2008 IEEE International Conference on
Robotics and Automation
Pasadena, CA, USA, May 19-23, 2008

Inverse versus Direct Kinematics Model based on Flatness and Escape Lanes to control CyCab Mobile Robot

Nicolas Morette, Cyril Novales and Pierre Vieyres

Laboratoire Vision & Robotique, 63 av. de Lattre, F-18020 Bourges cedex
Nicolas.Morette@bourges.univ-orleans.fr

Abstract— **This paper presents two approaches developed to control our CyCab mobile robot. The first one introduces the notion of virtual axle to determine a flat Inverse Kinematics Model; the second one, based on the Escape Lanes method, uses the Direct Kinematics Model. In the last part of this paper, a comparison between the two methods is carried out in order to validate the inverse model and to discuss the two methods respective advantages.**

I. INTRODUCTION

The control of autonomous robots relies on mathematical models providing relationships between robot trajectories and system inputs. For wheeled mobile robots, kinematics models are common and easy to use as they directly relate the robot displacements in its environment with respect to the inputs. However, using these models implies the assumption that ground slipping is negligible [3].

The direct kinematics model (DKM), $\dot{X} = J.\dot{q}$, allows to express the robot displacement velocities -typically in the Cartesian space- as a function of the inputs in the articular space; this function is always well defined. However, determining articular inputs, corresponding to a desired trajectory in the Cartesian space using an inverse kinematics model (IKM), is not always possible due to non-holonomy or to several existing solutions (i.e. singularity): the J matrix is rarely a square matrix.

Navigation systems for mobile robots are generally based on inverse models and use analytic methods to solve these problems [1][12][14]. These methods are often difficult to apply in practice; they require important computation capacities and are specific of a given robot, and do not always guarantee a solution. Navigators based on direct kinematics models are easier to implement. They typically use trajectory projections and are limited to the local environment of the robot [2][10]. The complete trajectory is built part after part and the global behavior is hardly predictable.

In this paper, we first introduce the notion of virtual axle to determine an inverse kinematics model for the CyCab automated vehicle. In the second part, a navigation method based on the robot DKM is presented in order to validate, in the last part of this paper, the IKM.

II. INVERSE KINEMATICS MODEL BASED ON FLATNESS

Our CyCab robot is an electrical bi-steerable car with four driving and steering wheels (fig. 1). When performing a turn, the CyCab front wheels present an antagonist angle to the rear wheels, in order to minimize the robot turning radius. This property enhances the robot maneuverability in cluttered environments. Moreover, there is a proportionality coefficient k between the rear and the front steering angle: *i.e.* for a front wheels steering angle of ξ, the rear wheels steering angle value is $k.\xi$ ($k \in R^+$).

Fig. 1. The CyCab robot[1]

A. Control point of the robot

When considering a top view of such a robot during a turning maneuver (fig. 2), we can assume there is a non-steering virtual axle between the front and the rear axles. This virtual axle is orthogonal to the main axis of the vehicle, and the instantaneous center of gyration belongs to this axle. If L is the distance between the rear and front axles, the virtual axle passes through the point C, at χL from the front axle.

When choosing point C as the control point of our mobile robot, we can directly benefit from the flat model of classical car-like robots.

The distance from the control point to the front and rear axles is given by (Fig. 2):

[1] This first prototype of CyCab was designed by INRIA Rhone-Alpes

$$\chi L = r\tan(\xi) \quad \text{and} \quad (1-\chi)L = r\tan(k\xi) \qquad (1)$$

By dividing these two equations leads to:

$$\chi = \frac{\tan(\xi)}{\tan(k\xi) + \tan(\xi)} \qquad (2)$$

From equation (2), the control point position on the robot is determined. Two different cases can be studied: k = 1 (front and rear steering angles are equal) and k ≠ 1.

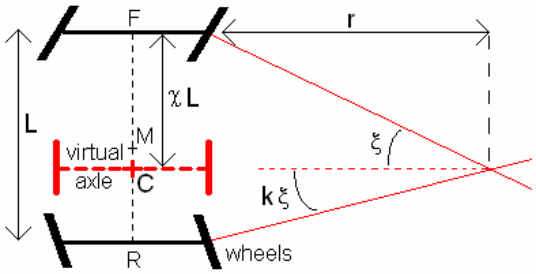

Fig. 2. The CyCab mechanical system

B. Definition of a flat system

A system $\dot{X} = f(X,u)$ is said to be differentially flat if there exist flat (or linearized) outputs $Z = (z_1, z_2 ..., z_m)$ differentially independent such that:

- any system variable (state, controls…) can be expressed only from the linearized outputs and their successive derivatives

- the flat outputs can be expressed as a function of the system variables and their successive derivatives [15]

Under these conditions, this kind of system is equivalent to a linear one when using an endogenous feedback [6]. However, the main problem about flatness, remain the fact that there exists no systematical method to compute the flat outputs of a system. In part C, we present a method to calculate flat outputs (x, y) of the Cycab bi-steerable system given on the control point of the robot.

C. Case k = 1: identical rear/ front steering angle

In this case χ = 0.5 and point C is fixed on the barycenter M of the four wheels. We obtain almost the same kinematics equations as the car-like vehicle ones:

$$\dot{x}_C = v.\cos\theta \quad \text{and} \quad \dot{y}_C = v.\sin\theta \qquad (3)$$

$$\dot{\theta} = 2v.\tan\left(\frac{\xi}{L}\right) \qquad (4)$$

Where (x_C, y_C, θ, ξ) are the Cartesian coordinates of the control point C, the robot orientation and the front steering angle, respectively.

This model is a differentially flat system, so we can express the inputs (v, ξ) of the robot using flat outputs z_1, z_2, and a finite number of its derivatives:

$$z_1(t) = x_C(t) \quad \text{and} \quad z_2(t) = y_C(t) \qquad (5)$$

The inverse kinematics model of the robot is given by:

$$\theta = \arctan\left(\frac{\dot{z}_2}{\dot{z}_1}\right) \quad \text{and} \quad v = \sqrt{\dot{z}_1^{\,2} + \dot{z}_2^{\,2}}$$

$$\xi = \arctan\left(\chi L.\frac{\dot{z}_1\ddot{z}_2 - \ddot{z}_1\dot{z}_2}{\left(\dot{z}_1^2 + \dot{z}_2^2\right)^{\frac{3}{2}}}\right) \qquad (6)$$

D. Case k ≠ 1: different front/rear steering angle

In this case, we can note that χ becomes a trigonometric function of the steering angle ξ (2); this means that the control point C will move along the main axis of the robot. The relation between the barycenter M of the four wheels and the control point C is given by:

$$x_C = x_M + (\chi - 0.5).L.\cos\theta \qquad (7)$$

$$y_C = y_M + (\chi - 0.5).L.\sin\theta \qquad (8)$$

However, the inputs (v, ξ) using flat outputs x_C and y_C are not obtained similarly to the k=1 case as the control point position becomes a function of the steering angle.

On the previous CyCab prototype, the k value measured experimentally is about 0.7. On the CyCab presented here, this value can be modified to 1 by adding an extra mechanical piece onto the physical link between the two axles. Hence, it was decided to study cases where k ranges from 0.5 to 2 (k=2 corresponds to the inverse configuration). Fig. 3 shows that, for a steering angle ranging from –30° to 30° (maximum practical angles), the control point position is almost constant when ξ varies.

As the steering angle position presents small variations, we can assume it is constant in order to express the flat outputs model of the robot. Taylor Series are used to compute χ from (2):

$$\tan\xi = \xi + \sigma(\xi^3) \qquad (9)$$

$$\chi = \frac{1}{1+k} + \sigma(\xi^3) \qquad (10)$$

The robot inverse kinematics model is the same as in equation (6), but the control point C position, knowing the position of the barycenter M, is determined using formulas (7), (8) and (10).

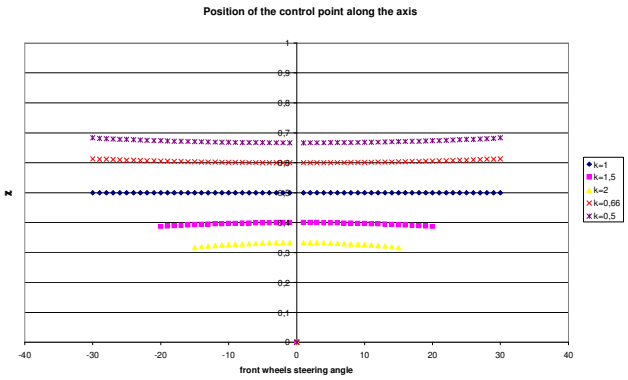

Fig. 3. Position of the control point

III. DIRECT KINEMATICS MODEL BASED ON ESCAPE LANES

In this part, a navigation method to generate trajectories for our CyCab robot is presented. This method uses the robot direct kinematics model.

Similarly to animal strategy, our navigation is based on direct models. For a mobile robot considered in a given state, we can project on a few seconds horizon all trajectories it can perform. These trajectories can be merged with the environment and with the motion goal in order to select the most appropriate trajectory. These selected trajectories become the new set points of the robot. In order to give the ability for the robot to react in real time in its environment, the process is repeated periodically.

The different steps of the escape lanes method are to:

- generate all acceptable trajectories Γ to be performed by the robot (called escape lanes) on a temporal horizon τ,
- eliminate the blocked escape lanes (e.g. intersecting or passing too close to obstacles),
- choose a free escape lane, using a selection criterion.

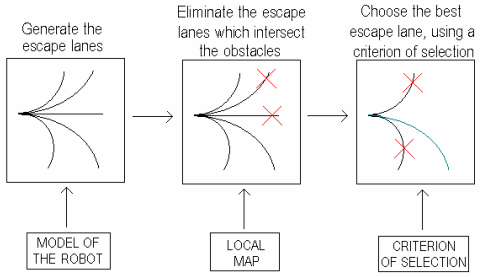

Fig. 4 Escape Lanes principle

The strong asset of this method is that the IKM does not need to be defined. The whole operation is reiterated periodically and is represented on Fig. 4.

A. *Acceptable Trajectories by the Robot and Escape Lanes*

An input function $\omega = (v, \xi)$, belonging to the input space U of the robot, is acceptable on an interval $[t_0, t_0+ \tau]$ when it respects the following constraints on:

-the direct kinematics model of the robot

-the actuators saturation,

-the dynamics of the actuators (maximum accelerations, saturations…)

$\Omega[t_0, t_0+ \tau]$ is called the set of the acceptable input functions on the interval $[t_0, t_0+ \tau]$:

$$\Omega[t_0, t_0+\tau] = \{\omega \in U[t_0, t_0+\tau] / \omega \text{ acceptable } \} \quad (11)$$

A trajectory Γ defined on an interval of time $[t_0, t_0+ \tau]$ is composed by the simplified state function ζ and the input function ω defined on the same interval: $\Gamma = (\zeta, \omega)$ [16].

$\Lambda[t_0, t_0+ \tau]$ is called the set of the robot acceptable trajectories on the interval $[t_0, t_0+ \tau]$. It corresponds to the set of trajectories associated with the input function $\omega \in \Omega[t_0, t_0+ \tau]$. $\Lambda[t_0, t_0+ \tau]$ is also called the robot escape lanes at time t_0 and on the temporal horizon τ.

$$\Lambda[t_0, t_0+\tau] = \{\Gamma = (\zeta, \omega) / \omega \in \Omega[t_0, t_0+\tau] \} \quad (12)$$

It is then necessary to choose a family of acceptable input functions that must be selected according to the robot capacities. For example, a car-like robot may not be controlled the same way as a robot with differential wheels. Linear functions are used for the CyCab robot (Fig. 5).

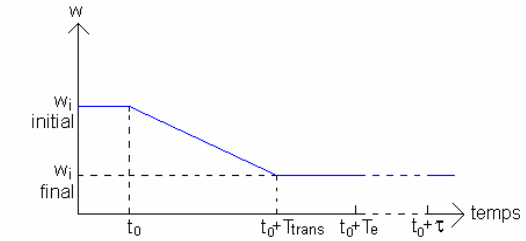

Fig. 5. Function of entry

This family of input functions can be expressed by:

$$\omega_i(t) = \omega_{i\,initial} + \frac{(\omega_{i\,final} - \omega_{i\,initial}) \times (t - t_0)}{T_{trans}} \quad for \quad t_0 < t < t_0 + T_{trans} \quad (13)$$

Where $\omega_1(t) = v(t)$ and $\omega_2(t) = \xi(t)$. From this family of acceptable input functions, a family of acceptable trajectories $\Lambda_{limited}$ is given by:

$$\Lambda_{LIMITED} = \{\Gamma = (\zeta, w_{familly}) / w_{familly} \in \Omega[t_0, t_0+\tau]\} \quad (14)$$

where:

$$w_{familly} = (v(t), \xi(t))^T \quad \zeta = (\dot{x}, \dot{y}, \dot{\theta}) \quad (15)$$

The corresponding trajectories are obtained using the robot DKM. R and F represent the middle points of the rear and the front axles, respectively (fig. 2). We express the distances between the control point C and these two points:

$$RC = (1 - \chi L) = L\frac{\cos(\xi).\sin(k\xi)}{\sin(\xi + k\xi)}$$
$$CF = \chi L = L\frac{\cos(k\xi).\sin(\xi)}{\sin(\xi + k\xi)} \quad (16)$$

The turning radius on the front (ρ_F) and on the rear (ρ_R) are given by:

$$\rho_R = L\frac{\cos(\xi)}{|\sin(\xi + k\xi)|}$$
$$\rho_F = L\frac{\cos(k\xi)}{|\sin(\xi + k\xi)|} \quad (17)$$

The derivative of the robot orientation is:

$$\dot{\theta} = \frac{v_R}{\rho_R} = \frac{v_F}{\rho_F} \quad (18)$$

The kinematics of the point M, which is the middle of the [RF] segment, can be expressed:

$$\dot{\theta}_M = v.\frac{\sin(\xi + k\xi)}{L.\cos(\xi)}$$
$$\dot{x}_M = \frac{1}{2}v(\cos(\theta + k\xi) + \cos(\theta + \xi)) \quad (19)$$
$$\dot{y}_M = \frac{1}{2}v(\sin(\theta + k\xi) + \sin(\theta + \xi))$$

Where $\dot{\theta}_M, \dot{x}_M, y_M$ are the rotation velocity and the translation velocities, respectively, of point M in the robot configuration space.

Fig. 6 presents a sample of possible trajectories, which can be performed by the robot, projected from different kinematics states.

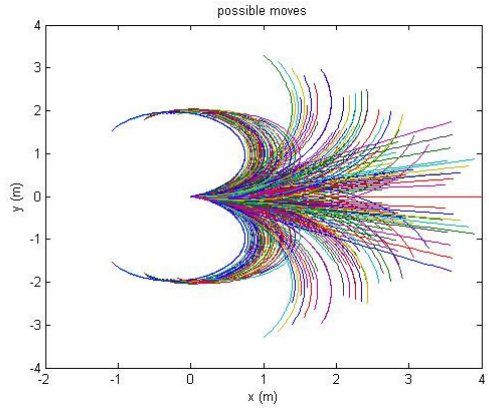

Fig. 6. Sample of possible trajectories of the robot

B. Blocked Escape Lanes Elimination

The next stage consists in comparing the image of each trajectory Γ belonging to Λ in the output space Ψ to the obstacles map called Θ. This map contains the obstacles position with respect to the robot position. Using an elimination criterion C_{el}, escape lanes which pass too close to the obstacles are eliminated. The remaining escape lanes constitute the free escape lanes set Λ_L.

The ensemble of the free acceptable input functions Ω_L can be defined as the entire set of the input functions ω which correspond to the free escape lanes Λ_L.

$$\Omega_L = \{\omega \in \Omega / , \Gamma \in \Lambda_L \} \quad (20)$$

To determine the obstacles map Θ, a local model of the robot environment is used [4], built in-line using 2 laser-range finders [3]. This map is composed as a set of segments of known two ends coordinates. These segments represent the obstacles perimeter in the robot local environment, projected within the moving plane of the robot. The C_{el} elimination criterion is used between its image Λ_i in the output space and each obstacle segment of Θ. $C_{el}(\lambda_i, \Theta) = 0$ if:

$$dist\{\lambda_i(t) / sgmt_j\} - (l + margin) > 0 \ \forall \ j. \quad (21)$$

Where l is the maximum distance between the periphery of the robot and the control point C; $dist\{\lambda_i(t) / sgmt_j\}$ is the minimal distance between λ_i points and segment j. The ensemble of the remaining escape lanes is $\Lambda_{L/limited}$.

To improve the relevance of the elimination criterion C_{el}, the margin value in the formula (24) may be adjusted according to the orientation of the robot with respect to the obstacle position (C-obstacles [9]).

C. Selection of the Best Free Escape Lane

Finally, a criterion is used to choose the escape lane to be kept for the robot. The choice of this criterion takes into account the distance and the orientation of the robot, at the end of the trajectory, compared to a passing point ε provided by a path planner. Indeed, the efficiency of this method can be enhanced when used with a path planner working on a global map of the environment. The path planner will provide a set of passing points to the escape lane navigator in order to reach a final goal. To select the best free escape lane, we used the following criterion:

$$C_{choice}(\Gamma_i,\varepsilon)=\sqrt{(x_i(t_0+\tau)-x_\varepsilon)^2+(y_i(t_0+\tau)-y_\varepsilon)^2}\times fact \qquad (22)$$

$$fact=1+k_\theta\times\left|\theta_i(t_0+\tau)-\theta_{\varepsilon/robot}\right| \qquad (23)$$

where k_θ is a positive real chosen empirically by carrying out simulations and experimentations.

This criterion computes the Cartesian distance from the robot to ε; this criterion is weighted by the robot orientation with respect to ε. The chosen $\Lambda_{L/limited}$ escape lane which minimizes this criterion is called Γ_{chosen}. The associated input function ω_{chosen} is applied to the robot.

The input function corresponding to the selected trajectory (ω_{chosen}) is actually applied to the robot. Indeed, the complete operation (from the generation of escape lanes to the selection of Γ_{chosen} and ω_{chosen}), is performed periodically, and with a period T_e about ten times smaller than the temporal horizon τ. Therefore, only a short part of the Γ_{chosen} trajectory is actually followed by the robot; a new one is proposed every T_e.

D. Simulation results

Fig. 7 shows simulation results representing displacements of our CyCab robot evolving on a 40^2 m² area. The trace of the displacements achieved by the control point is represented in blue, and each iteration of the navigation program is symbolized by a short line perpendicular to the trace. The red crosses represent the passing points provided to the navigator.

The distance between the robot and its nearest obstacle along the path is represented on fig. 8. The robot never enters in collision with the obstacles as the distance from obstacles is always larger than the robot width (1.44m).

This simulation validates the escape lanes direct model for bi-steerable like vehicles, such as CyCab. Moreover, an important advantage of this method is its intrinsic robustness to the perturbations. Indeed, since the robot moves are projected in an on-line refreshed local map, the errors on the real position of the robot are not integrated; this aspect guarantees the robustness of the method.

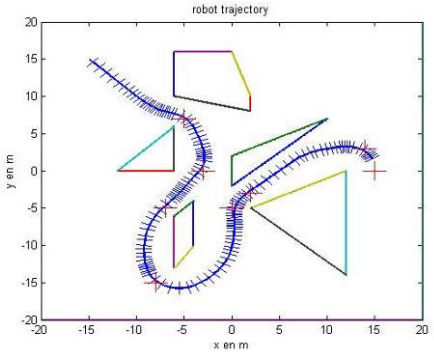

Fig. 7. Simulation results of obtained trajectories

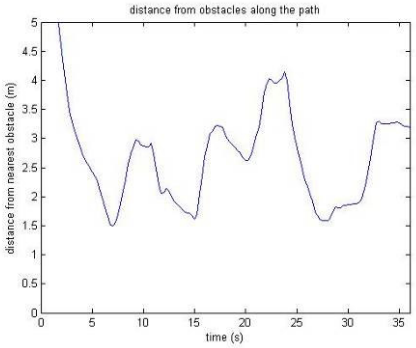

Fig. 8. Distances from obstacles along the path

IV. SYNTHESIS: FLAT OUTPUTS VS ESCAPE LANES

A comparison between the two models (the direct and the inverse one) is done. Starting from the trajectories obtained in the simulation described above, the robot velocities are computed along the path ($\hat{v}$) using the flat output inverse kinematics model (fig. 9). Hence, the input velocities, sent as inputs for the Escape Lanes method (v), can be compared to those computed by the flat outputs model.

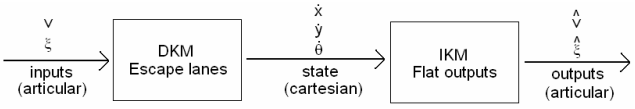

Fig. 9. Comparison principle

Fig. 10 shows the results for a front steering angle 2 times lower than the rear one. On this figure, the two velocities (input of the direct model and output of the inverse model) are represented, and the second graph shows the difference between them. For this test, in order to test the robustness of the model, a value of 2 was chosen for k as it corresponds to the maximal error on the control point position.

We notice that the output articular velocities are very close to those sent as input functions for each iteration of the Escape lanes method. These results show that our approximation, when the control point C position is

considered as a constant, does not generate a large error : in the worst case, the velocity error is less than 0.3 $m.s^{-1}$. This can be explained by the fact that the χ coefficient (3) shows little variations when the steering angle ξ remains (Fig. 3) within the angles range control (i.e. less than 30°) we use for the Cycab robot.

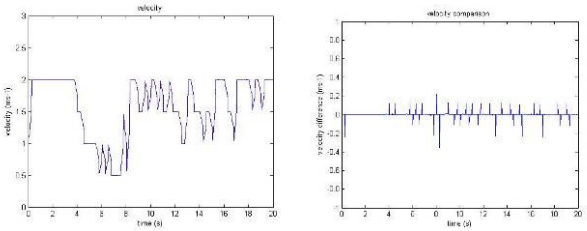

Fig. 10. Velocities along the path for k=2

V. CONCLUSION

In this paper, we developed an inverse kinematics model to control our CyCab mobile robot and we introduced the notion of virtual axle to carry out the robot control point. These results can be extended to the bi-steerable cars families whose rear steering angle is proportional to the front steering angle. Simulations have been presented in order to validate the Escape Lanes navigation method. Our CyCab has four driving and steering wheels, with a proportionality between the rear and front wheels steering angle. However, thanks to a linearization on the position of the control point, we have been able to apply almost the same flat output model as for a car-like robot. Then, the inverse kinematics model has been validated by a simulation. Currently, this model is going to be tested on the robot platform in order to check the validity of our simulation predictions.

Applications fields for these two models are different. The Escape Lanes method has been developed to control autonomous mobile robots in an unknown environment. The aim of this method is to generate dependable and reliable trajectories for the robot. However, this method has to be used complementary to a global path planning method in order to receive path points.

The Flat Output inverse kinematics model is foreseen to be used for trajectory following. Indeed, inverse kinematics goal is to calculate the inputs corresponding to a given trajectory (meaning that the trajectory has to be known). For example, inverse model can be used in order to tele-operate the mobile robot or to perform leader robot trajectory following for robots cooperation.

In this study, our interest is focused on two kinds of approaches. Our goal is to build a control architecture able to generate a reliable control approach, integrating tele-operation and autonomy switch modes; this appears as a necessity to counter the latencies inherent to tele-operation via the internet networks in order to perform complex tasks.

References

[1] Agirrebeitia, J., Aviles, R., de BUSTOS, I. F., Ajuria, G., June 2005, *A new APF strategy for path planning in environments with obstacles*, Mechanism and Machine Theory, Volume 40, Issue 6, , Pages 645-658

[2] Belker, T., Schulz, D., *Intelligent Robots and System, 2002, Local Action Planning for Mobile Robot Collision Avoidance*, IEEE/RSJ International Conference on Volume 1, 30 Sept.-5 Oct. 2002 Page(s):601 - 606 vol.1

[3] Campion, G.; Bastin, G.; Dandrea-Novel, B., *Structural properties and classification of kinematic and dynamic models of wheeled mobile robots,* Robotics and Automation, IEEE Transactions on Volume 12, Issue 1, Feb 1996 Page(s):47 - 62

[4] Canou, J., Mourioux, G., Novales, C. Poisson, G., April 26 – May 1st 2004 , *A local map building process for a reactive navigation of a mobile robot*, Proceedings of IEEE International Conference on Robotics and Automation, pp4839-4844, New Orleans, USA

[5] Conner, D.C., Choset, H., Rizzi, A.A., *Integrated Planning and Control for Convex-bodied Nonholonomic systems using Local Feedback Control Policies,* Proceedings of Robotics: Science and Systems II (RSS'06), August, 2006

[6] Fliess, M., Lévine, J., Martin, P., Rouchon, P., *Flatness and Defect of Nonlinear Systems : Introductory Theory and Examples*, Int. Journal of Control, 6, 61, 1995, p. 1327-1361.

[7] Fraisse, P., Gil, A. P., Zapata, R., Perruquetti, W., Divoux, T., 2002, *Stratégie de commande collaborative réactive pour des réseaux de robots*.

[8] Lebedev, D. V., April 2005. *The dynamic wave expansion neural network model for robot motion planning in time-varying environments, Neural Networks*, Volume 18, Issue 3, Pages 267-285.

[9] Lozano-Perez, T., 1983, *Spatial Planning: a Configuration Space Approach*, *IEEE Transaction and computers,* vol. C-32, no. 2, Fev. 1983.

[10] Novales, C., 1994, *Navigation Locale par Lignes de Fuite, rapport de thèse : Pilotage par actions réflexes et navigation locale de robots mobiles rapides*, chapitre IV, soutenue le 20 octobre 1994, Pages 87 à 107

[11] Novales, C., Mourioux, G., Poisson, G., April 6,7 2006 , *A multi-level architecture controlling robots from autonomy to teleoperation*, First National Workshop on Control Architectures of Robots – Montpellier

[12] Munoz, V., Ollero, A., Prado, M., Simon, A., 1994 IEEE International Conference on 8-13 May 1994, *Mobile robot trajectory planning with dynamic and kinematic constraints, Robotics and Automation*, 1994. Proceedings, Page(s):2802 - 2807 vol.4

[13] Papadopoulos, E., Poulakakis, I., Papadimitriou, I., *On Path Planning and Obstacle Avoidance for Nonholonomic Mobile Manipulators: a polynomial approachs,* in the International Journal of Robotics research, Vol. 21, No. 4, pp 367-383, April 2002

[14] Samson, C., Morin, P., *Commande par retour d'état de systèmes non-linéaires sans dérive. L'approche par fonctions transverses*, in: CIFA'2002 (Conférence Internationale Francophone d'Automatique), Nantes, France, juillet 2002.

[15] Sekhavat, S., Hermosillo, J., Rouchon, P., *Motion planning for a Bi-Steerable Car*, IEEE International Conference on Robotics and Automation, ICRA'2001. Seoul, Korea. May 21-26, 2001

[16] Sontag, E. D., 1990. Mathematical control Theory – Deterministic Finite Dimensional Systems, ED- Spronger-Velag New-York 1990.