Motion Planning and Sensor-Guided Manœuvre Generation for an Autonomous Vehicle

C. Laugier, Ph. Garnier, Th. Fraichard, I. Paromtchik and A. Scheuer
Inria* Rhône-Alpes — Gravir†
ZIRST. 655 avenue de l'Europe
38330 Montbonnot Saint Martin
France

Abstract

This paper deals with a novel motion control approach for a car-like vehicle evolving in a dynamic and partially known environment. After having briefly presented the overall architecture of the control system and the related Global Trajectory Planner, we will put the emphasis onto the Manoeuver Execution module and its inter-connections with the off-line and the on-line motion planning functions. The key idea of the approach is to plan and carry out sensor-guided manœuvres: first, a nominal trajectory is generated using a reconstructed model of the world and a prediction of the most likely behaviors of the moving entities; then, the involved motion controls are generated and executed using a reactive scheme for taking into account the unforeseen events. This is done using local trajectories associated with some generic sensor-based manœuvres. Such trajectories are planned *on-line* and immediately executed by the system. Experimental results obtained with our automatic car-like vehicle are presented for two types of manœuvres: lane following/changing and autonomous parallel parking.

1 Introduction

Motion autonomy for various types of vehicles has already been widely studied. The state-of-the-art on this topic exhibits approaches of various complexity, integrating in different ways purely reactive approaches with more traditional hierarchical decisional schemes. A classical way to solve the motion autonomy problem for a car-like vehicle moving in a partially known environment, is to combine an *off-line global path/trajectory planner* (usually using the Dubins' curves [Dubins, 1957] or the Reed & Shepp curves [Reeds and Shepp, 1990]), with a *reactive execution controller* having the capability to track the generated nominal trajectory while avoiding collisions with unexpected obstacles. Unfortunately, this approach usually generates oscillatory movements and inconsistent behaviors (since it results from the combination of two contradictory functions: trajectory tracking and obstacle avoidance) [Garnier and Fraichard, 1996]. In order to be able to generate smooth and safe motions, while satisfying both the task constraints (i.e. the planned nominal trajectory) and the kinematic & dynamic constraints of the vehicle, we have introduced the concept of *behavior-based local trajectory*[1]. The basic idea is to combine appropriate local trajectories (generated on-line according to both the task constraints and the execution context) with the current nominal trajectory when an unforeseen event occurs.

In this paper, we present a novel architecture for controlling a car-like vehicle moving autonomously in a dynamic and partially known environment, while satisfying some strict kinematic & dynamic constraints. First, the overall architecture of the control system is briefly presented; this system includes a Global Trajectory Planner (*GTP*) and a Manœuvre Execution (*ME*) module (see &2). As it is explained in section 3, the *GTP* generates continuous-curvature trajectories[2] which satisfy the imposed kinematic constraints (including the non-holonomic constraints of the car-like vehicle). Afterwards, we put the emphasis onto the *ME* module (see &4) and its inter-connections with the off-line and the on-line motion planning functions. The key idea of the approach is to plan and carry out sensor-guided manœuvres: first, a nominal trajectory is generated (by the *GTP*) using a reconstructed model of the world and a

*Inst. Nat. de Recherche en Informatique et en Automatique.

†Lab. d'informatique GRAphique, VIsion et Robotique de Grenoble.

[1] The term "trajectory" represents both a geometric path (i.e. a smooth curve) and its associated velocity profiles.

[2] Such trajectories can easily be tracked by a car-like vehicle in a smooth way; this property doesn't hold for trajectories constructed using the Dubins' curves.

prediction of the most likely behaviors of the moving entities; then, the involved motion controls are executed (under the supervision of the *ME* module) using a reactive scheme for taking into account the unforeseen events. Since replanning cannot be done in real time, local trajectories associated with some generic sensor-based manœuvres are planned *on-line* and immediately executed by the *ME* module when an unforeseen event occurs. Two type of manœuvres are considered in the paper: lane following/changing, and autonomous parallel parking. Experimental results obtained with our automatic car-like vehicle are presented in section 5. This research work has been done within the scope of the French Praxitele programme aiming at developing a new urban transportation system based on a fleet of electric computer-driven vehicles [Parent and Daviet, 1996].

A kinematic model of a car-like vehicle is shown in Fig. 1. The vehicle's coordinates are denoted as a configuration $q = (x,\ y,\ \theta)^T$ relative to some reference coordinate system where $x = x(t)$ and $y = y(t)$ are the coordinates of the midpoint of the rear wheel axle, $\theta = \theta(t)$ is the orientation of the vehicle, and t is time. The motion of the vehicle is described by the equations

$$\begin{cases} \dot{x} = v\ \cos\phi\ \cos\theta, \\ \dot{y} = v\ \cos\phi\ \sin\theta, \\ \dot{\theta} = \frac{v}{L}\ \sin\phi, \end{cases} \qquad (1)$$

where $\phi = \phi(t)$ is the steering angle, $v = v(t)$ is the locomotion velocity of the midpoint of the front wheel axle, and L is the wheel base. The steering angle and locomotion velocity are two control commands $(\phi,\ v)$. Equations (1) correspond to a system with non-holonomic constraints because they involve the derivatives of the coordinates of the vehicle and are non-integrable [Latombe, 1991]. Equations (1) are valid for a vehicle moving on flat ground with a pure rolling contact without slippage between the wheels and the ground. This purely kinematic model of the vehicle is adequate to control low-speed motions, e.g. during parallel parking or lane following/changing in areas where only low-speed motions are allowed. For the high-speed motions, the dynamics of the vehicle must also be considered.

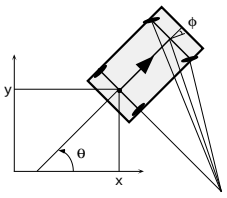

Figure 1: Kinematic model of a vehicle with front wheel steering.

In the sequel, we will call "automatic vehicle" a vehicle equipped with the following capabilities: (1) - a *sensor unit* to measure relative distances between the vehicle and environmental objects, (2) - a *servo unit* to control the steering angle and the locomotion velocity, (3) - a *control unit* that processes data from the sensor and servo units, and "drives" the vehicle by issuing appropriate servo commands.

2 The Overall Architecture

Our control architecture is shown in Fig. 2. Since the purpose of this paper is to show how sensor-guided manœuvres can be automatically generated and combined with some previously planned trajectories, we will focus on two main modules of this architecture: the Global Trajectory Planner (*GTP*) and the Manœuvre Execution (*ME*) module.

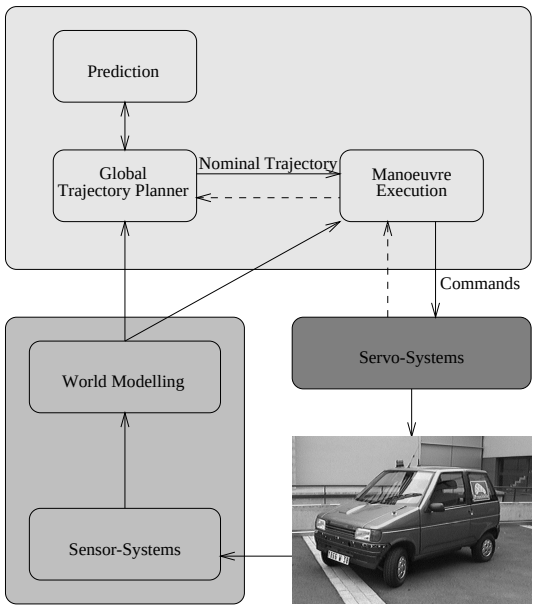

Figure 2: Overall architecture of the control system.

Global Trajectory Planner. The purpose of this module is to compute safe trajectories, i.e. time-ordered sequences of $(q, \dot{q})$, for executing the desired motions. The determination of such trajectories is done using: (1) an *a priori or a reconstructed model* of the vehicle environment, (2) the current *sensory data* (e.g. position and velocity of the moving obstacles), and (3) a *world prediction* which gives the most likely behaviors of the moving entities. The trajectory planning is done by taking into account both the non-collision constraints and the kinematic & dynamic constraints of the vehicle. It generates *nominal trajectories* which are tracked in a reactive way by the *ME* module.

Manœuvre Execution. The purpose of this module is to generate and to execute the required motion controls. It takes as input the nominal trajectory provided by the *GTP*, and tries to track this trajectory while reacting appropriately to unforeseen events. Depending on the context and on the type of motion task to execute, it selects the most appropriate generic manœuvre to carry out. This manœuvre is instantiated according to some known or sensed parameters (e.g. road curvature, available lateral and longitudinal displacements, velocity & acceleration, distance to an obstacle, etc.). Whenever the situation changes (e.g. intrusion of an unexpected obstacle), the *ME* module either updates the parameters of the current manœuvre, or it selects another manœuvre which is more appropriate to the new situation.

In order to execute the selected manœuvres, the *ME* module computes the required steering and velocity controls. The main characteristics of these controls are predefined, and depend on the type of manœuvre to execute (see for instance [Paromtchik and Laugier, 1996a] and [Paromtchik and Laugier, 1996b] for a parking manœuvre). These controls are computed on-line (using the involved sensor data), and applied to the vehicle's servo-systems.

A more detailed description of these modules is given in the next sections.

3 Global Trajectory Planner

3.1 Trajectory Planning

As mentioned earlier, the purpose of the Global Trajectory Planner is to compute a nominal trajectory between the current configuration of the vehicle and its goal. The nominal trajectory is a time-ordered sequence of states $(q, \dot{q})$ which respects different types of constraints:

- *Kinematic constraints:* a car-like vehicle has a limited steering range that restricts the geometric shape of its motion.
- *Dynamic constraints:* these constraints due to engine power, ground-wheel interaction, etc., restrict the accelerations and velocities that can be applied to the vehicle.
- *Collision avoidance constraints:* collision with the stationary and moving obstacles of the environment are forbidden.

Given that a trajectory can be represented also by a geometric path and a velocity profile along this path, the GTP addresses the problem at hand in two complementary steps:

1. *Path planning:* a geometric path leading the vehicle to its goal is computed. The left-hand side of Fig. 3 depicts an example of such a path: it is a continuous curve whose curvature is upper-bounded (so as to respect the kinematic constraints of a car-like vehicle). Besides it is collision-free with the stationary obstacles of the environment.
2. *Velocity planning:* the velocity profile of the vehicle along its path is computed; this profile respects the dynamic constraints of the vehicle and yields no collision between the vehicle and the moving obstacles of the environment. Velocity planning requires the knowledge of the future behavior of the moving obstacles; this information is provided by a prediction module. The right-hand side of Fig. 3 illustrates the velocity planning: it depicts a space-time diagram (the horizontal axis being the position along the path and the vertical one the time dimension). The curve represents the motion of the vehicle through time whereas the thick black lines are the traces left by moving obstacles when they cross the path of the vehicle.

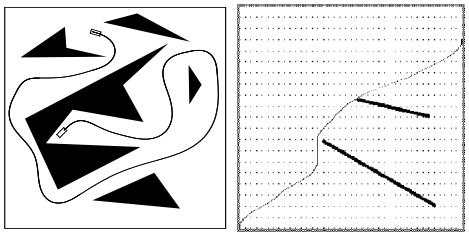

Figure 3: (a) path planning and (b) velocity planning.

In the next section, we will focus on path planning step. The reader is referred to [Fraichard and Laugier, 1993] and [Fraichard and Scheuer, 1994] for more details about velocity planning.

3.2 Path Planning

A car-like vehicle is subject to two non-holonomic constraints: it can only move along a direction perpendicular to its rear wheel axle (continuous tangent direction), and its turning radius is lower bounded (maximum curvature) [Barraquand and Latombe, 1989]. Numerous works, e.g. [Barraquand and Latombe, 1989; Mirtich, 1992; Laumond *et al.*, 1994; Švestka and Overmars, 1995a], have been done to plan paths for such vehicles, but almost all of them generate sequence of Dubins' curves [Dubins, 1957], i.e. paths made of circular arcs connected by tangential line segments. The main reason for this is that these paths are the shortest ones for such a vehicle [Dubins, 1957]. The main drawback of these paths is that their curvature is not continuous. Accordingly, a vehicle following such a path has to stop at each curvature discontinuity in order to reorient its front wheels.

Since we are mostly interested in planning forward paths only, i.e. paths without manœuvres, we do not

want the vehicle to stop, except possibly at the initial and final configurations. For this reason, we add a continuous-curvature constraint to the classical nonholonomic path planning problem for car-like vehicles. In addition, we introduce a constraint on the curvature derivative; it is upper bounded so as to reflect the fact that the vehicle can only reorient its front wheels with a finite velocity.

Addressing a similar problem (but without the maximum curvature constraint), Boissonnat et al. [Boissonnat *et al.*, 1994] proved, using the Pontryagin's Maximum Principle, that the shortest path between two vehicle's configurations is made up of line segments and clothoid[3] arcs of maximum curvature derivative. Later, Kostov and Degtiariova-Kostova proved that these shortest paths are, in the general case, made of an infinity of pieces [Kostov and Degtiariova-Kostova, 1995].

Similar results can be extended to the particular problem we consider, adding circular arcs of maximum curvature to the set of locally optimal paths. Therefore, in order to come up with a practical solution to the problem at hand, we define a set of paths, derived from Dubins' curves, that have continuous curvature and maximum curvature derivative. These paths contain at most eight pieces, each piece being either a line segment, a circular arc of maximum curvature, or a clothoid arc. They are called *SCC-paths* (for Simple Continuous Curvature paths). They are used to design a *local path planner*, i.e. a non-complete collision-free path planner, which in turn is embedded in a global path planning scheme, namely the Probabilistic Path Planner [Švestka and Overmars, 1995b]. The result is the first path planner for a car-like vehicle that generates collision-free paths with continuous curvature and maximum curvature derivative.

The reader is referred to [Scheuer and Fraichard, 1997] for a complete presentation of this continuous curvature path planner. Some experimental results are shown in Fig. 4.

4 Manœuvre Execution

4.1 Lane Following/Changing

Autonomous lane following is performed by tracking a nominal trajectory produced by the *GTP*. Whenever an unforeseen obstacle obstructs the way of the vehicle, the nominal trajectory is locally modified in real time, in order to avoid the collision. This local modification of the trajectory is done in order to satisfy a set of different constraints: collision avoidance, time constraints, kinematic & dynamic constraints. In a previous approach, we have used a fuzzy behavior merging process for combining a trajectory tracking behavior with a collision avoidance behavior; but, such an approach generates oscillations, and it does not guaranty that all the previous constraints will be always satisfied [Garnier and Fraichard, 1996]. In the current approach, the *ME* generates *smooth local trajectories* for avoiding the detected obstacles. These local trajectories allow the vehicle to move away from the obstructed nominal trajectory, and to catch up this nominal trajectory when the (stationary or moving) obstacle has been overtaken. All these local trajectories verifies the previous constraints, and allow the vehicle to always move along a smooth path.

[3] A clothoid is a curve whose curvature is a linear function of its arc length.

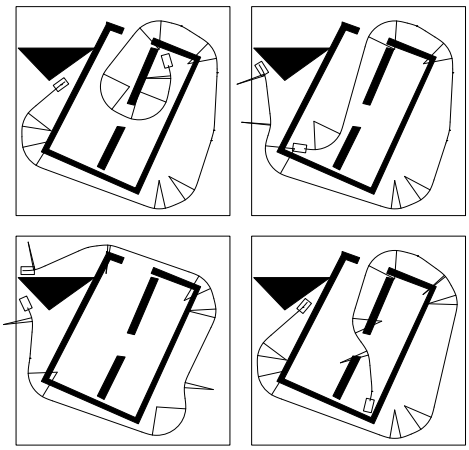

Figure 4: Some safe and smooth paths generated by the planner.

Lane Following

Any trajectory tracking method working for nonholonomic mobiles, can be used for implementing the lane following behavior. We have chosen to apply the Kanayama approach [Kanayama *et al.*, 1991], which guarantees the stable tracking of a feasible trajectory when the vehicle's control commands are of the following form:

$$\dot{\theta} = \dot{\theta}_{ref} + v_{R,ref}(k_y y_e + k_\theta \sin\theta_e), \tag{2}$$

$$v_R = v_{R,ref} \cos\theta_e + k_x x_e, \tag{3}$$

where $q_e = (x_e, y_e, \theta_e)^T$ represents the error between the reference configuration q_{ref} and the current configuration q of the vehicle ($q_e = q_{ref} - q$), $\dot{\theta}_{ref}$ and $v_{R,ref}$ are the reference velocities, $v_R = v\cos\phi$ is the control command for the locomotion velocity of the midpoint of the rear wheel axle, k_x, k_y, k_θ are positive constants, and $\phi = \arctan\left(\frac{\dot{\theta} L}{v_{R,ref}}\right)$.

Lane Changing

The lane changing behavior may be executed for executing a lane change, for avoiding a stationary obstacle, or for overtaking another vehicle. Lane changing is carried out by generating and tracking a smooth local trajectory. Let $\mathcal{T}$ be the nominal trajectory to track. The obstacle

avoidance or overtaking manœuvres roughly consist in smoothly moving towards a collision-free "parallel" trajectory, and in catching up $\mathcal{T}$ as soon as it becomes possible (see Fig. 5). The applied algorithm is the following:

1. Generate a smooth local trajectory τ_1 which connects $\mathcal{T}$ with a collision-free local trajectory τ_2 "parallel" to $\mathcal{T}$ (τ_2 is obtained by translating appropriately the involved piece of $\mathcal{T}$).
2. Track τ_1 and τ_2 until the obstacle has been overtaken.
3. Generate a smooth local trajectory τ_3 which connects τ_2 with $\mathcal{T}$, and track τ_3.

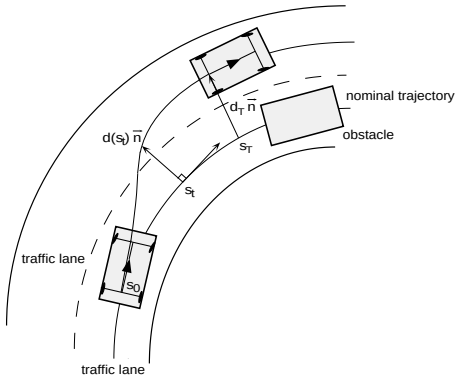

Figure 5: Generation of smooth local trajectories for avoiding an obstacle.

A feasible smooth trajectory for lane changing can be obtained using a quintic polynomial:

$$d(s) = d_T \left(10 \left(\frac{s}{s_T}\right)^3 - 15 \left(\frac{s}{s_T}\right)^4 + 6 \left(\frac{s}{s_T}\right)^5 \right), \quad (4)$$

where d_T is the distance between the two traffic lanes, s_T is the curviligne distance along $\mathcal{T}$ which is necessary to complete the lane change, and $s = s_t$ is the curviligne abscissa along $\mathcal{T}$ since the starting point of the lane change [Nelson, 1989]. The distance d_T is supposed to be known beforehand (it can also be computed according to the size of the obstacle). The minimal value of s_T is estimated as:

$$s_{T,min} = \frac{\pi \sqrt{k\, d_T}}{2\, \mathcal{C}_{max}}, \quad (5)$$

where $\mathcal{C}_{max}$ stands for the maximum allowed curvature:

$$\mathcal{C}_{max} = \mathbf{min} \left\{ \frac{\tan(\phi_{max})}{L}, \frac{\gamma_{max}}{v_{R,ref}^2} \right\}, \quad (6)$$

γ_{max} is the maximum allowed lateral acceleration, and $k > 1$ is an empirical constant (e.g. $k = 1.17$ in our experiments).

When an obstacle is detected, a value $s_{T,min}$ is calculated according to (5) and compared with the distance separating the vehicle from the obstacle. The result of this computation is used to decide which behavior to apply: performing a lane change, or slowing down (and possibly stopping) the vehicle. As for the lane changing manœuvre, the related local trajectory is tracked as follows: at each time t from the starting time T_0, the reference position p_{ref} is translated along the vector $d(s_i).\vec{n}$, where $\vec{n}$ represents the unit normal vector to the nominal velocity vector along $\mathcal{T}$; the reference orientation θ_{ref} is converted into $\theta_{ref} + \arctan\left(\frac{\partial d}{\partial s}(s_i)\right)$, and the reference velocity $v_{R,ref}$ is obtained using the following equation:

$$v_{R,ref}(t) = \frac{dist(p_{ref}(t), p_{ref}(t + \Delta t))}{\Delta t}, \quad (7)$$

where $dist$ stands for the euclidean distance.

4.2 Parallel Parking

Autonomous parallel parking involves three main phases: localizing a sufficient space (parking bay), obtaining an appropriate starting location for the vehicle relatively to the parking bay, and performing the parallel parking manœuvre. During the first phase, the vehicle moves slowly along the traffic lane, and it uses its range sensors for constructing a local map of the environment and for detecting obstacles. The map is used for selecting an appropriate free space for parking the vehicle.

Drivers know from experience that before the parking manœuvre starts, the vehicle must be oriented near parallel to the parking bay and it must also reach a convenient start position in front of the bay. A start location for parallel parking is shown in Fig. 6 where an automatic vehicle A1 is in a traffic lane. The parking lane with parked vehicles B1, B2 and a parking bay between them is on the right-hand side of the vehicle A1. L1 and L2 are respectively the length and width of A1, and D1 and D2 are the distances available for longitudinal and lateral displacements of A1 within the bay. D3 and D4 are the longitudinal and lateral displacements of the corner A13 of A1 relative to the corner B24 of B2. The distances D1, D2, D3 and D4 are computed by the control unit from data obtained by the sensor and servo units. The control unit compares the length (D1-D3) and width (D2-D4) of the parking bay with the length L1 and width L2 of A1, where L1 and L2 include sufficient clearance for the vehicle to move around. If (D1-D3) > L1 and (D2-D4) > L2, the parking bay is sufficient for parallel parking.

During parallel parking, iterative low-speed backwards-and-forwards motions with coordinated control of the steering angle and locomotion velocity

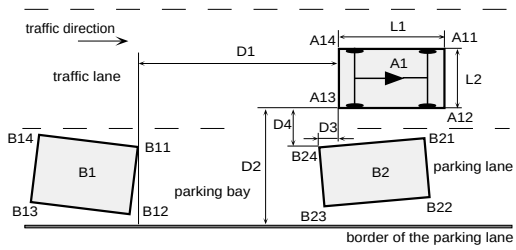

Figure 6: Start location for parallel parking.

are performed to produce a lateral displacement of the vehicle into the parking bay. The number of such motions depends on the distances D1, D2, D3, D4 and the necessary parking "depth" which depends on the width L2 of the vehicle A1. The start and end orientations of the vehicle are the same for each iterative motion $i = 1, \ldots, N$.

For the i-th iterative motion (but omitting the index "i"), let the start coordinates of the vehicle be $x_0 = x(0)$, $y_0 = y(0)$, $\theta_0 = \theta(0)$ and the end coordinates be $x_T = x(T)$, $y_T = y(T)$, $\theta_T = \theta(T)$, where T is duration of the motion. The "parallel parking" condition means that

$$\theta_0 - \delta_\theta \; < \; \theta_T \; < \; \theta_0 + \delta_\theta, \tag{8}$$

where $\delta_\theta > 0$ is a small admissible error in orientation of the vehicle.

The following control commands of the steering angle ϕ and locomotion velocity v provide the parallel parking manœuvre [Paromtchik and Laugier, 1996b]:

$$\phi(t) = \phi_{max}\, k_\phi\, A(t), \quad 0 \le t \le T, \tag{9}$$

$$v(t) = v_{max}\, k_v\, B(t), \quad 0 \le t \le T, \tag{10}$$

where $\phi_{max} > 0$ and $v_{max} > 0$ are the admissible magnitudes of the steering angle and locomotion velocity respectively, $k_\phi = \pm 1$ corresponds to a right side (+1) or left side (–1) parking bay relative to the traffic lane, $k_v = \pm 1$ corresponds to forward (+1) or backward (–1) motion,

$$A(t) = \begin{cases} 1, & 0 \le t < t', \\ \cos \frac{\pi(t-t')}{T^*}, & t' \le t \le T - t', \\ -1, & T - t' < t \le T, \end{cases} \tag{11}$$

$$B(t) = 0.5 \left(1 - \cos \frac{4\pi t}{T}\right), \quad 0 \le t \le T, \tag{12}$$

where $t' = \frac{T - T^*}{2}$, $T^* < T$.

The commands (9) and (10) are open-loop in the (x, y, θ)-coordinates. The steering wheel servo-system and locomotion servo-system must execute the commands (9) and (10), in order to provide the desired (x, y)-path and orientation θ of the vehicle. The resulting accuracy of the motion in the (x, y, θ)-coordinates depends on the accuracy of these servo-systems. Possible errors are compensated by subsequent iterative motions.

For each pair of successive motions $(i, i+1)$, the coefficient k_v in (10) has to satisfy the equation $k_{v,i+1} = -k_{v,i}$ that alternates between forward and backward directions. Between successive motions, when the velocity is null, the steering wheels turn to the opposite side in order to obtain a suitable steering angle ϕ_{max} or $-\phi_{max}$ to start the next iterative motion.

In this way, the form of the commands (9) and (10) is defined by (11) and (12) respectively. In order to evaluate (9)-(12) for the parallel parking manœuvre, the durations T^* and T, the magnitudes ϕ_{max} and v_{max} must be known.

The value of T^* is lower-bounded by the kinematic and dynamic constraints of the steering wheel servo-system. When the control command (9) is applied, the lower bound of T^* is

$$T^*_{min} = \pi\, \mathbf{max} \left\{ \frac{\phi_{max}}{\dot{\phi}_{max}}, \sqrt{\frac{\phi_{max}}{\ddot{\phi}_{max}}} \right\}, \tag{13}$$

where $\dot{\phi}_{max}$ and $\ddot{\phi}_{max}$ are the maximal admissible steering rate and acceleration respectively for the steering wheel servo-system. The value of T^*_{min} gives duration of the full turn of the steering wheels from $-\phi_{max}$ to ϕ_{max} or vice versa, i.e. one can choose $T^* = T^*_{min}$.

The value of T is lower-bounded by the constraints on the velocity v_{max} and acceleration $\dot{v}_{max}$ and by the condition $T^* < T$. When the control command (10) is applied, the lower bound of T is

$$T_{min} = \mathbf{max} \left\{ \frac{2\pi\, v'(\mathrm{D1})}{\dot{v}_{max}}, T^* \right\}, \tag{14}$$

where the empirically-obtained function $v'(\mathrm{D1}) \le v_{max}$ serves to provide a smooth motion of the vehicle when the available distance D1 is small.

The computation of T and ϕ_{max} aims to obtain the maximal values such that the following "longitudinal" and "lateral" conditions are still satisfied:

$$|\, (x_T - x_0)\, \cos\theta_0 + (y_T - y_0)\, \sin\theta_0 \,| \; < \; \mathrm{D1}, \tag{15}$$

$$|\, (x_0 - x_T)\, \sin\theta_0 + (y_T - y_0)\, \cos\theta_0 \,| \; < \; \mathrm{D2}. \tag{16}$$

Using the maximal values of T and ϕ_{max} assures that the longitudinal and, especially, lateral displacement of the vehicle is maximal within the available free parking space. The computation is carried out on the basis of the model (1) when the commands (9) and (10) are applied. In this computation, the value of v_{max} must correspond to a safety requirement for parking manœuvres (e.g. $v_{max} = 0.75\ m/s$ was found empirically).

At each iteration i the parallel parking algorithm is summarized as follows:

1. Obtain available longitudinal and lateral displacements D1 and D2 respectively by processing the sensor data.
2. Search for maximal values T and ϕ_{max} by evaluating the model (1) with controls (9), (10) so that conditions (15), (16) are still satisfied.
3. Steer the vehicle by controls (9), (10) while processing the range data for collision avoidance.
4. Obtain the vehicle's location relative to environmental objects at the parking bay. If the "parked" location is reached, stop; else, go to step 1.

When the vehicle A1 moves backwards into the parking bay from the start location shown in Fig. 6, the corner A12 (front right corner of the vehicle) must not collide with the corner B24 (front left corner of the bay). The start location must ensure that the subsequent motions will be collision-free with objects limiting the bay. To obtain a convenient start location, the vehicle has to stop at a distance D3 that will ensure a desired minimal safety distance D5 between the vehicle and the nearest corner of the bay during the subsequent backward motion. The relation between the distances D1, D2, D3, D4 and D5 is described by a function

$$\mathcal{F}(\mathrm{D1}, \mathrm{D2}, \mathrm{D3}, \mathrm{D4}, \mathrm{D5}) = 0. \qquad (17)$$

This function can not be expressed in closed form, but it can be estimated for a given type of vehicle by using the model (1) when the commands (9) and (10) are applied. The computations are carried out off-line and stored in a look-up table which is used on-line, to obtain an estimate of D3 corresponding to a desired minimal safety distance D5 for given D1, D2 and D4 [Paromtchik and Laugier, 1996a].

When the necessary parking "depth" has been reached, some clearance between the vehicle and the parked ones is provided, i.e. the vehicle moves forwards or backwards so as to be in the middle of the parking bay between the two parked vehicles.

5 Implementation and experiments

The approach described in the paper has been implemented and tested on our experimental automatic vehicle (a modified LIGIER electric car). This vehicle can either be manually driven as a classical car, or it can move autonomously using a *control unit* based on a Motorola VME162-CPU board and a transputer net. The *sensor unit* of the vehicle makes use of a belt of ultrasonic range sensors (Polaroid 9000) and of a linear CCD-camera. The steering wheel servo-system is equipped with a direct current motor and an optical encoder to measure the steering angle. The locomotion servo-system of the vehicle is equipped with a 12 kW asynchronous motor and two optical encoders located onto the rear wheels (for odometry data). The vehicle also has an hydraulic braking servo-system. The developed steering and velocity control is implemented using the ORCCAD software [Simon *et al.*, 1993] running on a SUN workstation. The compiled code is transmitted via Ethernet to the VME162-CPU board.

An example of our experimental setup for lane following/changing on a circular road is shown in Fig. 7. In this experiment, the LIGIER vehicle follows a nominal trajectory along the curved traffic lane, and it finds on its way another vehicle moving at a lower velocity (see Fig. 7a). When the moving obstacle is detected, a local trajectory for a right lane change is generated by the system, and LIGIER performs the lane changing manœuvre, as illustrated in Fig.7b. Afterwards, LIGIER moves along a trajectory parallel to its nominal trajectory, and a left lane change is performed as soon as the obstacle has been overtaken (see Fig. 7c. Finally, LIGIER continues to follow its nominal trajectory, as illustrated in Fig. 7d.

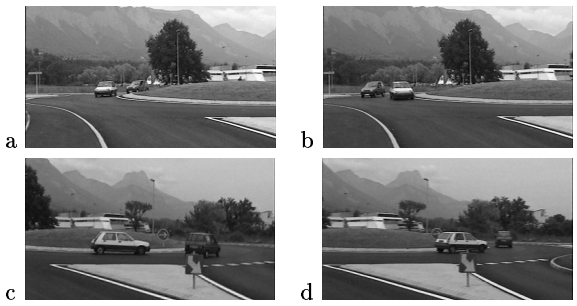

Figure 7: Sequence of motions for lane following/changing on a circular road: (a) following the nominal trajectory,(b) lane changing (to the right) and overtaking, (c) lane changing to the left, (d) continuing with the nominal trajectory.

The steering and velocity controls applied during the lane following/changing manœuvres are shown in Fig. 8; the related motion of the vehicle is depicted in Fig. 9. It can be noticed on this example, that the velocity of the vehicle has increased when moving along the local "parallel" trajectory (see Fig. 9); this is due to the fact that the vehicle has to satisfy the time constraints associated to its nominal trajectory.

An example of our experimental setup for autonomous parallel parking in a street is shown in Fig. 10. This type of manœuvre can be carried out in an environment including moving obstacles, e.g. a pedestrian or some other vehicles. In this experiment, LIGIER was manually

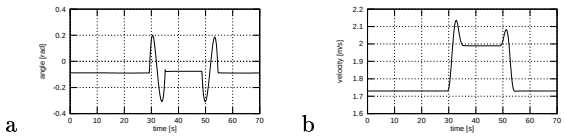

Figure 8: Controls applied during a lane change: (a) steering angle, (b) locomotion velocity.

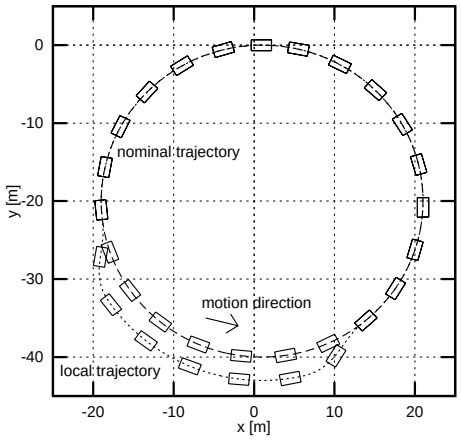

Figure 9: Lane following/changing on a circular road.

driven to a position near the parking bay, before being left by the driver (after having switch on the automatic mode). Then, LIGIER moved forward autonomously in order to localize the parking bay, obtained a convenient start location, and performed a parallel parking manœuvre. As it is shown in Fig. 10a, LIGIER reacts appropriately if a pedestrian crosses the street in a dangerous proximity to the vehicle. Fig. 10b shows the starting position which has been selected for executing the parking manœuvre, and Fig. 10c shows the execution of the first backwards motion. It can be noticed in Fig. 10d, that the vehicle located in front of LIGIER has moved backwards during the execution of the parking manœuvre; this led to a reduction of the length of the parking bay, and consequently to a modification (in real-time) of the backward and forward motions executed by LIGIER.

The control commands (9) and (10) generated for executing a parallel parking manœuvre are shown in Fig. 11. The corresponding motion of the vehicle is depicted in Fig. 12. The distances available relatively to the start location was D1=4.9 m, D2=2.7 m; the lateral distance D4=0.6 m was measured by the sensor unit, and the longitudinal distance D3=0.8 m was estimated so as to ensure the minimal safety distance D5=0.2 m. In this experiment, five iterative motions was performed to park the vehicle. As seen in Fig. 11 and Fig. 12, the durations T of the iterative motions, magnitudes of the steering angle ϕ_{max} and locomotion velocity v_{max} correspond to the available displacements D1 and D2 within the parking bay (e.g. the values of T, ϕ_{max} and v_{max} differ for the first and last iterative motion).

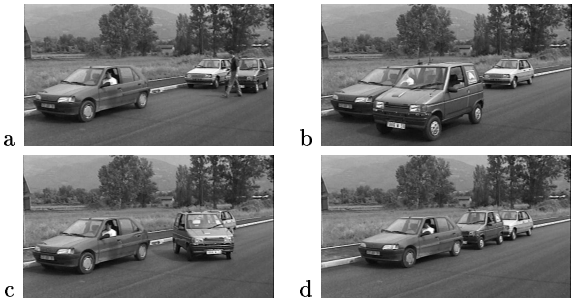

Figure 10: Sequence of motions for parallel parking: (a) autonomous motion to localize a parking bay; (b) selecting an appropriate start location; (c) backward motion into the bay; (d) the parallel parking is completed.

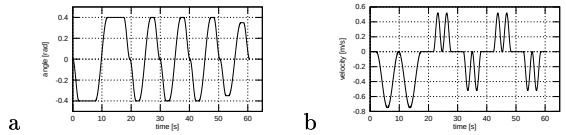

Figure 11: Controls generated for the backward and forward motions involved in the parallel parking manœuvre: (a) steering angle; (b) locomotion velocity.

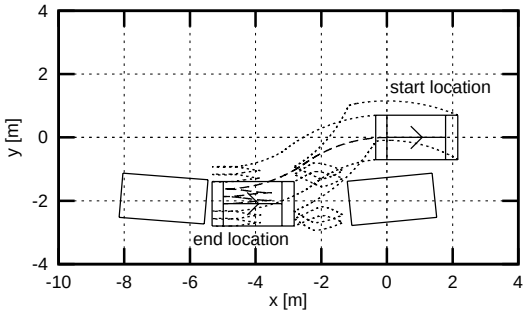

Figure 12: Parallel parking when backward and forward motions are performed.

6 Conclusion

We have presented a novel architecture for controlling a car-like vehicle moving autonomously in a dynamic and partially known environment, while satisfying some

strict kinematic & dynamic constraints. This architecture, which includes a Global Trajectory Planner (*GTP*) and a Manœuvre Execution module (*ME*), allows us to plan and to carry out sensor-guided manœuvres: first, a nominal trajectory is generated (by the *GTP*) using a reconstructed model of the world and a prediction of the most likely behaviors of the moving entities; then, the involved motion controls are executed (under the supervision of the *ME* module) using a reactive scheme for taking into account the unforeseen events. We have shown how the *ME* module has been connected to some off-line and on-line motion planning capabilities, in order to obtain both the required reactivity and to satisfy the task constraints. We have also shown two types of generic manœuvres (lane following/changing, and autonomous parallel parking), how appropriate smooth local trajectories could be generated in real-time for obtaining the required behaviors. Various experiments performed with our automatic car-like vehicle have been described; these experiments have exhibit consistent behaviors.

Acknowledgment

This work was partially supported by the Inria-Inrets[4] Praxitèle programme on urban public transport *[1994-1997]*, and the Inco-Copernicus ERBIC15CT960702 project "Multi-agent robot systems for industrial applications in the transport domain" *[1997-1999]*. The authors would like to thank the members of the Praxitele and Sharp teams for their support during this work.

References

[Barraquand and Latombe, 1989] J. Barraquand and J.-C. Latombe. On non-holonomic mobile robots and optimal maneuvering. *Revue d'Intelligence Artificielle*, 3(2):77–103, 1989.

[Boissonnat *et al.*, 1994] J.-D. Boissonnat, A. Cerezo, and J. Leblond. A note on shortest paths in the plane subject to a constraint on the derivative of the curvature. Research Report 2160, Inst. Nat. de Recherche en Informatique et en Automatique, January 1994.

[Dubins, 1957] L. E. 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*, 79:497–516, 1957.

[Fraichard and Laugier, 1993] Th. Fraichard and C. Laugier. Dynamic trajectory planning, path-velocity decomposition and adjacent paths. In *Proc. of the Int. Joint Conf. on Artificial Intelligence*, volume 2, pages 1592–1597, Chambéry (FR), September 1993.

[4]Inst. Nat. de Recherche sur les Transports et leur Sécurité.

[Fraichard and Scheuer, 1994] Th. Fraichard and A. Scheuer. Car-like robots and moving obstacles. In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, volume 1, pages 64–69, San Diego (CA), May 1994.

[Garnier and Fraichard, 1996] Ph. Garnier and Th. Fraichard. A fuzzy motion controller for a car-like vehicle. In *Proc. of the IEEE-RSJ Int. Conf. on Intelligent Robots and Systems*, volume 3, pages 1171–1178, Osaka (JP), November 1996.

[Kanayama *et al.*, 1991] Y. Kanayama, Y. Kimura, F. Miyazaki, and T. Noguchi. A Stable Tracking Control Method for a Non-Holonomic Mobile Robot. *IEEE/RSJ International Workshop on Intelligent Robots and Systems IROS'91, Osaka, Japan*, pages 1236–1241, November 1991.

[Kostov and Degtiariova-Kostova, 1995] V. Kostov and E. Degtiariova-Kostova. Some properties of clothoids. Research Report 2752, Inst. Nat. de Recherche en Informatique et en Automatique, December 1995.

[Latombe, 1991] J. C. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, 1991.

[Laumond *et al.*, 1994] J.-P. Laumond, P. E. Jacobs, M. Taïx, and R. M. Murray. A motion planner for non-holonomic mobile robots. *IEEE Trans. Robotics and Automation*, 10(5):577–593, October 1994.

[Mirtich, 1992] B. Mirtich. Using skeletons for nonholonomic motion planning. Research Report ESRC 92-16/RAMP 92-6, Departement of Electrical Engineering and Computer Science, University of California, Berkeley, June 1992.

[Nelson, 1989] W. L. Nelson. Continuous curvature paths for autonomous vehicles. In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, volume 3, pages 1260–1264, Scottsdale (AZ), May 1989.

[Parent and Daviet, 1996] M. Parent and P. Daviet. Automated urban vehicles: towards a dual mode prt (personal rapid transit). In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pages 3129–3134, Minneapolis (MN), April 1996.

[Paromtchik and Laugier, 1996a] I. E. Paromtchik and C. Laugier. Autonomous parallel parking of a nonholonomic vehicle. In *Proc. of the IEEE Int. Symp. on Intelligent Vehicles*, pages 13–18, Tokyo (JP), September 1996.

[Paromtchik and Laugier, 1996b] I. E. Paromtchik and C. Laugier. Motion generation and control for parking an autonomous vehicle. In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pages 3117–3122, Minneapolis (MN), April 1996.

[Reeds and Shepp, 1990] J. A. Reeds and L. A. Shepp. Optimal paths for a car that goes both forwards and backwards. *Pacific Journal of Mathematics*, 145(2):367–393, 1990.

[Scheuer and Fraichard, 1997] A. Scheuer and Th. Fraichard. Continuous-curvature path planning for car-like vehicles. In *Proc. of the IEEE-RSJ Int. Conf. on Intelligent Robots and Systems*, volume 2, pages 997–1003, Grenoble (FR), September 1997.

[Simon *et al.*, 1993] D. Simon, B. Espiau, E. Castillo, and K. Kapellos. Computer-Aided Design of a Generic Robot Controller Handling Reactivity and Real-Time Control Issues. In *IEEE Transactions on Control Systems Technology*, pages 213–229, December 1993.

[Švestka and Overmars, 1995a] P. Švestka and M. H. Overmars. Coordinated motion planning for multiple car-like robots using probabilistic roadmaps. In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, volume 2, pages 1631–1636, Nagoya (JP), May 1995.

[Švestka and Overmars, 1995b] P. Švestka and M. H. Overmars. Probabilistic path planning. Technical Report UU-CS-1995-22, Utrecht University, P.O.Box 80.089, 3508 TB Utrecht, the Netherlands, May 1995.