, i.e. automatic programming. This paper presents a new approach for automatically creating human behavior models through learning by observation. This strategy learns the behavior of a subject matter expert by merely observation his/her performance in a simulator.

Introduction

Building human behavior models is very complex and time-consuming. Extracting and processing tactical knowledge from the subject matter expert is a very intricate task. To get the expert to express the behavior in an articulate way, analyze the information, and then implement it in an agent are time consuming tasks and clear sources of misinterpretation. It is almost impossible to develop a mathematical formalism of human behavior, and the cost and effort to build good models can be very high. In the real world, there often exist problem domains where the knowledge might be incomplete, imprecise or even conflicting. Often, the models are built on inflexible doctrines. This can cause the entities to behave "too perfectly" wit[1]hout human similarities (Henninger et al., 2000). It has also been shown that the manual routines taught by the experts, are not necessarily the routines used by the experts themselves (Deutsch, 1993).

The use of a learning system that could automatically extract knowledge and construct a behavior model could reduce the problems mentioned above. If the system would be able to observe a human's behavior and automatically build a behavior model the development cost could be dramatically reduced.

This paper presents an approach to building human behavior models automatically. The approach employs Context-Based Reasoning (CxBR) and Genetic Programming (GP) to implement learning by observation, which will learn the behavior of a human merely by observation. First, the objectives of learning by observation are addressed. Then CxBR and GP are briefly presented. Finally, experiments and results are examined and the conclusions are presented.

Learning by Observation

Learning by observation has its roots in biology. Studies have shown that humans fully develop observational learning by the age of 24 months (Abravanel, Ferguson, 1998). By that age, children can easily learn a simple task by observing another person performing the task.

Inspired by how humans and other mammals learn by observation, the machine learning community has developed a number of theories on learning by observation, applied in different areas. The interest in this research is to investigate learning human behavior by observation. The intent is not only to use the observations to learn, but also to learn the behavior of the observed entity. Thus, interest herein is to observe a human in action to be able to model his behavior. A number of advantages could be gained by using learning by observation instead of traditional knowledge acquisition and development methods.

- Reduce time and cost of development, debugging and maintenance.
- Potential to incorporate learning from experience.
- Potential to incorporate new features of humanness, such as emotions.
- Relax operators programming skills.
- Develop simulated entities in real time.

The research described in this paper defines learning by observation as follows:

The agent adopts the behavior of the observed entity only through the use of data collected through observation.

Models of human behavior encompass several different features of humanness. These features range from rather basic behavior such as goal-driven behavior and planning to more complex behavior such as overlapping of performance of multiple high-level tasks, multi-agent coordination, communication, temporal reasoning and maintaining episodic memory. To create a model with these features there must exist an efficient modeling framework. Context-Based Reasoning (CxBR) was proposed by Gonzalez and Ahlers (1993) to have many of these features. The Context-Based representation in CxBR is modeled from the pattern of human behavior. Using CxBR as a framework satisfies the prerequisites for implementation of tactical human behavior. If the model is to be created automatically, the CxBR architecture needs to be equipped with a learning paradigm that will work in conjunction with the architecture without disturbing the supported human features. The learning paradigm used in this research is Genetic Programming (GP). This paper shows that CxBR and GP have the right characteristics to automatically build human behavior models, very synergistically.

Context-Based Reasoning

Gonzalez and Ahlers (1993) presented Context-Based Reasoning (CxBR) as a technique that can efficiently model the behavior of humans in intelligent agents. Further results showed that it is especially well suited to modeling tactical behavior. CxBR is based on the idea that:

- A recognized situation calls for a set of actions and procedures that properly address the current situation.
- As a mission evolves, a transition to another set of actions and procedures may be required to address the new situation.
- Things that are likely to happen while under the current situation are limited by the current situation itself.

CxBR encapsulates knowledge about appropriate actions and/or procedures as well as compatible new situations into hierarchically-organized contexts.

Mission Contexts define the mission to be undertaken by the agent. While it does not control the agent per se, the Mission Context defines the scope of the mission, its goals, the plan, and the constraints imposed (time, weather, rules of engagement, etc). The *Major Context* is the primary control element for the agent. It contains functions, rules and a list of compatible next Major Contexts. Identification of a new situation can now be simplified because only a limited number of all situations are possible under the currently active context. *Sub-Contexts* are abstractions of functions performed by the Major Context which may be too complex for one function, or that may be employed by other Major Contexts. This encourages re-usability. Sub-Contexts are activated by rules in the active Major Context. They will de-activate themselves upon completion of their actions.

One and only one specific Major Context is always active for each agent, making it the sole controller of the agent. When the situation changes, a transition to another Major Context may be required to properly address the emerging situation. For example, the automobile may enter an interstate highway, requiring a transition to an **InterstateDriving** Major Context. Transitions between contexts are triggered by events in the environment – some planned, others unplanned. Expert performers are able to recognize and identify the transition points quickly and effectively.

CxBR is a very intuitive, efficient and effective representation technique for human behavior. For one, CxBR was specifically designed to model tactical human behavior. As such, it provides the important hierarchical organization of contexts. A full description of CxBR can be found in Gonzalez and Ahlers (1998).

This concept has received significant interest from other researchers in the technical literature. Turner (1993, 1999), and Bass (1996) have all independently developed context-based approaches to modeling human behavior.

Genetic Programming

Genetic Programming (GP) is developed from Genetic Algorithms and both are stochastic search algorithms. The search process looks for the best suitable program that will solve the problem at hand. The target system for the GP could be a CPU, compiler, simulation or anything else that could execute the pre-defined instructions. From now on we refer to these as a *program*. GP evolves source code representing a program that can address a specific problem. This makes it very suitable for use with CxBR. GP can build complete software programs that support the internal structure of the CxBR (i.e. the context-base).

To make GP work, some basic requirements must be satisfied. First, we need to have a set of individuals (i.e. programs that represent different solutions to the problem). Furthermore, all the individuals need to be evaluated in some manner as to what degree they are able to solve the problem. Individuals with better suitability would preferably be preserved and survive or breed new individuals for the next generation. The next GP step would be to evolve the individuals (i.e. reproduction) in some manner to preserve the "good" features and develop even better individuals. The most common genetic operators are crossover and mutation. They will support the development and evolution of the individuals. Evolving a program with GP can be described in five steps:

1. Create an initial population of programs (usually randomly generated).
2. Evaluate the performance of each individual through a fitness function.
3. Based on the evaluation, decide which individuals will survive, reproduce or be killed.

4. Apply genetic operations to the individuals selected for reproduction.
5. If the criterion of stopping the process is not met, return to step 2.

The criteria for stopping the evolutionary process can be a maximum number of evaluations made, a maximum number of generations evolved, or the fitness reaching a certain level or other measurable criteria given. When the genetic process is finished in GP, there will exist a program that will solve the problem.

Since GP creates source code, it could be used to incorporate knowledge in any context level or in any instances within CxBR where intelligent behavior is encoded. This means that we could choose to implement learning in any specific part of CxBR and construct the knowledge thereof.

Towards automation

To be able to build models automatically, CxBR needs to be equipped with learning capabilities. Intelligent behavior, within CxBR, can be categorized in two groups, actions and sentinel rules. At different Context levels, the sentinel rules determine which context will be active for that specific context level. The structure of these sentinel rules are similar in all contexts at all levels. Each context has its own set of sentinel rules that determine if this context should still be active or if it should turn over control to another context at the same level. This can be viewed as state transition rules where each state (i.e. context) has its own transition table. At the Major Context level, the action set tends to be more a collection of Sub-Contexts and less of other functions, variables and constants. At lower context levels, the action set is less composed of Sub Contexts and at the lowest context level, there are no Sub Context calls. Experience has shown that three or at most four, levels of contexts, including the Mission Context, are normally sufficient.

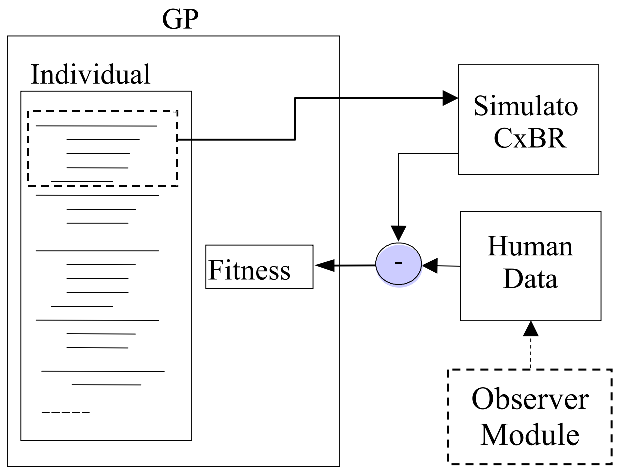

Figure 1. Learning by Observation: CxBR+GP

From the discussion above, we can conclude that if we want to incorporate learning into CxBR, the learning paradigm must be able to learn the proper behavior in a specific context (i.e. actions) and also the appropriate context switches (i.e. sentinel rules). An extra learning feature would be if the system were able to optimize the number of appropriate contexts. This infers that the learning system be able to create new contexts if it seems to be appropriate, and also be able to prune less useful contexts. A learning paradigm that has those features is Genetic Programming (GP).

Instead of creating the contexts by hand, we use the GP process to build the contexts. The GP's evolutionary process provides the CxBR frame with appropriate context actions and rules. The individuals in the genetic population are context parts and a simulator is used to simulate the model's behavior. The behavior from the simulator is then compared with the human performance, and a fitness measure is established to evaluate the models appropriateness (see figure 1). The evolutionary process will strive to minimize the discrepancies between the performances of the contexts created by GP and the human performance. The features of CxBR and GP show that their combination could be a feasible approach to learning tactical behavior by observation. By this combination, a system could be constructed that, by observing a human, builds a context base for simulated entities that exhibit human behaviors'.

A positive feature of using GP as the learning algorithm is that it preserves many of the features of CxBR. The tools GP uses to store the knowledge learned is the same tools a programmer or knowledge engineer would have used in developing the knowledge base, source code statements. This implies that the knowledge is easy to interpret if we want to include communicative features in the agents. The agents are able to express their action in a way that is understandable. It is easy to interpret source code and convert it to written language. It also enables manual coding to be performed in conjunction with the learning algorithm, either before or after learning.

Experiments

The first experiments with this new approach to learning by observation were conducted to model human car driving behaviors. These models were designed to store the knowledge in contexts and be applied to simulated agents. Five different drivers were used to drive a commercial driving simulator in city traffic. Data was collected from 30 minutes of driving from each driver in a realistic environment. The objective was to determine whether the new approach could model the drivers' behavior during normal operation conditions merely from its observation, i.e. learning by observation. Hence, no single scenario was repeated during the run. These first experiments aim is to test the validity of the learning paradigm that combines CxBR and GP. The experiments are organized so that 1) the knowledge within the different context (i.e. action knowledge) and 2) the knowledge that determines the appropriate context to activate (i.e. situational awareness) were learned. If the model should cover the basic city driving, it must be able to handle traffic lights and intersections, besides normal driving on a straight road segment. The prerequisite for the learning is described in figure 2. The action within the context **Urban Driving**, the Sub-Contexts **Traffic-Light-Driving** and **Intersection-Driving** and the Sub-Sub-Contexts **Red-Light-Driving** and **Green-Light-Driving** need to be evolved by the GP algorithm. Additionally, the rules controlling the activation of contexts (i.e. Sentinel Rules) need to be evolved by the GP. The environment for the experiments was set up to ensure that the behavioral patterns of the drivers were neither predictable nor trivial. An example of this unpredictable behavior is when a traffic light changes from green to yellow and then to red. If this change takes place at an appropriate distance from the car, the driver will make a decision on whether to stop when the light turns yellow or if they continue and pass the light while it still is yellow. The distance to trigger this diverse behavior among people seemed to be when the car is 30 meter prior to the light when driving in city traffic. The aim of these experiments was to show the validity of this new approach to automatically build context knowledge from observation. Hence, the aim is not to build the perfect model of a driver but to model the individual driver. So, five different driver models were built. If one of the drivers was a poor driver, there will be a model created of a poor driver. In other words, the experiments succeed when the deviation between the model and the driver observed is very small.

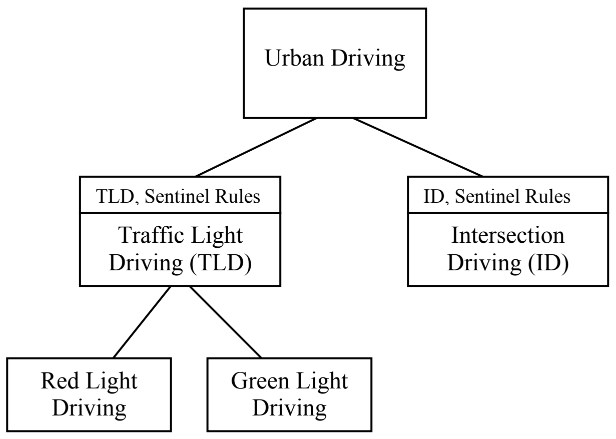

Figure 2. The Context base

Enabling evolution, this hierarchical structure of contexts and sentinel rules that together will represent the behavior pattern, enforces some learning strategy. A strategy in evolving a complex behavior with knowledge stored in a hierarchical structure has been used by Hsu and Gustafson (2001) and called Layered Learning GP (LLGP). The strategy is a bottom up approach where the evolution of the simple behavior in the lower layers takes place prior to evolving the behaviors of the higher layers. In these experiments the first task was then to evolve the behavior at red and green lights. At the higher level of contexts, the Traffic Light Driving context will conclude the combined behavior when a traffic light is approached.

Here the Traffic Light Driving context will include the management of the activation of the lower level contexts within the action knowledge of the context. This way of determine context activation is very similar to the competing context strategy (Saeki and Gonzalez, 2000). The activation of the lower level context will be determined by the parent context's action rule. By including the activation mechanism in the action rules of the higher level context the best fitting context will be activated.

Another way of controlling the context activation is to let each context to be self aware and signal for a possible context switch. This type of context switching is called direct context transition. This type of context switching is a bit more complex for a machine learning algorithm to handle. If one context urges its own activation while another context is active at the same context level, the latter needs to release control before switching can be done. This is because contexts in the same level are mutually exclusive. Therefore, the knowledge in the sentinel rules of contexts at the same level is interdependent. Hence, these sentinel rules need to be evolved in a co-evolutionary fashion, since their behavior affects each other.

The data used when the agent learned the driver's behavior were not preprocessed in any way. To make the learning feasible, the data was reduced from 5000 usable samples per driver to less than 350 samples. These samples need to be complete enough to represent all possible situations within city driving with different state changes of traffic light, intersections and normal city driving.

The data were partitioned to contain similar amount of data from the different scenarios to be used in training. This was done to ensure that the search pressure would not favor any particular situation. The data points were selected randomly from some typical scenarios (e.g. within 100 meters before a traffic light or an intersection). The scenarios are further selected to complete the behavior in the context to be learning. As an example, if Traffic-Light-Driving is to be learned, data from several different scenarios (e.g. stopping at light turning red, running yellow lights, passing traffic lights when making an intersection turn, etc.) need to be included in the training data. The data points for each scenario selected needs to be picked with a constant time frame (e.g. 0.4 seconds) so the algorithm know when to stop the simulation and compare the individual's performance with the human's performance.

Note that the aim here is to get the agent to learn by observation (i.e. only by observing the drivers behavior). The agent is unaware of all traffic rules and regulations and the only way of learning is to mimic the driver's behavior. As an example, the agent is never told to stop at a red light but it will learn this by observing the driver's behavior.

Results

The only type of driving conducted within these experiments was city driving. No learning was focused on turning the car. Examining the data shows that all the drivers keep the car close to the center of the lane. Trying to learn this task would be both trivial and uninteresting. The focus of the learning algorithm is to capture and generalize the behavior of the driver regarding how to apply brake and throttle pressure in the different situations experienced.

To evaluate the results from the experiments, three different evaluation criteria were defined:

- Learning capabilities
- Generalization
- Long term reliability

The first set of tests, *Learning capabilities*, simply measures how well the agent has learned the driver's behavior. This is simply the deviation between the agents' and the drivers' output when fed with the same data used during the training of the agents.

In this research we evolved five autonomous agents. Therefore, it is necessary to test their ability to perform in normal environment (i.e. running in a simulation). It is not sufficient to only test input vectors and compare them with the anticipated output vectors since this will not tell us much about the agents' accumulated errors and their long term reliability. We need to ensure that the agents are autonomous and that the agents' performances in this simulated environment actually are comparable to the drivers' performances even after minutes, hours and days. Except for the Learning Capabilities evaluation, the results are gathered when the agents operate autonomously within the simulated environment. At each simulator cycle, the performances of the agents are compared to the behavior of the drivers at the same position. The deviation in performance could primarily be measured in speed and time deviations. Since the comparison is made at specific locations, it would have taken the agent and the driver some time to get there from the start of the simulation. Hence, there will always be a time deviation between the agent and the corresponding driver at a certain location in the simulation environment, making time an inadequate standard for comparison.

Generalization measures how well the agent can handle new situations not seen in the training data. The agent operates in the simulated environment as described above and is compared with the recorded driver's performance. In this case, however, the driver's recorded performance was not seen by the agent during its evaluation. During *Long term reliability* the agent is tested in a variety of situations during a long term scenario to examine if its behavior is consistent and stable.

Learning Capabilities

When testing learning capabilities of our evolved agents the same data used during training was fed into the learned model and the output is then compared to that of the corresponding driver. The only sound comparison to make here is to look at the speed output. Neither the time nor position will show any deviations of importance since the agent is not operating within a simulator. The results of training capabilities evaluation are shown in table 1.

	Speed deviation		Speed
	[km/h]	%	Correlation
Driver A/Agent A	1.92	3.14%	0.988
Driver B/Agent B	2.03	3.53%	0.983
Driver C/Agent C	1.85	3.41%	0.990
Driver D/Agent D	1.69	2.93%	0.989
Driver E/Agent E	3.81	6.25%	0.852

Table 1. Training error

The results from test of learning capabilities presents low discrepancies between the agents and their respectively driver. The speed deviation is low and the correlation is high. A high correlation, close to one, means that the agent increases and lowers its speed in the same manner as the real driver. Hence, GenCM show good learning capabilities.

Generalization

During the generalization test, the agents operate autonomously in a Context-Based Simulation environment with the same configuration in the simulated world as experienced by the corresponding driver during his simulator run. Even if the data used during training was collected from some sections of the training simulation environment, the agent will never experience the same input pattern as during learning, since the agent is autonomous and will by its action generate its own input state of the next simulator cycle. The agent will also experience at least four totally new traffic lights and three intersection turns never exposed to during training. This could be regarded as validation of the agents. Each agent will run the same route which will take between 140 to 170 seconds to complete depending on the agent's behavior. This equals 1400 to 1700 data samples to be compared to the drivers' behavior. Remember that the agent was only trained with less than 350 data samples.

The first comparison made was a qualitative comparison of the agents' behavior at the traffic lights passed. Table 2 shows the comparison of each agent to its corresponding driver at the different lights. Lights 2, 3, 6 and 7 change from green to red. S stands for stop and R for running the light while it's still yellow. Lights 4, 8 and 4 (at the second pass) changes from red to green. Ok means that the agent performs in accordance with its corresponding driver (i.e. slows down when the light is still red and picks up speed when it turns green). When light 3 is passed the second time it is constantly green, and OK refers to the agent is performing in accordance with its corresponding driver. Here we see that all the agents perform in accordance with their corresponding drivers.

	L 2	L 3	L 4	L 6	L 7	L 8	L3	L4
A	S/S	R/R	Ok	R/R	R/R	Ok	Ok	Ok
B	S/S	S/S	Ok	R/R	R/R	Ok	Ok	Ok
C	S/S	S/S	Ok	S/S	S/S	Ok	Ok	Ok
D	S/S	S/S	Ok	R/R	R/R	Ok	Ok	Ok
E	R/R	S/S	Ok	R/R	R/R	Ok	Ok	Ok

Table 2. Qualitative comparison of the drivers / agents performance

Next, a quantitative comparison between the agents' performance and their corresponding driver was conducted. Table 3 show the average deviations recorded in speed and time over the whole run. The speed correlation is also shown in Table 3.

	Speed [km/h]		**Time [s]**		**Speed**
	RMS	**Std.Dev**	**RMS**	**Std.Dev**	**Corr**
A	5.81	7.35	4.27	4.11	0.825
B	5.38	7.92	1.98	2.79	0.893
C	4.79	6.72	1.48	2.07	0.920
D	6.14	8.45	2.38	3.12	0.842
E	7.43	8.42	3.06	3.72	0.783

Table 3. Relationship between the agents and the drivers

This first validation of the agents indicates that the algorithm is able to generalize the agent's behavior.

Long term reliability

An additional test was conducted where the five agents operated in the simulated environment for 40 minutes, pass more than 60 traffic lights and 25 intersections. Now the agents were exposed to a variety of traffic light scenarios where none was similar to the other. Their behavior was recorded when the light in the agent's proximity was either yellow or red.

Since the traffic lights now change their states at different distances (i.e. the lights are time scheduled and not related agents distance) and agents might approach the lights at different speeds, it is difficult to make an exhausted statistical analysis of their behavior. Anyhow, Table 4 shows a simple compilation of the agents' behavior when they approach lights that is either yellow or red. Two different events occur: the light turns from green to red or from red to green.

A qualitative measure could be performed of the agents' action when the lights turn red. The stopping column in Table 4 shows how many lights the agents stop at, compared to the total number of lights passed turning red. All the agents, except agent D, stop at all lights turning red. Agent D runs three lights when they turn red late (i.e. the light actually turns red before the agent pass the light). Investigating the results more it shows that if the lights turn red when the agent is further away than 27 meters the agent will stop and the occasions where the lights turn red when the agent is closer than 23 meters the agent will run it. Even if the agent some times runs the light, it is consistent and acts the same in similar situations.

	Light turning Red			Light turning Green
	Stopping	Avg.Dist	Std.Dev	
Agent A	20/20	34.7	12.9	20/20
Agent B	22/22	8.04	1.95	22/22
Agent C	25/25	5.89	1.03	8/8
Agent D	31/34	4.50	1.31	6/6
Agent E	22/22	13.5	0.551	11/11

Table 4. Agent's long term behavior

As the agents come to a stop at the red lights, a comparison could be made on their different stopping distances. Table 4 show that all the agents except agent A, stop at almost the same distance every time and therefore their standard deviation on the stopping distance is small. The surprising fact is actually that the other four agents manage to generalize so well that they stop at approximately the same distance, even if the time of light change is different. Remember that in the data presented to the agents during learning, lights changed their state when the driver was 30 meter prior to the light. Hence, all the agents stopped at consistently at the same distance during training (approximately thirty meters after the light turns from green to yellow).

The final observations on the agents' long term behavior are their behavior when approaching traffic lights turning green. Two observations can be made as the agents approaching a red light about to turn green. The first thing that institutes correct behavior of the agents is that they do not stop at the red light when they are far from the light. The other behavior to investigate is that they lower the speed as they get closer and that they pick up speed when the light turn green. The column that describes the correct behavior at a light turning green in Table 4 compares the number of correct behaviors to the total numbers of lights turning green exposed to each agent. All the agents show a correct behavior all the time as they approach a red light about to turn green.

This test has shown that the agents show consistent and stable performance at traffic lights throughout the long term stability test.

Result Summary

The training results show that the learning paradigm is able to successfully capture the behavior of the five drivers. The only agent that shows somewhat deficient performance is agent E. The data was not preprocessed in any way other than being reduced to a reasonable size. Hence, driver E shows some inconsistent behavior at some traffic lights. At one light turning from green to red when he approaches at low speed, he actually picks up speed and pass it while it is yellow. This occasion was not consistent with his behavior at other lights and was treated as an outlier.

We conclude from Tables 2 and 3 that the agents have shown the ability to generalize their behavior. The qualitative comparison shows that they have the same behavior as their corresponding driver at all the traffic lights and the deviations in table 3 is fairly low and the speed correlation fairly high.

The long term reliability test shows that the agents created are capable of learning and generalizing the behavior of the driver. They are also able in a stable fashion within the simulated environment.

Conclusions

Our research shows that the approach to learning by observation using GP and CxBR are able not only to learn but also to generalize the behavior pattern to be used in new situations. We have shown the ability to use Genetic Programming to automatically create context knowledge only by observing a subject matter expert. The algorithm to automatically build contextual knowledge also manages to produce agents with consistency and long term reliability.

References

Abravanel, E., Ferguson, S. (1998) “Observational learning and the use of retrieval information during the second and third years” Journal of Genetic Psychology; Dec 1998, v.159, 4, 455(2)

Bass, E. J., Zenyuh, J.P., Small, R.L. and Fortin, S.T. (1996). “A Context-based Approach to Training Situation Awareness”, Proceedings of the Third Annual Symposium on Human Interaction with Complex Systems, Los Alamintos, CA: IEEE Computer Society Press, pp. 89-95

Deutsch, S. (1993) “Notes Taken on the Quest for Modeling Skilled Human Behavior” Proceedings of the Third Conference on Computer Generated Forces and Behavioral Representation, Orlando, FL, March 17-19, pp.359-365

Gonzalez, A. J. and Ahlers, R. H. (1993) "Concise Representation of Autonomous Intelligent Platform in a Simulation Through the Use of Scripts", Florida Artificial Intelligence Research Society Conference, 1993.

Gonzalez A. J. and Ahlers, R. H. (1998) "Context-Based Representation of Intelligent Behavior in Training Simulations” Transactions of the Society for Computer Simulation International, volume 15, no 4, December 1998

Gonzalez A. J. Georgiopoulos M. DeMara R. F. Henninger A. Gerber W. (1998) "Automating the CGF Model Dvelopment and Refinement Process by Observing Expert Behavior in a Simulation" Proceedings of the Computer Generated Forces Conference, Orlando, 1998

Henninger A., Gonzalez A., Gerber W. Georgiopoulos M., DeMara R. (2000) “On the Fidelity of SAFs: Can Performance Data Help?” Proceedings of the Inter-service/Industry Simulation and Education Conference, Orlando, FL, 2000.

Holland, J.H. (1975) "Adaptation in Natural and Artificial Systems", University of Michigan Press, 1975. Reprinted by MIT Press, 1992

Hsu, W.H., Gustafson, S.M. “Genetic Programming for Leyered Learning of Multi-agent Taks”, Proceedings of the Genetic Evolutionary Computation Conference, GECCO 2001, San Francisco, CA, July 9-11, 2001

Koza, J. R. (1992) "Genetic Programming", MIT Press, 1992, ISBN 0-262-11170-5

Luke, S. (1998) “Genetic Programming Produced Competitive Soccer Softbot Teams for RoboCup-97”, Proceedings of the Third Annual Genetic Programming Conference, p.204-222, Morgan Kaufmann, Los Altos, CA, 1998

Mendes, R.F. Voznika, F.B. Nievola, J.C. Freitas, A.A. ”Discovering Fuzzy Classification Rules with Genetic Programming and Co-Evolution”, Proceedings of the Third Annual Genetic Programming Conference, p.204-222, Morgan Kaufmann, Los Altos, CA, 1998

Schaal, S., (1999) “Is imitation learning the route to humanoid robots?” Trends in Cognitive Sciences,3(6), 233-242.

Turner, R.M. (1993) “Context-sensitive reasoning for autonomous agents and cooperative distributed problem solving”, Proceedings of the 1993 IJCAI Workshop on Using Knowledge in Its Context, Chambéry, France

Turner, R.M. (1999). “A Model of Explicit Context Representation and Use for Intelligent Agents”, Springer-Verlag, 1999

van Lent, M. and Laird, J. (1998). Learning by Observation in a Tactical Air Combat Domain, In Proceedings of the Eighth Conference on Computer Generated Forces and Behavior Representation. Orlando, FL., May, 1998

Wolpert D.H. and Macready, W.G. (1995) “No free lunch theorems for search”, Technical Report SFI-TR-95-02-010, 1995.

Wolpert D.H. and Macready, W.G. (1997) “No free lunch theorems for optimization. IEEE Transactions on Evolutionary Computation, Vol. 1, No. 1, pp. 67-82, 1997