An autonomous car-like robot navigating safely among pedestrians

— Source link ↗

Cédric Pradalier, J. Hermosillo, Carla Koike, Christophe Braillon ...+2 more authors

Institutions: French Institute for Research in Computer Science and Automation

Published on: 27 Sep 2004 - International Conference on Robotics and Automation

Topics: Obstacle avoidance, Motion planning and Robot kinematics

Related papers:

- Motion planning and obstacle avoidance for mobile robots in highly cluttered dynamic environments
- Integrating Perception and Planning for Autonomous Navigation of Urban Vehicles
- Real-time motion planning and safe navigation in dynamic multi-robot environments
- Mobility planning for autonomous navigation of multiple robots in unstructured environments
- Model predictive collaborative motion planning and control of mobile robots including safety aspects

An Autonomous Car-Like Robot Navigating Safely Among Pedestrians

Cédric Pradalier, Jorje Hermosillo, Carla Koike, Christophe Braillon, Pierre Bessiere, Christian Laugier

An Autonomous Car-Like Robot Navigating Safely Among Pedestrians

C. Pradalier, J. Hermosillo, C.Koike, C.Braillon, P.Bessière, C.Laugier

GRAVIR – INRIA – INPG Grenoble

INRIA Rhône-Alpes, 38334 Saint Ismier cedex France

Abstract— The recent development of a new kind of public transportation system relies on a particular double-steering kinematic structure enhancing maneuverability in cluttered environments such as downtown areas. We call *bi-steerable car* a vehicle showing this kind of kinematics. Endowed with autonomy capacities, the bi-steerable car ought to combine suitably and safely a set of abilities: simultaneous localisation and environment modelling, motion planning and motion execution amidst moderately dynamic obstacles. In this paper we address the integration of these four essential autonomy abilities into a single application. Specifically, we aim at reactive execution of planned motion. We address the fusion of controls issued from the control law and the obstacle avoidance module using probabilistic techniques.

Index Terms— Car-like robot, navigation, path planning, obstacle avoidance, autonomy.

I. INTRODUCTION

The development of new Intelligent Transportation Systems (ITS), more practical, safe and accounting for environmental concerns, is a technological issue of highly urbanised societies today [12]. One of the long run objectives is to reduce the use of the private automobile in downtown areas, by offering new modern and convenient public transportation systems. Examples of these, are the CyCab robot – designed at INRIA and currently traded by the Robosoft company (see www.robosoft.fr) – and the pi-Car prototype of IEF (Institut d'Electronique Fondamentale, Université Paris-Sud).

The kinematic structure of these robots differs from that of a car-like vehicle in that it allows the steering of both the front axle and the rear one. We call a vehicle showing this feature a bi-steerable car (or -BiS-car for short).

Endowed with autonomy capacities, the bi-steerable car ought to combine suitably and safely a set of abilities that eventually could come to the relief of the end-user in complex tasks (e.g. parking the vehicle). Part of these abilities have been tackled separately in previous work: simultaneous localisation and environment modelling[14], motion planning execution amidst static obstacles[8], [7] and obstacle avoidance in a moderately dynamic environment without accounting for a planned motion[9].

In this paper we address the integration of these four essential autonomy abilities into a single application.

Specifically, we aim at reactive execution of planned motion. We address the fusion of controls issued from the control law and the obstacle avoidance module using probabilistic techniques. We are convinced that these results represent a step further towards the motion autonomy of this kind of transportation system. The structure of the paper follows.

In section 2, we sketch the environment reconstruction and localisation methods we used and we recall how the central issue regarding the motion planning and execution problem for the general BiS-car was solved. Section 3 explains how our obstacle avoidance system was designed and section 4 how it was adapted to the trajectory tracking system. In section 5 we present experimental settings showing the fusion of these essential autonomy capacities in our bi-steerable platform the CyCab robot. We close the paper with some concluding remarks and guidelines on future work in section 6.

II. LOCALISATION, ENVIRONMENT MODELLING, MOTION PLANNING AND EXECUTION

In the design of an autonomous car-like robot, we are convinced that localisation, modelling of the environment, path planning and trajectory tracking are of fundamental importance.

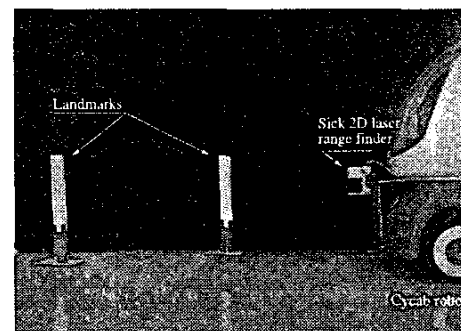### A. Map-building and Localisation

Fig. 1. CyCab robot and landmarks

A localisation system has been implemented on the CyCab (see [14] for details). Fig. 1 shows our robot, its sensor and the landmarks: cylinders covered with reflector sheets, specially designed for our Sick laser
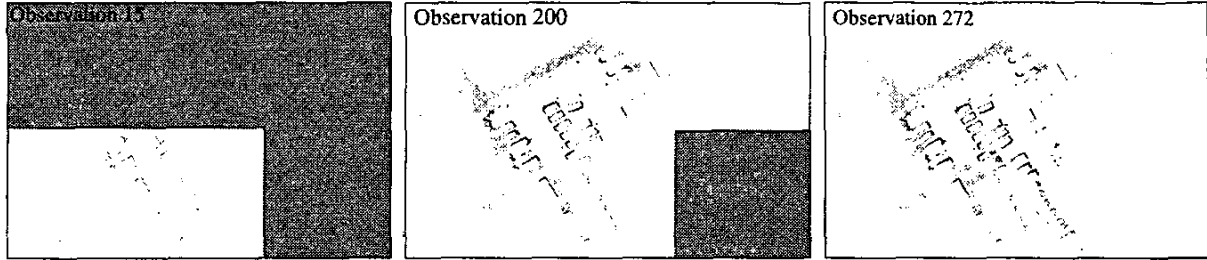
Fig. 2. Obstacle map evolution: Experimental map of the car-park area built during the obstacle map-building phase.

range finder. This system provides us with not only a estimate of robot configuration (position and orientation) but also, using efficient SLAM[1] techniques, with a a map of the environment. Actually, this is a two-faces map: on the first hand, a map of landmarks observed in the robot environment, used for accurate and robust localisation, And on the second hand, a kind of simplified occupancy grid ([3]).

Fig. 2 shows how this occupancy map evolves while we are exploring the environment. Once the map-building phase finished, the occupancy map is converted into a pixmap and passed to the Motion Planning stage.

B. Motion Planning Amidst Static Obstacles

The Motion Planner adopted for the CyCab was presented in [16]. Essentially, it is a two step approach, dealing separately with the physical constraints (the obstacles) and with the kinematic constraints (the non-holonomy). The planner first builds a collision-free path without taking into account the non-holonomic constraints of the system. Then, this path is approximated by a sequence of collision-free feasible sub-paths computed by a suitable[2] steering method. Finally, the resulting path is smoothed.

A key issue in non-holonomic motion planning is to find a steering method accounting for the kinematics of the robot. One way of designing steering methods for a non-holonomic system is to use its flatness property [5] allowing also for feedback linearisation of the nonlinear system (this is discussed in section II-C). This is what we did for the general BiS-car for which a flat output—or linearising output—was given in [16].

The striking advantage of planning a path in the flat space is that we only need to parameterise a 2-dimensional curve whose points and derivatives define everywhere the current 4-dimensional state (x, y, θ, ϕ). Fig. 3 shows the outcome of the motion planner using an obstacle map generated as described in the previous section.

[1]Simultaneous Localisation And Mapping
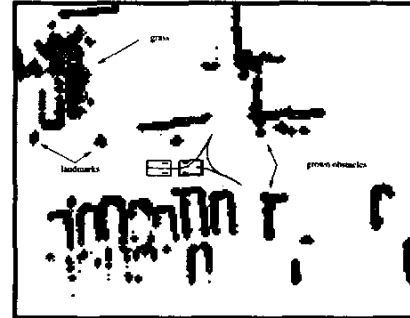[2]i.e. Verifying the topological property as explained in [16].

Fig. 3. Path computed by the motion planner using a real obstacle map (pixmap). The obstacles are grown as well as the robot before computing the path.

C. Trajectory tracking using flatness

It is well known that a non-holonomic system cannot be stabilised using only smooth state static feedbacks [1]. Ever since then, time-varying feedbacks [15] and dynamic feedbacks have been successfully used in particular for the canonical tractor-trailer and car-like robots [4].

Flat systems are feedback linearisable by means of a restricted class of dynamic feedback called endogenous [5]. The interest is that we are able to use state-of-the-art linear control techniques to stabilise the system. We present here results coming from recent work on feedback linearisation of the general BiS-car.

As we are interested in stabilising the BiS-car around a reference trajectory, we explored the fact that, owing to the flatness property, the system is diffeomorphic to a linear controllable one [5]. The endogenous dynamic feedback that linearises the general bi-steerable system is presented in [8]. Then, from linear control theory, it can be shown that the closed-loop control stabilising the reference trajectory y^* has the following form :

$$y_i^{(3)} = (y_i^*)^{(3)} - \sum_{j=0}^{2} k_{i,j} \left(y_i^{(j)} - (y_i^*)^{(j)} \right) \quad i = 1, 2 \quad (1)$$

where $(.)^{(p)}$ stands for the total derivative of order p. See [2] for details.

III. Obstacle Avoidance Using Probabilistic Reasoning

The previous approach considers trajectories in a static environment. In order to make the execution of these trajectories more robust, an obstacle avoidance system should be prepared to react to unpredicted changes in the environment. One solution to this problem is to implement a trajectory deformation mechanism (see [13], [10]). This solution is very efficient, but complex and computationally heavy. As our computational resources are quite limited, we would prefer some light reactive method that would achieve security even if it does not give always an optimal solution. This section will present the principles of our obstacle avoidance module. The originality of this module is not its performance, but rather its expression as a Bayesian inference problem.

A. Specification

The CyCab can be commanded through a speed V and a steering angle Φ. It is equipped with π radians sweeping laser range finder. As the full sensor output is too big to be managed efficiently with our computational power, we summarised it as 8 values : the distances to the nearest obstacle in a $\pi/8$ angular sector(see Fig. 4). We will call $D_k, k = 1\ldots8$ the probabilistic variables corresponding to these measures.

Besides, we will assume that this robot is commanded by some high-level system (trajectory following for instance) which provides him with a pair of desired commands (V_d, Φ_d).
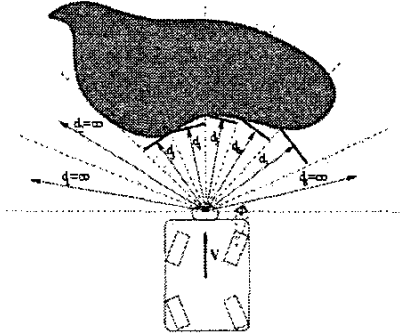
Fig. 4. Obstacle avoidance: situation

Our goal is to find commands to apply to the robot, guarantying the vehicle security while following the desired command as much as possible.

B. Sub-models definition

In angular sector i, we define a probability distribution over (V, Φ) knowing the desired commands and the distance D_i measured in this sector:

$$P_i(V\Phi \mid V_d\Phi_dD_i) = P_i(V \mid V_dD_i)P_i(\Phi \mid \Phi_dD_i) \quad (2)$$

where $P_i(V \mid V_dD_i)$ and $P_i(\Phi \mid \Phi_dD_i)$ are Gaussian distributions respectively centred on $\mu_V(V_d, D_i)$
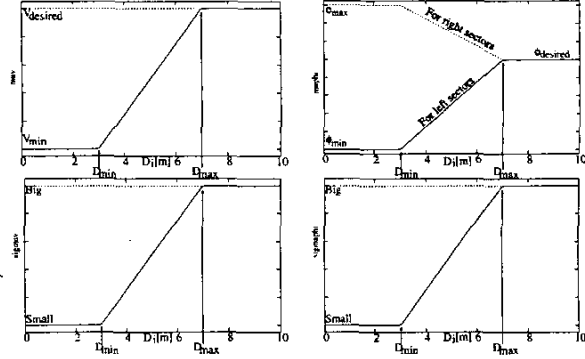
Fig. 5. Evolution of mean and standard deviation of $P_i(V \mid V_dD_i)$ and $P_i(\Phi \mid \Phi_dD_i)$ according to distance measured ($V_{min} = 0, \Phi \in [-0.4, 0.4]$
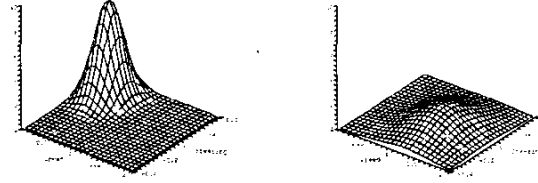
Fig. 6. Shape of $P_i(V\Phi \mid V_d\Phi_dD_i)$ for close and far obstacles

and $\mu_\Phi(\Phi_d, D_i)$ with standard deviation $\sigma_V(V_d, D_i)$ and $\sigma_\Phi(\Phi_d, D_i)$. Functions $\mu_V, \mu_\Phi, \sigma_V, \sigma_\Phi$ are defined with sigmoid shape as illustrated in Fig. 5. Example of resulting distributions are shown in Fig. 6.

There is two specific aspects to notice in Fig. 5 and 6. First, concerning the means μ_V and μ_Φ, we can see that, the farther the obstacle, the closer to the desired command μ will be, and conversely, the nearer the obstacle, the more secure μ: minimal speed, strong steering angle.

Second, the standard deviation can be seen as a constraint level. For instance, when an obstacle is very close to the robot (small D_i), its speed must be strongly constrained to zero, this is expressed by a small standard deviation. Conversely, when obstacle is far, robot speed can follow the desired command, but there is no damage risk in not applying exactly this command. This low level constraint is the result of a big standard deviation.

C. Command fusion

Knowing desired controls and distance to the nearest obstacle in its sector, each sub-model, defined by $P_i(V\Phi \mid V_d\Phi_dD_i)$, provides us with a probability distribution over the robot controls. As we have eight sectors, we will have to fuse the controls from eight sub-models. Then we will find the best control in term of security and desired control following.

To this end, we define the following joint distribution:

$$P(V \; \Phi \; V_d \; \Phi_d \; D_1 \ldots D_8 \; S) = \quad (3)$$

$$P(D_1 \ldots D_8)P(V_d \; \Phi_d)$$

$$P(S)P(V \; \Phi \; | \; V_d \; \Phi_d \; D_1 \ldots D_8 \; S)$$

where variable $S \in [1 \ldots 8]$ express which sector is considered. $P(D_1 \ldots D_8)$ and $P(V_d\Phi_d)$ are unknown distribution[3]. As there is no need to favour a specific sub-model, we define $P(S)$ as a uniform distribution. The semantic of S will be emphasised by the definition of $P(V\Phi \; | \; V_d\Phi_dD_1 \ldots D_8S)$:

$$P(V\Phi \; | \; V_d\Phi_dD_1 \ldots D_8[S = i]) = P_i(V\Phi \; | \; V_d\Phi_dD_i)$$

$P(V\Phi \; | \; V_d\Phi_dD_1 \ldots D_8S)$ express commands probability distribution should sector i be the only one considered.

Using equation 3, we can now express the distribution we are really interested in:

$$P(V \; \Phi \; | \; V_d \; \Phi_d \; D_1 \ldots D_8) = \quad (4)$$

$$\sum_S (P(S)P(V \; \Phi \; | \; V_d \; \Phi_d \; D_1 \; \ldots \; D_8 \; S))$$

This equation is actually the place where the different constraint level expressed by functions σ_V and σ_Φ will be useful. The more security constraints there will be, the more peaked will be the sub-model control distribution. So sub-models who see no obstacles in their sector will contribute to the sum with quasi-flat distribution, and those who see perilous obstacles will add a peaky distribution, hence having more influence (see Fig. 7). Finally the command really executed by the robot is the one which maximise $P(V \; \Phi \; | \; V_d \; \Phi_d \; D_1 \ldots D_8)$ (eq. 4).
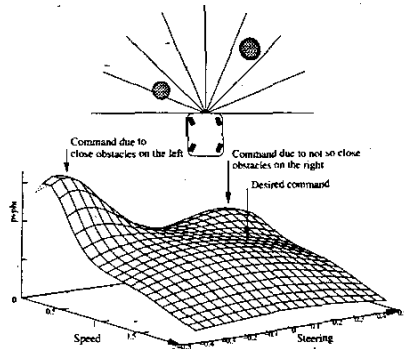

Fig. 7. Probability distribution over speed and steering, resulting from the obstacle avoidance system.

D. Results

Fig. 8 illustrates the result of the obstacle avoidance system applied on a simulated example. The simulated CyCab is driven manually with a joystick in a square environment. In this specific situation, the driver is continuously asking for maximum speed, straight forward

---
[3]Actually, as we know we will not need them in future computation, we don't have to specify them.

(null steering angle). We can observe on the dotted trajectory that obstacle avoidance module curves the trajectory in order to avoid the walls. From the density of dots, we can figure out the robot speed: it breaks when it comes close to the walls and while its turning and try to follow desired speed when obstacles are not so threatening.

Note that when going straight toward an obstacle, we obtain a bi-modal distribution expressing that system should steer either right or left. We implemented the optimisation process in order to favour turning right.
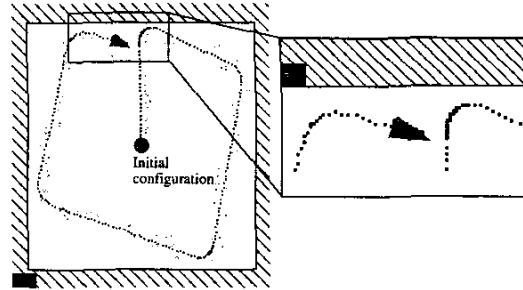

Fig. 8. Robot trajectory while driven manually with constant desired steering angle

E. Discussion

The method presented in this section provides us with an efficient way to fuse a security system and orders from a high level system. Nevertheless the perturbations introduced in the trajectory following system by obstacle avoidance are such that they can make it become unstable. Next section will show how we integrate trajectory tracking and obstacle avoidance.

IV. TRAJECTORY TRACKING WITH OBSTACLE AVOIDANCE

While executing a trajectory, obstacle avoidance will modify certain commands in order to follow as much as possible desired orders while granting security. These modifications may introduce delay and diversions in the control loop. If no appropriate action is taken to manage these delays the control law may generate extremely strong accelerations or even become unstable when obstacles are gone. This is typically the case when our system evolves among moving pedestrians. Thus we designed a specific behaviour to adapt smoothly our control system to the perturbations induced by obstacle avoidance.

A. Multiplexed trajectory tracking

a) Validity domain of flat control law: Experimentally, we found that the control law based on flatness can manage errors in a range of about 1 meter and 15 degrees around nominal trajectory. Furthermore, as this control law controls the third derivative of the flat output (eq.

1), it is a massively integrating system. For this reason, a constant perturbation such as immobilisation due to a pedestrian standing in front of the vehicle will result in a quadratic increase of the control law output. This phenomena is mainly due to the fact that when obstacle avoidance slows the robot down, it breaks the dynamic rules around which the flat control law was built. So, there is no surprise in its failure.

b) Trajectory tracking behaviour: In order to deal with the situations that flat control law cannot manage, we designed a simplistic trajectory tracking behaviour (*TTB*) based again on probabilistic reasoning[4]. As this behaviour has many similarities with a weighted sum of proportional control laws, we do not expect it to be sufficient to stabilise the robot on its trajectory. Nevertheless, it is sufficient to bring it back in the convergence domain of the flat control law when obstacle avoidance perturbations have occurred. Basically, the resulting behaviour is as follows: while the robot is close to its nominal position, it is commanded by flat control law. When, due to obstacle avoidance, it is too far from its nominal position, TTB takes control, and try to bring it back to flat control law's convergence domain. When it enters this domain, flat control law is reinitialised and starts accurate trajectory tracking.

c) Time control: Path resulting from path planning (section II-B) is a list of robot configuration indexed by time. So when the robot is slowed down by a traversing pedestrian, it compensates its delay by accelerating. Nevertheless, when the robot is stopped during a longer time, let's say fifteen seconds, it should not consider to be delayed of fifteen seconds, otherwise it will try to reach a position fifteen second ahead, without tracking the intermediary trajectory. To tackle this difficulty, we introduced a third mode to the trajectory tracking: when the robot is really too far from its nominal position, we freeze the nominal position, and we use the TTB to reenter the domain where nominal position can be unfrozen.

The global system is illustrated by Fig. 9: we implemented some kind of multiplexer/demultiplexer which manage transitions between control laws. In order to avoid oscillating between control laws when at the interface between two domains of validity, we had to introduce some hysteresis mechanism in the switching.
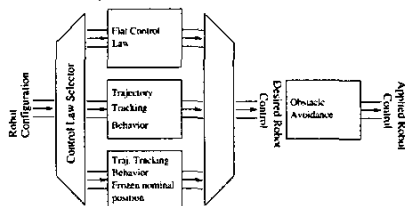

Fig. 9.   Basic diagram of the control law selector mechanism

[4]Due to limited space, it will not be described in this paper

Fig. 10 shows the collaboration of obstacle avoidance and trajectory following on a simulated example. Planned trajectory passes through an obstacle which was not present at map building time. Obstacle avoidance modifies controls in order to grant security. When errors with respect to nominal trajectory is too big, our control law selector switch to the trajectory tracking behaviour. Here it is a big longitudinal error, due to obstacle avoidance slowing down the vehicle, which trigger the switching.
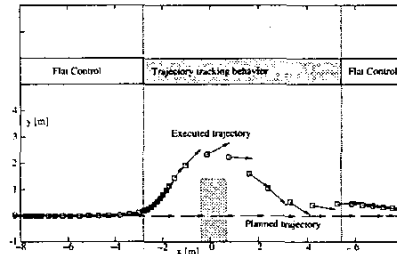

Fig. 10.   Collaboration of trajectory tracking and obstacle avoidance on a simulated example

B. Discussion

Using the multiplexed control laws we managed to integrate, in the same control loop, our flat control, accurate but sensible to perturbation, with our trajectory tracking behaviour, inaccurate but robust to perturbation. By this way we obtained a system capable of tracking trajectory generated by our path planner while accounting for dynamic object in the environment.

Note that obstacle avoidance, as implemented here, has many similarities with a potential field obstacle avoidance system. So, it should not be expected to be safe in any dynamic environment[5]. Nevertheless, we found it safe in a moderately dynamic environment, such as our car park with maneuvering cars and moving pedestrians.

Finally, when the robot has gone too far from reference trajectory, or when reactive obstacle avoidance can not find suitable controls anymore, it may be necessary to re-plan a new trajectory to the goal. This has not been implemented on the robot yet, but it is quite sure that it will not make neither technical nor scientific problem arise.

V. EXPERIMENTAL SETUP

We tested the integration of these essential autonomy capacities in our experimental platform the CyCab robot. The aim was to validate the theoretical considerations made for the BiS-car and to get insight into the limitations of the whole motion scheme.

The computation power on-board the CyCab is a *Pentium II*^{TM} 233MHz running a RedHat^{TM} Linux system. All programs were written in C/C++ language.

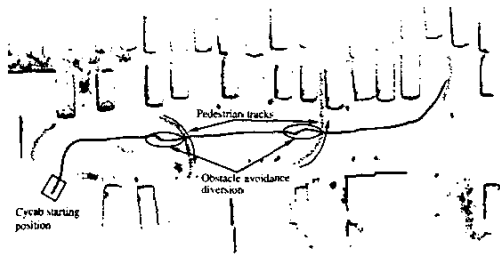[5]Interested reader may refer to [11] for solutions to these problems.

Fig. 11. Executed trajectory among static obstacles and moving pedestrians. Rear middle point trajectory is drawn.
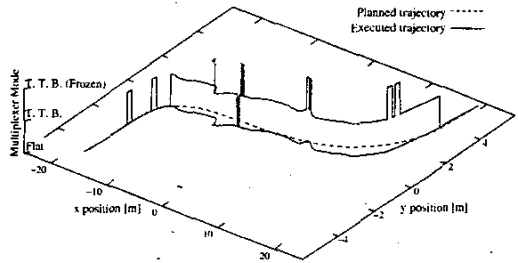

Fig. 12. Executed trajectory with respect to planned trajectory, and multiplexer mode.

During the experiments the speed of the robot was limited to $1.5ms^{-1}$. The control rate of the robot was fixed at $50ms$. The throughput rate of the laser range-finder is limited to $140ms$[6]; therefore the control system relies momentarily in odometry[6] readings.

Figs 11 and 12 illustrates how a planned trajectory is executed while avoiding moving pedestrians. In this environment, the control law using flatness could only be used at the beginning and at the end of the trajectory. On the remaining of the trajectory, speed and steering angle are adjusted in order to maintain security while keeping pace with the plan as much as possible.

VI. DISCUSSION & CONCLUSIONS

In this paper, we presented our new steps toward the autonomy of a bi-steerable car. The integration of local-isation, map building, trajectory planning and execution in a moderately dynamic environment was discussed. Control law using the CyCab flatness property was found to be insufficient for trajectory tracking among moving pedestrians.

Even if this integration was successful and provides satisfactory results, we are convinced that a reactive be-haviour cannot be sufficient for the autonomy of vehicle in a real urban environment. For this reason, we are working on the perception and identification of road users (pedestrians, cars, bikes or trucks). By this way, we will be able to predict future movement of "obstacles" and

---

6This rate is fair enough for our needs, even though we could use a real-time driver.

---

to react accordingly, in a *smarter* way than the simple scheme proposed in this paper.

VII. REFERENCES

[1] R.W. Brockett. Asymptotic stability and feedback stabilization. In R.W. Brockett, R.S. Millman, and H.J. Sussmann, editors, *Differential Geometric Control Theory*, pages 181–191, Boston, MA: Birkhäuser, 1983.

[2] Carlos Canudas de Wit, Bruno Siciliano, and George Bastin Eds. *Theory of Robot Control*. Springer-Verlag, 1996.

[3] A. Elfes. Using occupancy grids for mobile robot perception and navigation. *IEEE Computer, Special Issue on Autonomous Intelligent Machines*, June 1989.

[4] M. Fliess, J. Lévine, P. Martin, and P. Rouchon. Design of trajectory stabilizing feedback for driftless flat systems. In *Proc. of the European Control Conference*, pages 1882–1887, Rome, Italy, september 1995.

[5] M. Fliess, J. Lévine, P. Martin, and P. Rouchon. Flatness and defects of nonlinear systems: introductory theory and examples. *Int. Journal of Control*, 61(6):1327–1361, 1995.

[6] J. Hermosillo, C. Pradalier, and S. Sekhavat. Modelling odometry and uncertainty propagation for a bi-steerable car. In *Proc. of the IEEE Intelligent Vehicle Symp.*, Versailles (FR), June 2002. Poster session.

[7] J. Hermosillo, C. Pradalier, S. Sekhavat, and C. Laugier. Experi-mental issues from map building to trajectory execution for a bi-steerable car. In *Proc. of the IEEE Int. Conf. on Advanced Robotics*, Coimbra (PT), July 2003.

[8] J. Hermosillo and S. Sekhavat. Feedback control of a bi-steerable car using flatness; application to trajectory tracking. In *Proc. of the American Control Conference*, Denver, CO (US), June 2003.

[9] C. Koike, C. Pradalier, P. Bessière, and E. Mazer. Proscriptive bayesian programming application for collision avoidance. In *Proc. of the IEEE-RSJ Int. Conf. on Intelligent Robots and Systems*, Las vegas, NV (US), October 2003.

[10] F. Lamiraux and D. Bonnafous. Reactive trajectory deformation for non- holonomic systems: Application to mobile robots. In *Proc. IEEE Int. Conf. on Robotics and Automation*, 2002.

[11] F. Large, S. Sekhavat, Z. Shiller, and C. Laugier. Towards real-time global motion planning in a dynamic environment using the NLVO concept. In *Proc. of the IEEE-RSJ Int. Conf. on Intelligent Robots and Systems*, Lausanne (CH), September-October 2002.

[12] Ch. Laugier, S. Sekhavat, L. Large, J. Hermosillo, and Z. Shiller. Some steps towards autonomous cars. In *Proc. of the IFAC Symp. on Intelligent Autonomous Vehicles*, pages 10–18, Sapporo (JP), September 2001.

[13] O. Lefebvre, F. Lamiraux, and C. Pradalier. Obstacles avoidance for car-like robots. integration and experimentation on two robots. In *Proc. IEEE Int. Conf. on Robotics and Automation*, 2004.

[14] C. Pradalier and S. Sekhavat. Concurrent matching, localization and map building using invariant features. In *Proc. of the IEEE-RSJ Int. Conf. on Intelligent Robots and Systems*, Lausanne (CH), September-October 2002.

[15] C. Samson and K. Ait-Abderrahim. Feedback stabilization of a nonholonomic wheeled mobile robot. In *Proc. of the IEEE-RSJ Int. Conf. on Intelligent Robots and Systems*, pages 1242–1246, Osaka (JP), November 1991.

[16] S. Sekhavat, J. Hermosillo, and P. Rouchon. Motion planning for a bi-steerable car. In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pages 3294–3299, Seoul (KR), May 2001.