Proceedings of the 2006 IEEE/RSJ
International Conference on Intelligent Robots and Systems
October 9 - 15, 2006, Beijing, China

A Robocentric Motion Planner for Dynamic Environments Using the Velocity Space

Eduardo Owen †‡
‡*Escuela de Ingeniería Eléctrica y Electrónica*
Universidad del Valle
Calle 13 No. 100-00, Cali, Col
efowen@unizar.es

Luis Montano †
†*Instituto de Investigación en Ingeniería de Aragón*
Dep. Informática e Ingeniería de Sistemas, Universidad de Zaragoza
María de Luna 3, E-50018 Zaragoza, Spain
montano@unizar.es

***Abstract*— This paper addresses a method to optimize the robot motion planning in dynamic environments, avoiding the moving and static obstacles while the robot drives towards the goal. The method maps the dynamic environment into a model in the velocity space, computing the times to potential collision and potential escape and the associated robot velocities. The problem of finding a trajectory to the goal is stated as a constrained nonlinear optimization problem. The initial seed trajectory for the optimization is directly generated in the velocity space using the model built. The method is applied to robots which are subject to both kinematic constraints (i.e. involving the configuration parameters of the robot and their derivatives), and dynamic constraints, (i.e. the constraints imposed by the acceleration/deceleration capabilities). Some experimental results are discussed.**

I. Introduction

This work addresses a method to plan robot motions in dynamic environments. It concerns to find a trajectory from start to goal that satisfies: i) avoids obstacles in the environment, ii) the trajectories are feasible (kinematic and dynamic constraints of the robot), and iii) minimizes a criterium (i.e., motion time). The collision avoidance problem in motion planning has been extensively treated in the robotics literature and recent methods based on reactive avoidance already consider some of these constraints. When the available velocity information of the moving objects is taken into account, the navigation system can compute more stable trajectories which improve the motion performance regarding others classical methods for motion planning. The information gathered by the sensors of the robot which reflects the dynamism of the environment it is necessary to plan optimal or near-optimal trajectories.

This way we use a technique to plan robot motions which transform the problem from the workspace or configuration space to the velocity space in order to make decisions about the ”best” strategy of motion directly in this space. The method maps the positions of the obstacles and their known or estimated trajectories into the velocity space of the robot taking into account its kinematic and dynamic constraints. It provides information about further collisions, providing a map of all the collision-free velocities available. An initial collision-free trajectory is computed on the model, which is utilized as a seed in the optimization procedure to find a trajectory which converges to the goal. This process is resumed every sampling time.

This paper is organized as follows: in section II some related works are presented. In section III the developed approach is outlined. Section IV presents the method to map the configuration space of a dynamic environment to the velocity space. The robocentric motion planning algorithms are addressed in V. The optimization of trajectories using the velocity space built is presented in section VI. Simulation results are discussed in section VII and in section VIII some conclusions are presented.

II. Related Works

The classical reactive methods such as [21], [3], [11], [20] are sufficient to keep the robot safe, but they are very sensitive to local minima. The common approach is to incrementally build a occupance grid, which assumes static obstacles and is updated from sensor data. These approaches usually exploit their reactivity to deal with moving obstacles. But they do not use the kinematic information of the objects to cope with the problem of the robot velocity planning.

The motion planning to reach a goal in a dynamic environment is resolved either using global planning methods or using local planning with a reactive obstacle avoidance method. This problem was originally addressed by adding the time dimension to the space in which general motion planning techniques were used ([15], [6], [4] are some examples).

[1] takes into account explicitly the velocities of the obstacles. The idea is to represent the obstacles directly in the velocity space of the robot, in order to compute the set of velocities leading to collision. In this work the robot and the obstacles have straight constant trajectories, extended in [18] and [19] to arbitrary obstacle trajectories. [12] defines a model to represent the dynamic environment and the non-holonomic and dynamic constraints of the robot in the velocity space (w, v), which allows to compute motion

1-4244-0259-X/06/$20.00 ©2006 IEEE

commands directly in this space, selecting them among the velocities not leading to collision. In [5] the concept of Inevitable Collision States is defined, corresponding to states for which no trajectory exists for the system can avoid the collision, and are used to plan trajectories ICS-free. But the ICS computation is a complicated task. In [14] A Partial Motion Planner for dynamic environments is proposed to be executed in real-time. The ICS concept is also used, building a tree using probabilistic techniques.

The problem of characterizing minimum-time trajectories linking any par of configurations where the robot is at rest has been studied by Jacobs in [7]. The authors have proven that minimum-time trajectories correspond to trajectories obtained by means bang-bang control, leading to pieces of clothoids and involute of circles. The problem is treated in free-space. [16] deals with time optimal control for mobile robots computing extremal controls as the optimal ones. They stated that it is an open problem. Fleury in [2] addresses the problem of smoothing mobile robot motions, proposing several sub-optimal strategies to smooth broken lines trajectories in a cluttered, but no dynamic, environment. On the other hand [13] presents a method for computing the time optimal trajectories of a robot manipulator moving in a dynamic environment by utilizing the concept of velocity obstacle, where the trajectory is computed using a steepest descent algorithm. It computes the switching times for an optimal bang-bang control. In [10] a system composed by a local goal-oriented obstacle avoidance method which uses the concept of non-linear velocity obstacle and an incremental global planner is developed. The local method optimizes a criterium which weights the velocities, the orientation change, a risk function and the time to collision. But the convergence to the goal is difficult to obtain because of the weighting parameters.

In [17], smooth paths composed by curves which maintain the curvature continuity (clothoids and arc of circles), obtaining feasible trajectories are proposed. But no time considerations are made. [9] computes near-optimal trajectories under dynamic and kinematic robot constraints using piece-wise trajectories which maintain curvature continuity, but no in a dynamic environment context. These trajectories are based in the work of [17], and the cost metric is based on the duration of the control.

III. Statement of the problem and the approach

The approach presented in this paper focuses in motion planning in dynamic environments. We assume that a global planner provides to the system a sequence of locations (subgoals) to reach the final goal (for instance a D^* algorithm can be used). We don't deal here with the global planning problem. The work is centered in computing trajectories to reach the subgoals, minimizing a time criterium.

We state the problem to solve as a constrained optimization one, in which the restrictions come form the robot itself (kinodynamic constraints) but also from the kind of paths selected to be followed by the robot and from the obstacles moving around the vehicle. The system can profit the information about the dynamism of the environment to compute the best motion, in terms of paths and velocities (that is, trajectories). The motion is computed by means a robocentric motion planner in the velocity space, which allows to plan the trajectories in every sampling period. This Planner is described in Section V. It utilizes an optimization technique to compute trajectories which converge to the goal, taking into account all the constraints involved in the problem. The method plans a trajectory which minimizes the time to reach the goal, but constrained to some paths which assure the continuity of the curvature. Only the optimal command for the next sampling period is applied, and a new optimization to the goal is made in every period. So the motion can comply with the changing environment, and the system plans again observing the new conditions.

We impose conditions to the trajectories to be followed by the robot. Not any geometric path is permitted. The kind of paths selected are: clothoid, anti-clothoid (involute), circular, and straight lines. Note that clothoids and anti-clothoids at maximum acceleration correspond to extremal controls. This kind of sequences does not correspond to minimum time trajectories (are not sequences of extremal controls), but allows to always have a continuous curvature in the paths followed by the robot, given as a result feasible trajectories, compatible with the acceleration/deceleration constraints.

IV. Modelling the Environment

In [12] a technique to map the dynamism of the environment was presented. This model is used in this work as the basis to compute the trajectories. We outline briefly some ideas of this model.

The mapping of the dynamic environment is based on computing the robot paths and velocities (trajectories) that would provoke further collisions with the objects. The collision time is also a relevant information implicit in the mapped space. We consider here the case of non holonomic robots. Besides, some constraints are imposed in this paper in order to present the method:

- The model is composed by *straight or circular paths* of the robot. This is a common constraint imposed to non holonomic robot motions. In this way, we take into account the kinematic constraints.
- In this paper the objects move with a constant velocity following straight paths. Anyway, the method can be easily extended to other kind of paths, thus there is not loss of generality.
- The moving objects are represented as polygons and the robot is considered as circular, reducing the complexity of computations in the Configuration Space.

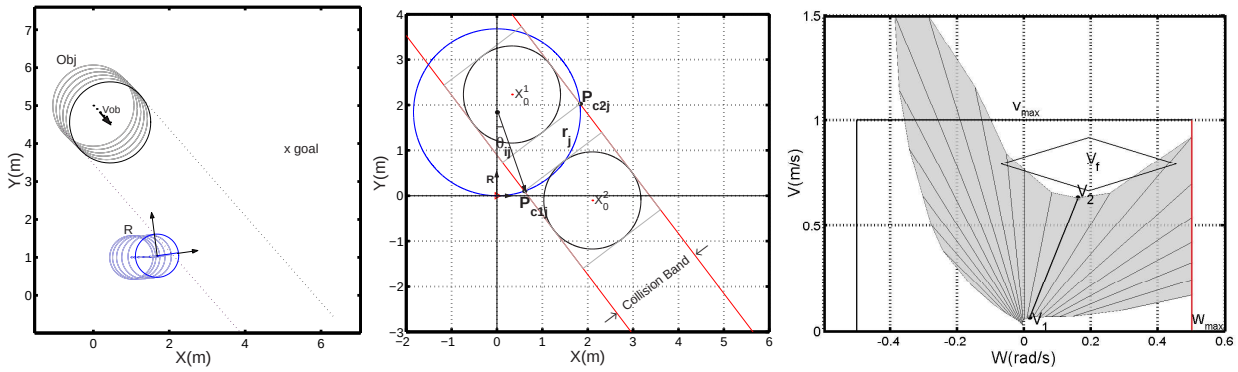


Fig. 1. (a) Workspace, (b) collision band, path r_j and collision points P_{c1j} and P_{c2j} in the Configuration Space (c) projection of $DOVS$, V_{DOVS}, on the plane (w, v)

The model is based on the idea of mapping the motion of the static and moving objects of the environment from the workspace to the velocity space of the robot. For this all the computations are made on the local reference of the robot (robocentric representation). Figure 1a represents the workspace (WS) with a robot and a moving obstacle at constant velocity following a straight path. Figure 1b depicts the configuration space (CS) at instant k. It shows the Collision Band (zone swept by the object moving along a straight line) and an obstacle in two locations ($\mathbf{x_o}^1$ and $\mathbf{x_o}^2$) representing the locations in which the robot, following a circular trajectory r_j, arrives when the object has just passed at time t_{1j} (point P_{c1j}) or escapes from collision crossing just before the object arrives at time t_{2j} (point P_{c2j}). The computation of this pair of points and their associated times for the set of trajectories that can lead to collision is the basis of the proposed model. Then the robot velocities (w_1, v_1) and (w_2, v_2) and their corresponding times t_1 and t_2 are computed and mapped in the velocity space of the robot (Figure 1b). These calculations are extended to the whole space, considering a range of curvature radii forming a surface in the velocity-time space (Dynamic Object Velocity, DOV). The union of all the zones of velocities DOV for all the objects provides the Dynamic Objects Velocity Set ($DOVS$) and represents the velocities for which could have collision if they were maintained for some time. In Figure 1c, the projection of $DOVS$, V_{DOVS} on the plane (w, v) is shown. The lower limits involve the maxima robot velocities to allow the object pass before the robot (i.e. V_1) and the upper limits represent the minima robot velocities to escape before the objet pass (i.e. V_2). As a consequence, choosing a velocity outside V_{DOVS} implies that the robot won't collide during the whole time horizon considered by the computation. Notice that the circular paths in WS are represented as straight lines in VS, whose origin lies the origin of the plane (w, v). This property makes easier to plan velocities and trajectories in this space.

V. The Robocentric Motion Planner

In this section we describe the Robocentric Planner (RP). The method developed follows the steps:

1) it computes the environment model, mapping the robot and moving objects trajectories into the velocity space (VS) (Section IV)
2) it computes a circular trajectory from the current robot location to the next subgoal, to select a feasible trajectory used as a heuristic for setting a seed for the optimization problem
3) it computes a trajectory towards the subgoal by solving a constrained optimization problem, in which all the restrictions are taken into account (Section VI)
4) it applies the next motion command to follow the trajectory solution and repeats cyclicly the process every sampling time, until convergence to the goal.

The reason to compute a circular trajectory in the second step is that the model reflects the free velocities the robot can choose without collision, when it follows that kind of paths (the straight line is a particular case). But this trajectory is only a seed for the next step, in which it will be optimized.

The final trajectory will be composed by a sequence of paths as the presented in section III (for instance Clothoid-Circular-Clothoid-Straight, Cl-C-Cl-S). These kind of paths have been used in previous works (i.e. [9], [17]) but in our case they are parameterized in time, because of we need it for time optimization.

Two situations can arise: 1) upper velocities are free ($S1$); 2) upper velocities are not free ($S2$). Using the representation in VS we can provide an intuitive explanation for these cases and the motion strategies selected for each.

A. Upper velocities free

Figure 2a represents the situation $S1$. It corresponds to the case in which high velocities (in the upper part of VS) can be chosen, that is the robot can pass before the moving objects arrive. The robot is at the current velocity $V1$. RP chooses one of upper free velocities, $V3$, using the

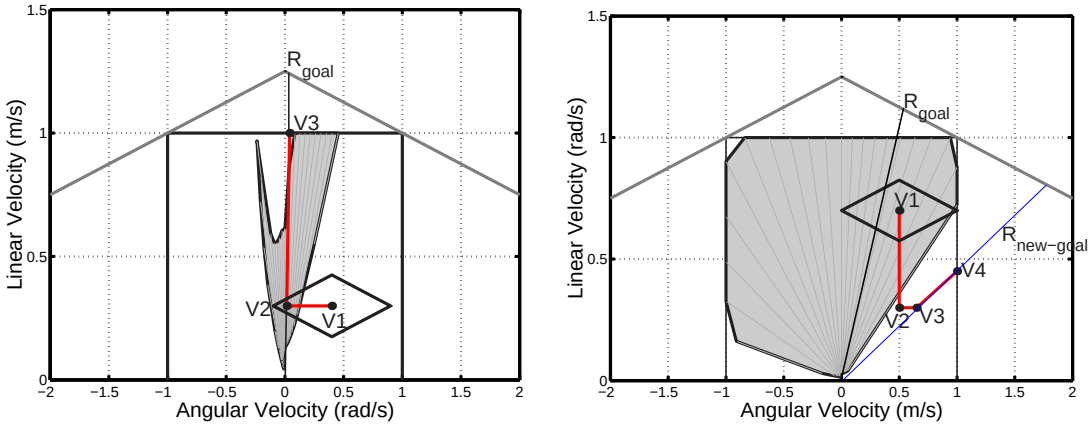


Fig. 2. (a) trajectories and V_{DOV} in the VS in situation $S1$, (b) trajectories and V_{DOV} in the VS in situation $S2$. The square represents the maximum linear and angular velocities reachables

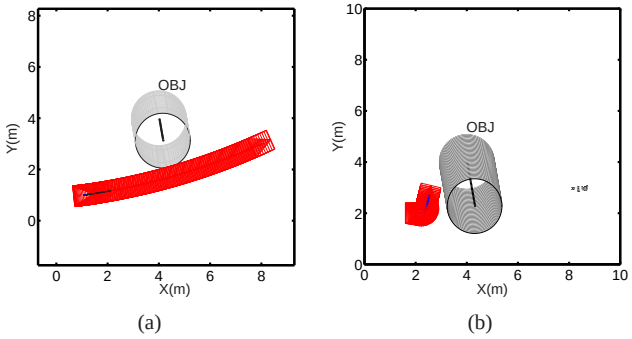


Fig. 3. (a) corresponding trajectory in WS is $S1$; the robot speed up to pass before the object, (b) corresponding trajectory in WS in $S2$; the robot decelerates to wait the object passes

criterium of proximity to the goal mapped (R_{goal}, in this case it match the line $V2-V3$) within the no-collision zone. As the vehicle has dynamic constraints, it is possible that it cannot reach instantaneously the new circular trajectory (line ($V2-V3$)). To do that, the system computes from the current velocity $V1$ a Clothoid trajectory ($V1-V2$), joining the initial and the final circular paths maintaining a continuous curvature. Notice that this circular trajectory has two pieces: the first one is followed with acceleration (linear and angular) reaching $V3$, and the second one is followed at maximum constant linear and angular velocities, $V3$. The times associate to each stretch are also computed to be further used in the optimization process. Every time RP verifies that the composed trajectory is collision free. Figure 3a shows the corresponding trajectory in the workspace.

B. Upper velocities not free

Figure 2b represents the situation $S2$. This is the case in which the current velocity leads to collision if it is maintained and the upper velocities are prohibited. Thus, a safe solution is taken, selecting a circular trajectory outside the zone of dangerous velocities ($R_{new-goal}$). Several solutions can be considered, for instance, to reduce the velocity to $V2$. This path is an anti-clothoid, a vertical straight line in VS. To link the new circular trajectory, a clothoid $V2-V3$, then a circular stretch on the $R_{new-goal}$ ($V3-V4$), and finally a constant velocity circular trajectory are computed. This allows to avoid the moving obstacles, reducing the velocity until the objects pass. After the object passes, the situation $S1$ is resumed. Figure 3b shows the corresponding trajectory in WS.

Table I shows the main algorithm for RP. The inputs are the Goal, the number of objects, the objects, the current robot velocity, and the output is a feasible trajectory. Function map_{obj} maps the obstacles in VS, $merge_{obj}$ orders the objects as a function of the time to collision, $detect_{obj}$ selects situations $S1$ or $S2$, $goal_{map}$ maps the trajectory to the Goal into VS, new_{goal} computes a new $R_{new-goal}$ for $S2$, $generate_{traj}$ generates the seed trajectory for the further optimization and $optimize$ computes an optimal trajectory to the goal. In the next Section we present how this optimal trajectory is computed.

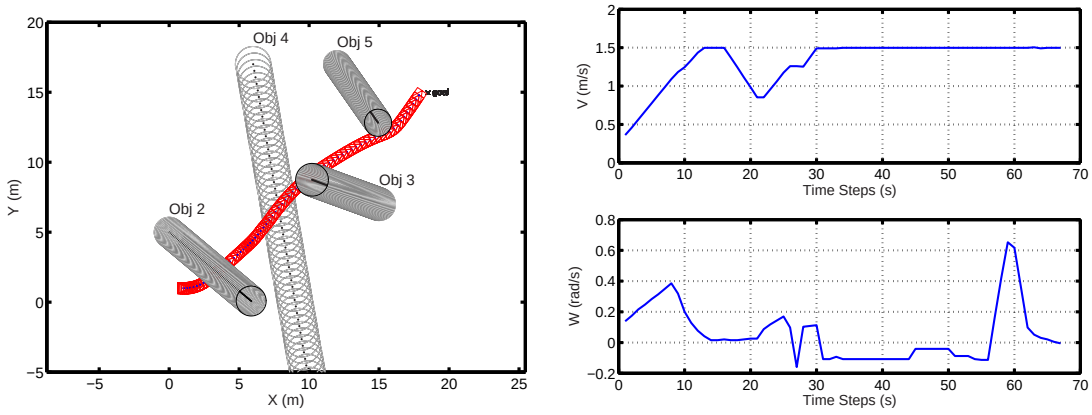
C. Kind of Trajectories

The trajectory computed in the previous section is used as a seed for the optimization algorithm assuring convergence to the goal. We present next the kind of trajectories used, the function to be optimized and the constraints imposed to the optimization.

As said above we impose compound trajectories to be followed by the robot, maintaining a continuous curvature. Depending on the system is in $S1$ or $S2$, the trajectories are different:

- $S1$: Cl-C_{ac}-C_{un}-Cl-S
- $S2$: ACl-Cl-C_{ac}-C_{un}

where Cl means Clothoid, ACl Anticlothoid, C_{ac} circular trajectory with acceleration, C_{un} circular trajectory with uniform (maximum velocity), and S straight line.



(a) (b)

Fig. 4. (a) trajectory when the plan is re-computed every sampling time, (b) Linear and angular velocity profiles in the first experiment

TABLE I

ALGORITHM: ROBOCENTRIC MOTION PLANNER

Input: *goal*, num_{obj}, *obj*, V_o
Output: $traj_{opt}$

begin
repeat
$[DOVs] = map_{obj}(obj)$;
$[k, DOVk] = merge_{obj}(num_{obj}, DOVs)$;
$[situation] = detect_{obj}(k, DOVk)$;
if situation $= 1$
$[R_{goal}] = goal_{map}(goal)$;
$[traj] = generate_{traj}(R_{goal}, k, DOVk, V_o)$;
else
$[R_{new-goal}] = new_{goal}(goal, k, DOVk)$;
$[traj] = generate_{traj}(R_{new-goal}, k, DOVk, V_o)$;
$[traj_{opt}] = optimize(traj, V_o)$;
end
until $reached(goal)$
end

VI. OPTIMIZATION OF TRAJECTORIES

The optimization problem can be formulated as follows:

$$t^* = argmin_{(t)} F(\mathbf{x}(t))$$

where $\mathbf{t} = [\ \sum_{j=1}^{k} t_j\ ]$ is a time vector whose components are the times of each stretch of the whole trajectory, and $F(\mathbf{x}(t))$ is a function representing the sequence of sub-trajectories parameterized in time (see Appendix). This function depends on several parameters: maxima accelerations, number of sub-trajectories, initial location and velocity, and geometric parameters of the robot.

The constraints are:

- nonlinear equality constraints:

$$x_k - x_{goal} = 0 \quad (1)$$
$$y_k - y_{goal} = 0 \quad (2)$$
$$\sum_{j=1}^{k} L_k(w_o) = 0 \quad (3)$$
$$\sum_{j=1}^{k} L_k(v_o) = 0 \quad (4)$$

where k is number of stretches, (1) and (2) lead the search towards the goal, (3) and (4) considers the continuities in curves.

- nonlinear inequality constraints:

$$\sum_{j=1}^{k} f(w_j, v_j) \geq 0 \quad (5)$$
$$V_k \geq 0 \quad (6)$$
$$\sum_{j=1}^{k} t_j \geq 0 \quad (7)$$
$$\sum_{q=1}^{num} (x - f_{xq})^2 + (y - f_{yq})^2 - r_q^2 \geq 0 \quad (8)$$

where num is the number of moving obstacles, (5) considers the constraints for the linear and angular velocities in VS (in our case for a differential-drive robot), (6) imposes positive linear velocities, (7) enforces solutions in positive times, and (8) reflects the constraints coming from the moving objects in the CS modelled as circles. f_{xq} and f_{xq} are functions of time, and r_q is the object radius enlarged with the robot radius.

VII. EXPERIMENTAL RESULTS

The objectives of the experiments in this section is to show how the planner works in different conditions. It has to adapt the decision to the dynamic of the objects and the robot dynamics. In the first experiment (Figure 4a), the planner re-computes the optimal trajectory every sampling time, yielding a smooth trajectory to the goal. It can be note that the planner makes a decision of waiting the object $O4$ passes and speeding up to pass before the objects $O2, O3, O5$ arrive. Figure 4b shows the velocity profiles. The robot reduces

the linear velocity when the object $O4$ moves towards it, permitting the object passes. The robot moves at maximum linear velocity to pass before $O2, O3$ and $O5$ arrive. Notice that to do it the robot has to manoeuver, avoiding collision.

In the second experiment the direction and the velocity of the objects were lightly changed, thus the plan differs from that of the previous experiment. The robot pass before all the objects arrive (Figure 5a), adapting the plan to the new situation. In Figure 5b the velocity profiles are depicted. It can be seen that during the initial acceleration period, stretches of circular and clothoid (linear velocity constant) trajectories are alternated, due to the planning every sampling time. It can be appreciated that the robot goes towards the goal at maximum linear velocity, rotating when needed to avoid the coming objects.

As conclusion, the robocentric local planner works well with different dynamics in changing environments. In a such scenario, classical obstacle avoidance methods would lead to oscillatory motions. The method drives the robot to the subgoals or goal, trying to generate motions at maximum velocity to improve the travelling time. The kind of trajectories selected assure that the motions are feasible and maintaining the continuity in the curvature.

VIII. Conclusion

A robocentric robot motion planner for dynamic environments has been reported. The planner minimizes a time to the goal criterium, constrained to some kind of trajectories which assure the curvature continuity, to the robot kinematic and acceleration restrictions, and to the velocities which avoids collisions with the objects.

The method computes trajectories from the current robot state (location and velocities) to the goal, which are recomputed every sampling time to comply with the changing environment and the new robot location and velocities. A model representing the moving objects around the robot in the Velocity Space allows to select collision-free velocities to initialize the local optimization procedure, computing the next velocity command.

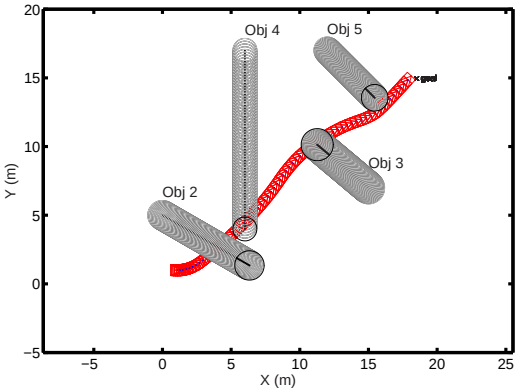
As further work, we propose a global planner which exploits the time-velocity information included in the model built, to be used jointly to the robocentric planner herein presented. Also, other sequences of trajectories have to be analyzed, in order to compute time optimal or near-optimal motions.

Acknowledgments

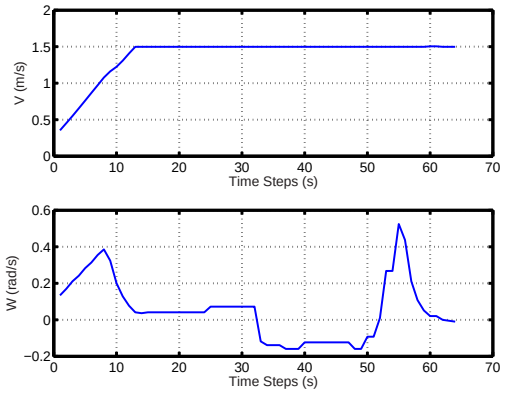
The work has been partially funded by the Spanish MCYT-FEDER DPI2003-07986 project.

Appendix

In this appendix we present the equations of curves for the general case from a initial configuration $C_0 = (x_0, y_0, \theta_0)$ and initial velocities v_0 and w_0 .



(a)



(b)

Fig. 5. (a) Trajectory planned in another scenario, with different object direction and velocities, (b) Linear and angular velocity profiles in the second experiment

A. Robot motion equations

The equations for a differential-drive robot (our case) are:

$$x(t) = x_0 + \int_0^t v(u)\cos\theta(u)du \quad (1)$$

$$y(t) = y_0 + \int_0^t v(u)\sin\theta(u)du \quad (2)$$

$$\theta(t) = \theta_0 + \int_0^t w(u)du \quad (3)$$

B. Equations of a Clothoid

We consider case $a_r = -a_l$. a_r and a_l represent the accelerations of the wheels. Now, $a_r = \pm a$, while $a_r = \mp a$. In this case $\dot{v}(t) = \frac{1}{2}(a_r + a_l) = 0$. Then $v(t) = v_0$. By integration $v_r(t) = \pm at + v_{ro}$, $v_l(t) = \mp at + v_{lo}$ and $w(t) = \dot{\theta}(t) = \pm\frac{2a}{d}t + w_0$. The expressions for $v(t)$ and $w(t)$ are replaced in (1), (2) and (3). The coordinates for the robot are given by the Fresnel Integrals (SF and CF stand for Sinus and Cosinus of Fresnel), respectively (See [8] for instance):

$$\begin{aligned} x(t) = \; & x_0 - \tfrac{1}{2}v_0\sqrt{2}\sqrt{\pi}\{-CF(*4)(*1)\cos\theta_0 \\ & -CF(*4)(*3)\sin\theta_0 - SF(*4)(*3)\cos\theta_0 \\ & +SF(*4)(*1)\sin\theta_0 + CF(*2)(*1)\sin\theta_0 \\ & +CF(*2)(*3)\sin\theta_0 + SF(*2)(*3)\cos\theta_0 \\ & -SF(*2)(*3)\sin\theta_0 + SF(*2)(*3)\cos\theta_0 \\ & -SF(*2)(*1)\sin\theta_0\}/\sqrt{\tfrac{a}{d}} \end{aligned}$$

$$y(t) = \; y_0 - \tfrac{1}{2}v_0\sqrt{2}\sqrt{\pi}\{SF(*4)(*1)\cos\theta_0$$

$$
\begin{aligned}
&+SF(*4)(*3)\sin\theta_0 - CF(*4)(*3)\cos\theta_0 \\
&+CF(*4)(*1)\sin\theta_0 - SF(*2)(*1)\cos\theta_0 \\
&-SF(*2)(*3)\sin\theta_0 + CF(*2)(*3)\cos\theta_0 \\
&-SF(*2)(*3)\sin\theta_0\ \}/\sqrt{\tfrac{a}{d}}
\end{aligned}
$$

$$\theta(t) = \quad \theta_0 + (\tfrac{at^2}{d} + w_0 t)$$

where $(*1) = \cos(\frac{1}{4}\frac{dw_0^2}{a})$, $(*2) = \frac{1}{2}\frac{\sqrt{2}w_0}{\sqrt{\pi}\sqrt{\frac{a}{d}}}$, $(*3) = \sin(\frac{1}{4}\frac{w_0^2 d}{a})$, and $(*4) = \frac{1}{2}\frac{\sqrt{2}(2at+w_0 d)}{\sqrt{\pi}d\sqrt{(a/d)}}$

C. Equations of a Anti-Clothoid

We consider the case $a_r = a_l$. Then $\dot{w}(t) = \frac{1}{d}(a_r - a_l) = 0$ and $w(t)$ is a constant and equal w_0. The acceleration $a(t)$ equals $sgn(a_r)at$. Thus $v(t) = sgn(a_r)at + v_0$.

$$
\begin{aligned}
x(t) = \quad & x_0 + \{a\cos(w_0 t + \theta_0) + a\sin(w_0 t + \theta_0) w_0 t \\
& + v_0 \sin(w_0 t + \theta_0) w_0 - a\cos\theta_0 \\
& - v_0 \sin(\theta_0) w_0)\}/w_0^2. \\
y(t) = \quad & y_0 - \{-a\sin(w_0 t + \theta_0) + a\cos(w_0 t + \theta_0) w_0 t \\
& + \cos(w_0 t + \theta_0) w_0 v_0 + a\sin\theta_0 \\
& - \cos(\theta_0) v_0 w_0)\}/w_0^2. \\
\theta(t) = \quad & \theta_0 + w_0 t;
\end{aligned}
$$

D. Equations of an arc of circle with linear and angular accelerations

The velocity $v(t)$ depends on both the initial linear velocity v_0 and the linear acceleration $\dot{v}$. The orientation $\theta(t)$ is a function of the initial orientation θ_0, the initial angular velocity w_0 and the angular acceleration $\dot{w}$. Substituting $v(t)$ and $\theta(t)$ as a function of the initial kinematic and dynamic configuration v_0, θ_0, w_0 and the accelerations $\dot{v}$ and $\dot{w}$ yields the expressions:

$$
\begin{aligned}
x(t) = \quad & x_0 + \{\, A[(*4)(BD + CE) + (*3)(CD - BE)] \\
& + F[E(GH - IJ) + D(IH + GJ)] \\
& + K[(*3)(BE - CD) - (*4)(BD + CE)] \\
& + A[-(*2)(BD + CE) + (*1)(CD + BE)] - L \\
& + K[(*2)(BD + CE) + (*1)(CD + BE)]\}/M
\end{aligned}
$$

$$
\begin{aligned}
y(t) = \quad & y_0 + \{\, A[(*4)(BD + CE) + (*3)(BE - CD)] \\
& + F[E(IJ - GH) + D(IH + GJ)] \\
& + K[(*3)(CD - BE) - (*4)(BD + CE)] \\
& + A[-(*2)(BD + CE) + (*1)(CD - BE)] - L \\
& + K[(*2)(BD + CE) + (*1)(BE - CD)]\}/M \\
\theta(t) = \quad & \theta_0 + w_0 t + (\tfrac{\dot{w}t^2}{d})
\end{aligned}
$$

where (*1), (*2), (*3), (*4) are given by the Fresnel integrals as: $(*1) = FS(\frac{ta_r + w_0}{\sqrt{\pi}\sqrt{a_r}})$, $(*2) = FC(\frac{ta_r + w_0}{\sqrt{\pi}\sqrt{a_r}})$, $(*3) = FS(\frac{w_o}{\sqrt{\pi}\sqrt{ar}})$, $(*4) = FC(\frac{w_o}{\sqrt{\pi}\sqrt{ar}})$. $A = v_0\pi\dot{w}$, $B = cos(\frac{w_0^2}{2\dot{w}})$, $C = sin(\frac{w_0^2}{2\dot{w}})$, $D = cos\theta_0$, $E = sin\theta_0$, $F = \dot{v}\sqrt{\dot{w}^3}$, $G = cos(w_0 t)$, $H = \cos(\frac{\dot{w}t^2}{2})$, $I = sin(w_0 t)$, $J = sin(\frac{\dot{w}t^2}{2})$, $K = \dot{v}w_0\pi$, $L = \dot{v}sin\theta_0\sqrt{\dot{w}^3}$, $M = \sqrt{\dot{w}^5}$.

REFERENCES

[1] P. Fiorini and Z. Shiller. Robot motion planning in dynamic environments. In *International Symposium of Robotic Research*, pages 237–248, 1995.

[2] S. Fleury, P. Soueres, J.-P. Laumond, and R. Chatila. Primitive for smoothing mobile robot trajectories. *IEEE Transactions on Robotics and Automation*, 11(3), 1995.

[3] D. Fox, W. Burgard, and S. Thrun. The Dynamic Window Approach to Collision Avoidance. *IEEE Robotics and Automation Magazine*, 4(1), 1997.

[4] T. Fraichard. Trajectory planning in a dynamic workspace: a state-time-space approach. *Advanced Robotics*, 13(1), 1999.

[5] T. Fraichard and A. H. Inevitable collsion states- a step towards safer robots? *Advanced Robotics*, 18(10):1001–1024, 2004.

[6] K. Fujimura and H. Samet. A hierarchical strategy for path planning among moving obstacles. *IEEE Trans. on Robotics and Automation*, 5(1):61–69, 1989.

[7] R. Jacobs and Laumond. Non-holonomic motion planning for hilare-like mobile robots. In *Proceedings of the Intenational Symposium on Intelligent Robots*, 1991.

[8] James and James. *Mathematics Dictionary*. Van Nost. Reinhold, 1992.

[9] M. Kobilarov and G. Sukhatme. Near-optimal constrainned trajectory planning on outdoor terrain. In *Int. Conference on Robotics and Automation*, pages 1833–1840, 2005.

[10] F. Large, S. Sekhavat, Z. Shiller, and C. Laugier. Towards real-time global motion planning in a dynamic environment using the nlvo concept. In *Int. Conf. on Intelligent Robots and Systems*, 2002.

[11] J. Minguez and L. Montano. Nearness diagram (ND) navigation: Collision avoidance in troublesome scenarios. *IEEE Trans. on Robotics and Automation*, 20(1):45–59, 2004.

[12] E. Owen and L. Montano. Motion planning in dynamic environments usin the velocity space. In *IEEE-RSJ Int. Conf. on Intelligent Robots and Systems*, pages 997–1002, 2005.

[13] Z. S. P. Fiorini. Time optimal motion planning in dynamic environments. *International Journal of Applied Mathematics and Computer Science*, 7(4):101–126, 1997.

[14] S. Petti and T. Fraichard. Safe motion planning in dynamic environments. In *IEEE-RSJ Int. Conf. on Intelligent Robots and Systems*, pages 3726–3730, 2005.

[15] J. Reif and M. Sharir. Motion planning in the presence of moving obstacles. *J. ACM*, 41(4):764–790, 1994.

[16] M. Renaud and J.-Y. Fourquet. Minimum time motion of a mobile robot with two independent, acceleration-driven wheels. In *IEEE Int. Conference on Robotics and Automation*, pages 2608–2612, Alburquerque, New Mexico, 1997.

[17] A. Scheuer and T. Fraichard. Continuous-curvature path planning for car-like vehicles. In *IEEE-RSJ Int. Conf. on Intelligent Robots and Systems*, pages 997–1003, 1997.

[18] Z. Shiller, F. Large, and S. Sekhavat. Motion planning in dynamic environments: Obstacles moving along arbitrary trajectories. In *IEEE Int. Conf. on Robotics and Automation*, pages 3716–3721, Seoul, Korea, 2001.

[19] Z. Shiller, F. Large, S. Sekhavat, and C. Laugier. Motion planning in dynamic environment: Obstacles moving along artitrary trajectories. In *Workshop on Autonomous navigation in Dynamic Environments, Int. Conference on Robotics and Automation*, 2005.

[20] R. Simmons. The Curvature-Velocity Method for Local Obstacle Avoidance. In *IEEE Int. Conf. on Robotics and Automation*, pages 3375–3382, Minneapolis, USA, 1996.

[21] I. Ulrich and J. Borenstein. VFH*: Local Obstacle Avoidance with Look-Ahead Verification. In *IEEE Int. Conf. on Robotics and Automation*, pages 2505–2511, San Francisco, USA, 2000.