Obstacles Avoidance for Car-Like Robots
Integration And Experimentation on Two Robots

Olivier Lefebvre[*] Florent Lamiraux[*] Cédric Pradalier[†]
[*]firstname.lastname@laas.fr

LAAS-CNRS
7, av du Colonel Roche
31077 Toulouse Cedex 4
FRANCE

[†]cedric.pradalier@inrialpes.fr
GRAVIR - INRIA - INPG
INRIA Rhône-Alpes
600, av. de l'Europe
38334 Saint Ismier Cedex
FRANCE

***Abstract*— In this paper we address the problem of obstacles avoidance for car-like robots. We present a generic non-holonomic path deformation method that has been applied on two robots. The principle is to perturb the inputs of the system in order to move away from obstacles and to keep the non-holonomic constraints satisfied. We present an extension of the method to car-like robots. We have integrated the method on two robots (Dala and CyCab) and carried out experiments that show the portability and genericity of the approach.**

Fig. 1. Robots Dala (on left) and CyCab

Introduction

Computing a collision-free trajectory for a non-holonomic mobile robot located in a map is a difficult task. It deals with two extensively studied topics : geometric path planning and non-holonomic motion. Nevertheless, if we want to apply results of research carried out in robotics to automatic motion of road vehicles for instance, some problems are still to be solved. Cars that would park automatically or that would be able to manage a stop-and-go mode in traffic jam, are potential industrial applications of non-holonomic motion planning. In this context, collision avoidance at the time of the execution of the trajectory is a prerequisite. A previously planned trajectory may have to be deformed during execution to avoid collisions. There are several reasons for that :

- the map of the environment can be inaccurate,
- new obstacles may appear that were not in the map,
- if the planned trajectory is not exactly followed due to a poor localization of the robot, unexpected collisions may occur.

Numerous approaches have been proposed to overcome these difficulties. [1] proposed a method to deform online the path followed by the robot in order to get away from obstacles detected along the motion. This approach has been extended to the case of a unicycle-like mobile robot by [2] and then to the case of a holonomic mobile manipulator by [3]. In both papers, the geometry of the robot is approximated by a set of balls and no or only one very simple non-holonomic constraint is treated.

The non-holonomic path deformation method [4] is a generic approach of the on-line trajectory deformation issue. It enables to deform a trajectory at execution time so that it moves away from obstacles and that the non-holonomic constraints of the system keep satisfied. It can be applied to any non-holonomic system, and was initially elaborated on robot Hilare 2 towing a trailer [5] and used in the context of path optimization for trucks carrying huge airplanes components on narrow roads.

This paper presents the integration of the method on two non-holonomic systems. Our contribution is the extension of the method in order to respect the curvature bound of a car-like robot during the deformation process (see sec. II). We also address in section III the issue of the algorithm integration on different systems with different architectures. Eventually, we present some convincing experimental results in section IV, that illustrate the genericity of the method and its portability.

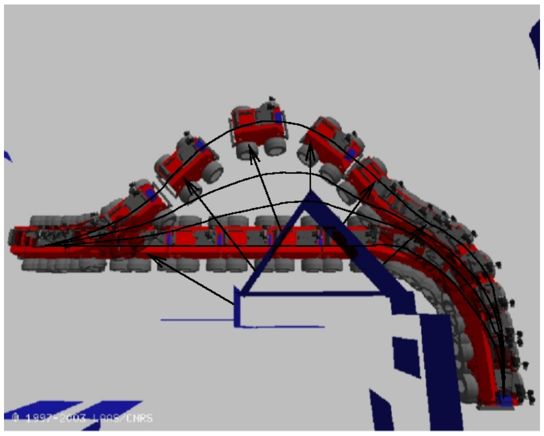

Fig. 2. Direction of deformation created by unexpected obstacles

I. The Non-holonomic Path Deformation Method

The path deformation method we present is a generic path optimization method upon a certain criterion. In the context of real-time obstacle avoidance, this criterion is a potential function that increases when the path gets closer to obstacles. The principle of the method is the following. **Given a feasible path** for a system, possibly in collision, the path is iteratively deformed in such a way that it moves away from obstacles.

We first need to compute a **potential function along the path**, that increases when the distance of the path to the obstacles decreases. Then a **direction of deformation** is computed in order to make this potential decrease, that is for the path to get away from obstacles. The seminal idea of the method is then to establish a **relationship between the direction of deformation and the inputs of the system** represented by the mobile robot. We compute input perturbations that will make the path go in the direction of deformation. If we want indeed the deformed path to be feasible by the system, not any path deformation that would make the potential decrease is admissible.

We present here the principle of the method. We refer interested reader to [4] for further details.

A. Nonholonomic systems

A drift-less non-holonomic system of dimension n is characterized by a set of $k < n$ vector fields $X_1(\mathbf{q})$,...,$X_k(\mathbf{q})$, where $\mathbf{q} \in \mathcal{C} = \mathbf{R}^n$ is the configuration of the system. For each configuration $\mathbf{q}$, the admissible velocities of the system are the linear combinations of the $X_i(\mathbf{q})$'s. Equivalently, a path $\mathbf{q}(s)$ defined over an interval $[0, S]$ is a feasible path if and only if:

$$\forall s \in [0, S] \qquad \mathbf{q}'(s) = \sum_{i=1}^{k} u_i(s) X_i(\mathbf{q}) \tag{1}$$

where $\mathbf{q}'(s)$ is the derivative of $\mathbf{q}(s)$.

B. Infinitesimal Path Deformation

A path is a mapping from an interval $[0, S]$ into the configuration space $\mathcal{C} = \mathbf{R}^n$ of the robot. A path is thus completely defined by the value of the inputs of system (1) on interval $[0, S]$: the $u_i(s)$ with $1 \leq i \leq k$.

To deform a given path we only need to perturb the input functions $u_1(s)$, ..., $u_k(s)$ of the initial path $\mathbf{q}(s)$. For that, we define n real functions $v_1(s)$,...,$v_n(s)$ called *input perturbations* and a real number τ that gives the deformation amplitude. Then a deformed path can be represented by a function of two variables: $\mathbf{q}(s, \tau)$. The inputs of the system are in this case $u_i(s) + \tau v_i(s)$.

Because we only perturb the entries of the system, we are sure that the non-holonomic constraints remain satisfied after the deformation.

Figure 2 represents the direction of deformation $\eta(s)$ created by some obstacles. It corresponds to the infinitesimal path deformation due to infinitesimal inputs perturbations : $\eta(s) = \frac{\partial \mathbf{q}}{\partial \tau}(s, 0)$.

Therefore, we have established a relation between the input perturbations and the direction of deformation.

C. A Relationship between Obstacles and Path Deformation

We now need to establish a relation between the obstacles and the direction of deformation. Given a set of obstacles detected while following the current path, we define a potential field $U(\mathbf{q})$ in the configuration space in such a way that the value of the potential increases when the robot gets closer to obstacles.

From the potential field in the configuration space, we define the potential of the current path by summing $U(\mathbf{q})$ along the path:

$$V = \int_0^S U(\mathbf{q}(s)) ds$$

To make the path go away from obstacles, we choose the function $\mathbf{v}(s) = (v_1(s) ... v_n(s))$ that minimizes the first-order variation of the path potential (*i. e.* the first-order derivative of the path potential when the first-order variation of the path is $\eta(s)$):

$$\frac{\partial V}{\partial \tau}(0) = \int_0^S \frac{\partial U}{\partial \mathbf{q}}(\mathbf{q}(s))^T \eta(s) ds \tag{2}$$

We emphasize the fact that only the gradient of the obstacle potential field is of interest to us. The way it is computed is the subject of section II-A.

This closes the loop: from a given obstacle potential field, we compute a direction of deformation that minimizes the gradient of the potential. Then the direction of deformation $\eta(s)$ provides us with the input perturbations to be applied to the dynamical system 1 so that the path gets away from obstacles.

D. Take into account the boundary conditions

The path optimization method we present leave free the choice of boundary conditions. That is in our case the inital and goal configurations of the deformed path. In fact we want the initial and goal configurations of the trajectory not to be changed by the deformation process. Therefore, we impose the following boundary conditions to the deformation process: $\forall \tau \in [0, +\infty), \mathbf{q}(0, \tau) = \mathbf{q}(0, 0)$ and $\mathbf{q}(S, \tau) = \mathbf{q}(S, 0)$. The computation of the projection of the controls of system (1) over the subspace characterized by these boundary conditions is presented in [5].

II. Extensions of the Non-Holonomic Path Deformation Method

We present in this section some extensions of the method. In a first part we give some details about the obstacle potential field computation. Then in a second part we present the issue of the bounded steering angle of a non-holonomic system.

The deformation process can indeed increase the curvature of a path and there is no guarantee that it will not exceed its bound. We present a method to keep the path curvature after the deformation, below a given bound. The main idea is to consider the curvature bound as an obstacle.

A. Obstacle Potential Field Computation

In order for the deformation method to calculate a direction of deformation, we must compute a continuous potential field function that increases when the robot gets closer to obstacles. We present here how to effectively compute this potential field.

Given a perception of the environment and a trajectory, we can compute a set of obstacle points whose distances to any point of the workspace are known. Let M be a point in the workspace W. We note $\nu_i(M)$ the potential generated by an obstacle point P_i located at distance d from M. Let $R(\mathbf{q})$ be a point of the robot, it can be the closest point to obstacle P_i for instance.
Then $\nu_i(R(\mathbf{q}))$ stands for the potential generated by obstacle P_i when robot is at configuration $\mathbf{q}$.

We notice that there exists a mapping from $(\mathcal{C}xW)$ into $\mathbf{R}$:
$(\mathbf{q}, P_i) \rightarrow d$, so that the potential ν_i can be expressed as a function of distance d.

The obstacle potential field at a configuration $\mathbf{q}$ is the sum of the potentials relative to each obstacle :

$$U(\mathbf{q}) = \sum_i \nu_i(R(\mathbf{q})) \tag{3}$$

The gradient of the potential field in the configuration space is obtained by differentiating equation 3:

$$\frac{\partial U}{\partial \mathbf{q}}(\mathbf{q}) \quad = \quad \sum_i \nabla \nu_i(R(\mathbf{q})) \frac{\partial R}{\partial \mathbf{q}}(\mathbf{q})$$

If we remember eq. (2), we are only interested in the gradient of the potential in the configuration space. We see from previous expression that $\nabla \nu_i(R(\mathbf{q}))$ can be any vector of the workspace. For genericity reasons, we make this vector derive from a potential, so that it is equivalent to a force: $\mathbf{f}_i(R(\mathbf{q})) = -\nabla \nu_i(R(\mathbf{q}))$.

We want this "force" to be null when obstacles are far from the robot ($d > d_1$) and to increase when the distance d to obstacles gets close to zero. A function which verifies these assumptions for obstacle P_i is for instance such that:

$$\begin{array}{lll} \|\mathbf{f}_i(R(\mathbf{q}))\| = \frac{1}{(d+d_0)^2} - \frac{1}{(d_1+d_0)^2} & \text{if} & 0 \leq d \leq d_1 \\ \|\mathbf{f}_i(R(\mathbf{q}))\| = 0 & \text{if} & d > d_1 \end{array} \tag{4}$$

And the expression of the potential from which the function derives is :

$$\begin{array}{lll} \nu_i(R(\mathbf{q})) = \frac{1}{d+d_0} + \frac{d}{(d_1+d_0)^2} & \text{if} & 0 \leq d \leq d_1 \\ \nu_i(R(\mathbf{q})) = \frac{1}{d_1+d_0} + \frac{d_1}{(d_1+d_0)^2} & \text{if} & d > d_1 \end{array}$$

Where d_0 is a parameter that enables the potential as a function of d to be defined on the interval $[0, +\infty]$.

The gradient of the potential in the configuration space is obtained by multiplying the force in the workspace by the Jacobian of $R(\mathbf{q})$ in eq. 4.

We have presented an explicit manner to compute an obstacle potential field which satisfies our assumptions. The gradient of this potential is the criterion minimized by the optimization method, so that the deformation makes the path move away from obstacles.

B. Bounded curvature of a Car-like Robot

We have integrated the non-holonomic path deformation method on a robot with a car-like kinematic : the CyCab. As a car, the CyCab has a bounded steering angle. We present an extension of the method that takes into account the curvature bound of the system during the deformation process. We explain how this extension can also be applied to a two-wheels robot.

1) Car Kinematic Model: A car-like robot has 4 configuration variables x, y, θ, ϕ as shown in figure 4 and two non-holonomic constraints. The front wheel angles are computed in order for the curvature center noted O to belong to the rear axle line. The angle ϕ is the angle that would place the curvature center of a wheel located in the middle of the front axle, on the same line.

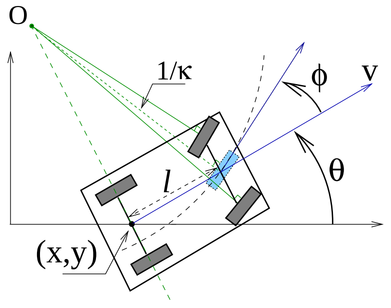

Fig. 3. Car Kinematic Model. Four configuration variables x, y, θ, ϕ. Curvature κ is derived from steering angle ϕ and distance l to rear axle.

The control vector fields for such a system are:

$$X_1 = \begin{pmatrix} \cos\theta \\ \sin\theta \\ \frac{\tan\varphi}{l} \\ 0 \end{pmatrix} X_2 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \tag{5}$$

2) How to Respect the Curvature Bound During the Deformation Process: As a car, the CyCab as a bounded curvature. That is: $0 \leq \kappa \leq \kappa_{max}$. It results in a constraint on its steering angle:
$0 \leq \phi \leq \phi_{max}$, with $\phi = \arctan(\kappa l)$.

If the non-holonomic path deformation method ensures that the wheels non-holonomic constraints of the system are respected, there is absolutely no guarantee that the deformed path will respect the curvature bound. And it can happen that given an initial path in collision, the deformation process produces a collision free-path with a non-admissible curvature for the system.

To counter this effect, the idea is to consider the steering angle bound ϕ_{max} as an obstacle. We define a potential on the steering angle ϕ in a similar way as the obstacle potential field II-A. The gradient which derives from this potential has the following expression: $\mathbf{f}_\phi(\mathbf{q}) =$

$$\begin{array}{lll} 0 & \text{if} & 0 \leq |\phi| \leq (\phi_{max} - d_1) \\ \frac{1}{((\phi-\phi_{max})+d_0)^2} - \frac{1}{(d_1+d_0)^2} & \text{if} & (\phi_{max} - d_1) \leq |\phi| \leq \phi_{max} \\ \frac{1}{(d_0)^2} - \frac{1}{(d_1+d_0)^2} & \text{if} & |\phi| \geq \phi_{max} \end{array} \tag{6}$$

where the parameters d_0 and d_1 represent the same magnitudes as in equation 4, but are here tuned to be homogeneous with angle values. Figure 5 represents the graph of $\|\mathbf{f}\phi(\mathbf{q})\|$. When at a configuration the value of ϕ is close to its bound, the gradient of the potential increases.

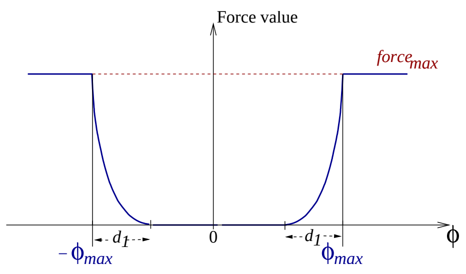

Fig. 4. Norm of the "force" $\mathbf{f}_\phi(\mathbf{q})$ due to the steering angle. The gradient of the potential field is indeed equivalent to a force. The value of the norm of the "force" increases when the steering angle gets closer to its bound.

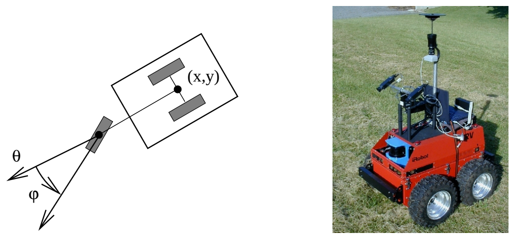

Fig. 5. Mobile robot Dala as a car-like robot: a non-holonomic system of dimension 4 with two non holonomic constraints.

3) A Virtual Wheel on a Two-Wheel Robot: We have also integrated the method on mobile robot Dala, an ATRV (see figure 6). This mobile robot is a differential-driven robot: rotation is performed by applying different velocities to the right and left wheels. In spite of its four wheels, it has the same kinematic as a two-wheel robot whose virtual axle would be located in the middle of the real axles. To avoid too much slipping and unnatural trajectories we consider robot Dala as a car-like robot with a bounded virtual steering angle. By using the same extension as before, we ensure this curvature bound is respected.

III. Effective Integration of the Method On Two Robots

The non-holonomic path deformation method is generic: it can be applied to any system subject to some non-holonomic constraints. We present in this section how we effectively integrated this method on two robots : CyCab and Dala. We describe the modular software architecture in which the method has been developed. We give some details about the way the obstacles are perceived, and about the localization issue in the context of the path deformation.

A. Modular Software Architecture

The algorithm of the deformation method has been implemented in a GenoM *module* [6]. A *module* is a software entity performing some functions, and capable of communicating with other modules. A *module* can for instance have some shared data structures that it updates, and that other modules can read.

Moreover the integration of the algorithm on robot CyCab had several constraints to cope with. First of all, the method has been developed at LAAS in Toulouse, and the robot CyCab is located at INRIA in Grenoble. Then the method had already been implemented in GenoM architecture, and we wanted to reuse these modules. Eventually the control algorithm of the CyCab had also already been implemented, but not in GenoM *modules*. Figure 7 presents the architecture chosen to cope with these constraints.

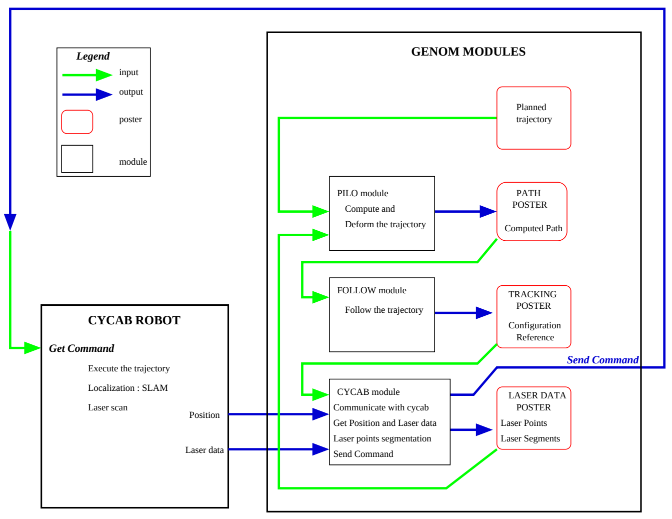

Fig. 6. Software Architecture of the Non-Holonomic path Deformation Method Integration on Robot CyCab. The CyCab communicates with GenoM modules: the two systems are distinct.

B. Obstacles Perception by the Robot

To compute the obstacles potential field along the path, we need to perceive obstacles. On both robots CyCab and Dala, a laser telemeter gives the distance to the closest obstacles in an horizontal plane in every direction. It returns at most 360 points for a 180° scan. A GenoM module is dedicated to the treatment of these points.

C. Localization of the Robot

The path deformation method can bear localization errors. If the robot is poorly localized during the execution of a given trajectory, a configuration that was planned as collision-free can happen to be in collision. But since the path deformation takes into account the distance to obstacles, it will deform the trajectory so that it moves away from obstacles. Thus, the path deformation can somehow make some localization errors non-critical.

The localization methods are different on robots CyCab and Dala:

On robot Dala an odometric sensor provides the linear and angular velocities, and the robot configuration is obtained by

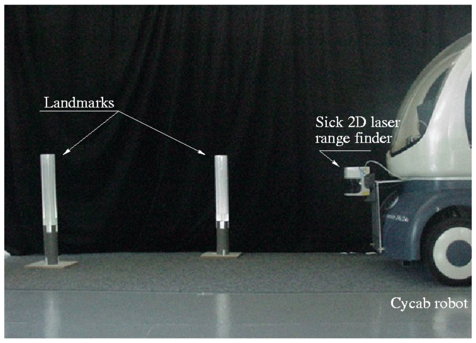

Fig. 7. Cycab robot and landmarks

integrating over time these latter. Thus, the drift is important and increasing.

On robot CyCab the localization is obtained by the fusion of odometric data and detection of landmarks (see [7] for details). Fig. 8 shows the CyCab, its sensor and the landmarks: cylinders covered with reflector sheets, specially designed for our Sick laser range finder. The landmarks are detected by the same laser telemeter sensor as the one used to detect obstacles. Due to an efficient use of advanced SLAM[1] techniques, the localization is much more accurate and does not suffer any drift (while in view of localization landmarks).

One could ask why we did not use a GPS to localize the robots. There are two reasons for that. The first reason is that a key point of the deformation method is its genericity. And we do not want to loose this genericity by taking the particular case of an exact localization of the robot. The second reason is that a accurate localization requires a differential GPS, which is not available in every environment.

IV. Experimental Results

In this section we present some experimental results consecutive to the integration of the path deformation method on the robots CyCab and Dala.

Given a model of the environment, the first step of the experiment consists in the computation of a collision-free path for the system. Then the computed trajectory is executed by the robot. At the time of execution, the robot detects obstacles, that were either :

- not in the model,
- imprecisely modeled,
- that appear to have moved because the robot does not follow exactly its trajectory since it is poorly localized.

The trajectory is deformed on line in order to move away from obstacles.

A. Experiment with robot Dala

Figure 11 presents an example of an initial trajectory in collision for robot Dala, which is iteratively deformed by the algorithm until collisions disappear. Figure 10 displays the input functions $u_1(s)$ and $u_2(s)$ (see eq. 5) computed at each iteration of the deformation process shown on figure 11 (both figures refer to the same experiment). In this example, the execution of the trajectory does not start before the collisions near the start configuration have been cleared.

[1]Simultaneous Localization And Mapping

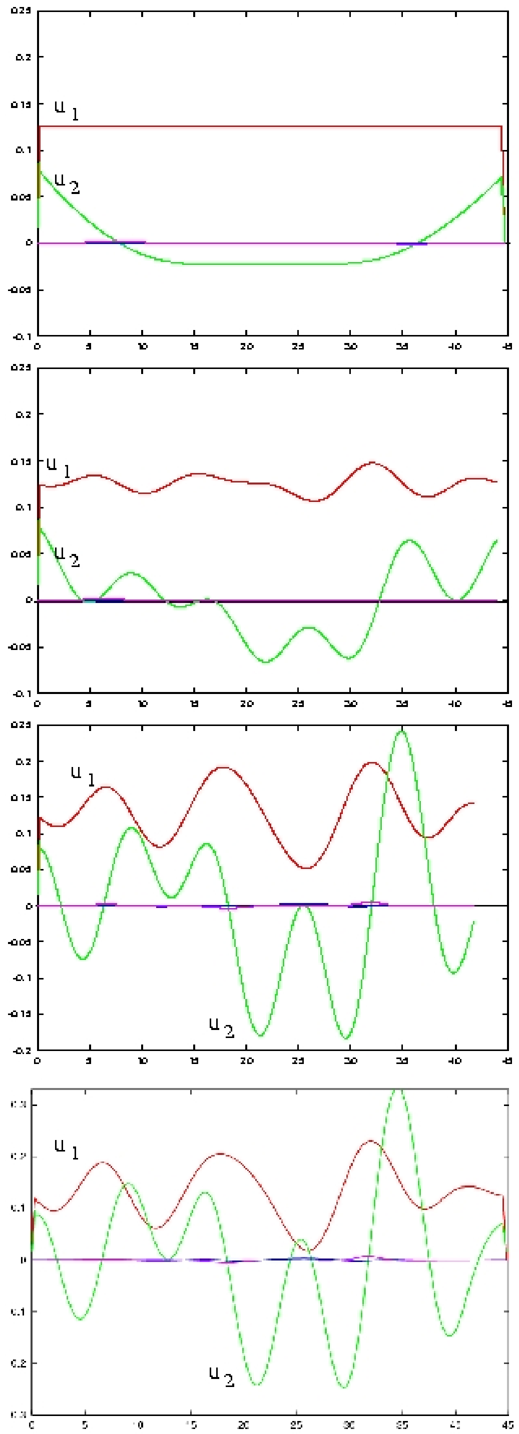

Fig. 9. *Input perturbations : effect of the deformation on inputs $u_i(s)$ along the trajectory ($s \in [0, 45]$).* At the top, the inputs computed for the planned path, and below the input computed by the deformation. As the deformation process goes along, u_1 and u_2 are more and more perturbed, in order to avoid the obstacles.

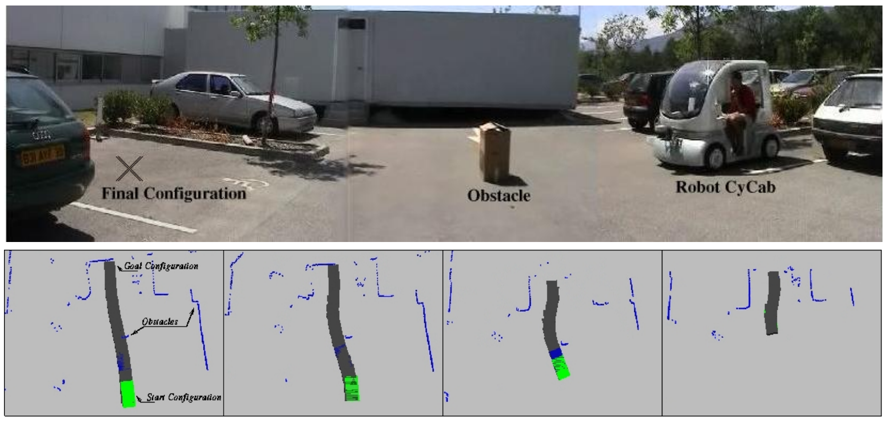

Fig. 8. *A trajectory planned by robot CyCab from a parking lot to another.* On top, experiment situation. On bottom, the trajectory deformation, bird's-eye viewed. An obstacle lies on the trajectory, and the CyCab has to deform its trajectory at execution time.

B. Experiment with robot CyCab

Figure 9 presents an experiment of automatic parking. Robot CyCab has to move from a parking lot to another. An obstacle which was not in the map used to plan the trajectory, lies between the two lots. The CyCab deforms its trajectory to avoid this new obstacle while keeping its curvature below its bound.

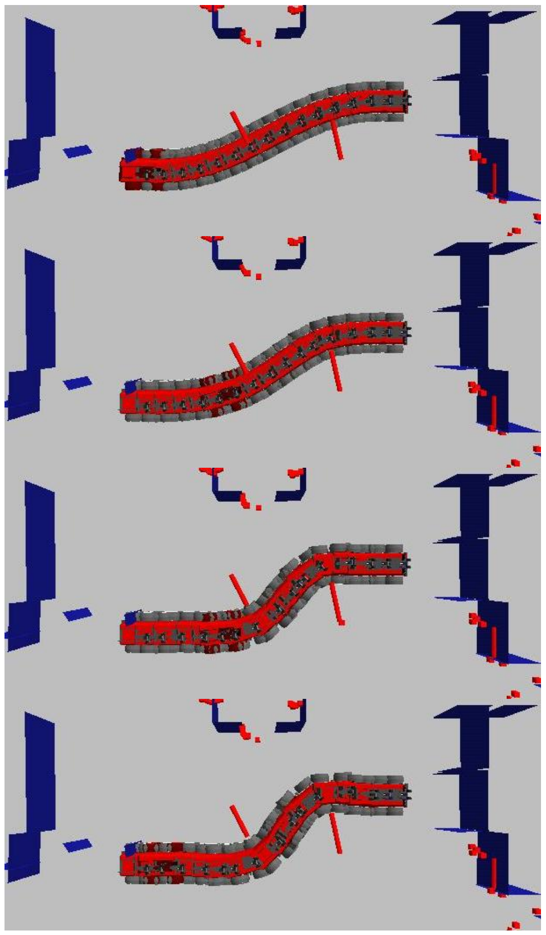

Fig. 10. *Trajectory deformation.* Two obstacles detected by a laser range finder lie on the path planned by the robot Dala (at the top). The trajectory is deformed in order to get a collision free path (at the bottom).

Conclusion

In this paper we have presented a generic non-holonomic path deformation method. It enables a given trajectory for a system to be deformed on line so that it moves away from obstacles. It ensures the non-holonomic constraints keep satisfied after the deformation.

We have extended the method in order for the deformation process to respect a given curvature bound of the trajectory. For that we have defined a potential field on the steering angle, in a similar way to the obstacles potential field.

These experimental results validate the genericity of the approach. Furthermore, effective implementation on several different platforms (Dala, CyCab) with different architectures (Linux, Sun, VxWorks,...) proves the genericity and the portability of involved techniques.

Acknowledgment: This work has been partially supported by the European Project MOVIE (IST-2001-39250).

References

[1] S. Quinlan, "Real-time modification of collision free paths," Ph.D. dissertation, Stanford University, 1995.

[2] M. Khatib, H. Jaouni, R. Chatila, and J.-P. Laumond, "Dynamic path modification for car-like nonholonomic mobile robots," in *International Conference on Robotics and Automation*. Albuquerque, NM: IEEE, Apr. 1997.

[3] O. Brock and O. Khatib, "Real-time replanning in high dimensional configuration spaces using sets of homotopic paths," in *International Conference on Robotics and Automation*. San Francisco, CA: IEEE, Apr. 2000, pp. 550–555.

[4] F. Lamiraux, D. Bonnafous, and C. V. Geem, *Control Problems in Robotics*. Springer, 2002, ch. Path Optimization for Nonholonomic Systems: Application to Reactive Obstacle Avoidance and Path Planning.

[5] F. Lamiraux and D. Bonnafous, "Reactive trajectory deformation for nonholonomic systems: Application to mobile robots," in *IEEE International Conference on Robotics and Automation*, 2002, pp. pp 3099–3104.

[6] S. Fleury, M. Herrb, and R. Chatila, "Genom: a tool for the specification and the implementation of operating modules in a distributed robot architecture," in *IROS, Grenoble, France*, vol. 2, Sept. 1997, pp. 842–848.

[7] C. Pradalier and S. Sekhavat, "Concurrent localization, matching and map building using invariant features," in *Proc. IEEE Int. Conf. on Intelligent Robots and Systems*, 2002.