Orbital Obstacle Avoidance Algorithm for Reliable and On-Line Mobile Robot Navigation

Lounis Adouane

Orbital Obstacle Avoidance Algorithm for Reliable and On-Line Mobile Robot Navigation

Lounis Adouane
LASMEA, UBP-UMR CNRS 6602, France
Email: Lounis.Adouane@lasmea.univ-bpclermont.fr

***Abstract*— This paper proposes an orbital obstacle avoidance algorithm which permits to obtain safe and smooth robot navigation in very cluttered environments. This algorithm uses specific reference frame which gives accurate indication on robot situation. The robot knows thus if it must avoid the obstacle in clockwise or counter-clockwise direction. Moreover, it knows the moment to go into the orbit of the obstacle and the moment to go out. These orbital behaviors are performed using adaptive limit-cycle trajectories. The later with a specific conflicting situations module permit to prevent robot oscillations, local minima and dead ends. The proposed algorithm is embedded in a specific bottom-up control architecture with stability proof given according to Lyapunov synthesis. The overall proposed structure of control allows to decrease significantly the time to reach the target. In fact, according to the proposed algorithm, robot anticipates the collisions with obstacles according to smooth local trajectory modifications. A large number of simulations in different environments are performed to demonstrate the efficiency and the reliability of the proposed control architecture.**

I. Introduction

Obstacle avoidance controllers have a predominating function to achieve autonomously and safely the navigation of mobile robots in cluttered and unstructured environments. Khatib in [1] proposes a real-time obstacle avoidance approach based on the principle of artificial potential fields. He assumes that the robot actions are guided by the sum of attractive and repulsive fields. Arkin in [2] extends Khatib's approach while proposing specific schema motors for mobile robots navigation. Nevertheless, these methods suffer from the local minima problem when for instance, the sum of local gradient is null. In [3], Elnagar et al., propose to model the repulsive potential field characterizing obstacles by Maxwell's equations which have the merit to completely eliminate the local minima problem. Fuzzy control is widely used to perform robust obstacle avoidance [4], [5]. This formalism allows to integrate several linguistic rules to avoid dead ends or local minima [6]. Unfortunately, its lacks of stability demonstration of the applied control laws. Another interesting approach, based on a reflex behavior reaction, uses the Deformable Virtual Zone (DVZ) concept, in which a robot kinematic dependent risk zone surrounds the robot [7]. If an obstacle is detected, it will deform the DVZ and the approach consists to minimize this deformation by modifying the control vector. An interesting overview of other obstacle avoidance methods is accurately given in [8].

Nevertheless, the obstacle avoidance controller is only a part of the different functions which must constitute an overall control architecture for navigation tasks. One part of the literature in this domain considers that the robot is fully actuated with no control bound and focuses the attention on path planning. Voronoï diagrams and visibility graphs [9] or navigation functions [10] are among these roadmap-based methods. However, the other part of the literature considers that to control a robot with safety, flexibility and reliability, it is essential to accurately take into account: robot's structural constraints (e.g., nonholonomy); avoid command discontinuities and set-point jerk, etc. Nevertheless, even in this method, there are two schools of thought, one uses the notion of planning and re-planning to reach the target, e.g., [11] and [12] and the other more reactive (without planning) like in [13], [14] or [15]. Our proposed control architecture is linked to this last approach. Therefore, where the stability of robot control is rigourously demonstrated and the overall robot behavior is constructed with modular and bottom-up approach [16]. The proposed on-line obstacle avoidance algorithm uses specific orbital trajectories given by limit-cycle differential equations [17], [18]. The proofs of controllers stability are given using Lyapunov functions. The proposed algorithm provides also several mechanisms to prevent oscillations, local minima and dead end robot situations.

The rest of the paper is organized as follows. Section II gives the specification of the task to achieve. The details of the proposed control architecture are given in section III. It presents the model of the considered robot and the implemented elementary controllers laws. Section IV gives in details the proposed obstacle avoidance algorithm whereas section V introduces the conflicting situations management module. Section VI is devoted to the description and analysis of simulation results. This paper ends with some conclusions and further work.

II. Navigation in presence of obstacles

The objective of the navigation task in an unstructured environment is to lead the robot towards one target while avoiding statical and dynamical obstacles. One supposes in the setup that obstacles and the robot are surrounded by bounding cylindrical boxes with respectively R_O and R_R radii [19]. The target to reach is also characterized by a circle of R_T radius. Several perceptions are also necessary for the robot navigation (cf. Figure 1):

- D_{ROi} distance between the robot and the obstacle "i",

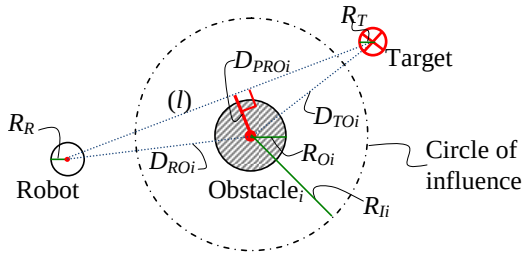


Fig. 1. The used perceptions for mobile robot navigation

- D_{PROi} perpendicular distance between the line (l) and the obstacle "i",
- D_{TOi} distance between the target and the obstacle "i".

For each detected obstacle we define a *circle of influence* (cf. Figure 1) with a radius of $R_{Ii} = R_R + R_{Oi} + Margin$. *Margin* corresponds to a safety tolerance which includes: perception incertitude, control reliability and accuracy, etc.

III. Control Architecture

The proposed structure of control (cf. Figure 2) aims to manage the interactions between elementary controllers while guaranteing the stability of the overall control as proposed in [15]. Its objective is also to obtain safe, smooth and fast robot navigation. It will permit for example to an autonomous application of travelers transportation [20] to have more comfortable displacements of the passengers. The specific blocks composing this control are detailed below.

A. Hierarchical action selection

Most reactive approaches activate the obstacle avoidance controller only when the robot is close to an obstacle (i.e. $D_{ROi} \leq R_{Ii}$) (cf. Figure 3(a)) [2], [21], [22], etc. In contrast, the proposed algorithm 1 activates the obstacle avoidance controller as soon as it exists at least one obstacle that can obstruct the future robot movement toward the target (i.e. $D_{PROi} \leq R_{Ii}$, cf. Figure 1). Thus, while anticipating the activation of obstacle avoidance controller (cf. Figure 3(b)), Algorithm 1 permits to decrease the time to reach the target, especially in very cluttered environments (cf. Section VI).

The proposed control architecture uses a hierarchical action selection mechanism to manage the switch between two or even more controllers. Obstacle avoidance strategy is integrated in a more global control architecture unlike what is proposed in [24]. Otherwise, the controller activations are achieved in a reactive way as in [23] or [16].

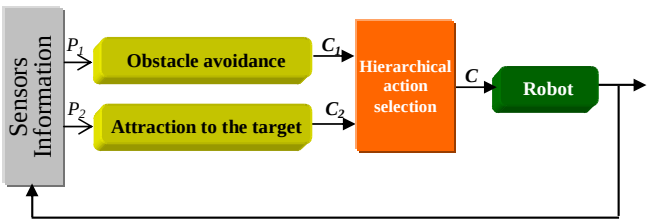
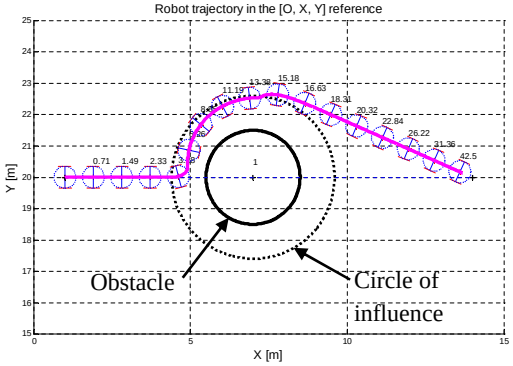
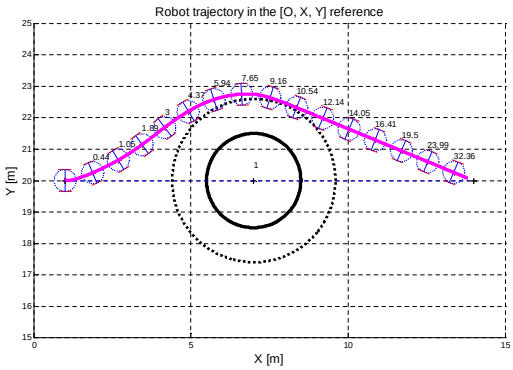


Fig. 2. Control architecture for mobile robot navigation



(a) Without orbital algorithm



(b) With orbital algorithm

Fig. 3. Robot trajectory while anticipating or not the obstacle collision

if *It exists at least one constrained obstacle* {*i.e.*, $D_{PROi} \leq R_{Ii}$ *(cf. Figure 1)* } **then**
  Activate obstacle avoidance controller
else
  Activate the attraction to the target controller
end

Algorithm 1: Hierarchical action selection

B. Elementary controllers

Each controller composing the control architecture (cf. Figure 2) is characterized by a stable nominal law. These laws are synthesized according to Lyapunov theorem. We will present here only some details about the stability demonstration of the used laws. More details are given in [25]. Before describing each elementary controller, let's show the used kinematic robot model (cf. Figure 4):

$$\dot{\xi} = \begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} \cos\theta & -l_2\cos\theta - l_1\sin\theta \\ \sin\theta & -l_2\sin\theta + l_1\cos\theta \\ 0 & 1 \end{pmatrix} \begin{pmatrix} v \\ w \end{pmatrix} \quad (1)$$

with:

- x, y, θ: configuration state of the unicycle at the point "P_t" of abscissa and ordinate (l_1, l_2) according to the mobile reference frame (X_m, Y_m),
- v: linear velocity of the robot at the point "P_t",
- w: angular velocity of the robot at the point "P_t".

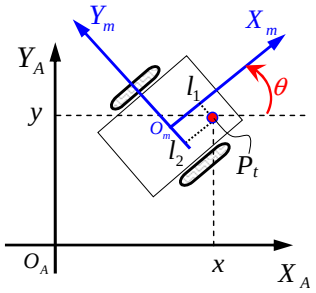


Fig. 4. Robot configuration in a cartesian reference frame

1) Attraction to the target controller: This controller guides the robot toward the target which is represented by a circle of (x_T, y_T) center and of R_T radius (cf. Figure 1). The used control law is a control of position at the point P_t = (l_1, 0) (cf. Figure 4). As we consider a circular target with R_T radius, therefore, to guarantee that the center of robot axis reaches the target with asymptotical convergence, l_1 must be $\leq R_T$ (cf. Figure 4).

$$\begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} = \begin{pmatrix} \cos\theta & -l_1 \sin\theta \\ \sin\theta & l_1 \cos\theta \end{pmatrix} \begin{pmatrix} v \\ w \end{pmatrix} = M \begin{pmatrix} v \\ w \end{pmatrix} \quad (2)$$

with M invertible matrix.

The errors of position are: $\begin{cases} e_x = x - x_T \\ e_y = y - y_T \end{cases}$

The position of the target is invariable according to the absolute reference frame (cf. Figure 6) $\Rightarrow \begin{cases} \dot{e}_x = \dot{x} \\ \dot{e}_y = \dot{y} \end{cases}$

Classical techniques of linear system stabilization can be used to asymptotically stabilize the error to zero [26]. We use a simple proportional controller which is given by:

$$\begin{pmatrix} v \\ w \end{pmatrix} = -KM^{-1} \begin{pmatrix} e_x \\ e_y \end{pmatrix} \quad (3)$$

with $K > 0$. Let's consider the following Lyapunov function

$$V_1 = \tfrac{1}{2} d^2 \quad (4)$$

with $d = \sqrt{e_x^2 + e_y^2}$ (distance robot-target).

Therefore, to guarantee the asymptotical stability of the proposed controller, $\dot{V}_1$ must be strictly negative definite, so, $d\dot{d} < 0$, what is easily proven as long as $d \neq 0$.

2) Obstacle avoidance controller: To perform the obstacle avoidance behavior, the robot needs to fellow accurately limit-cycle vector fields [18], [24], [27], [15]. These vector fields are given by two differential equations:

- For the clockwise trajectory motion (cf. Figure 5(a)):

$$\begin{aligned} \dot{x}_s &= y_s + x_s(R_c^2 - x_s^2 - y_s^2) \\ \dot{y}_s &= -x_s + y_s(R_c^2 - x_s^2 - y_s^2) \end{aligned} \quad (5)$$

- For the counter-clockwise trajectory motion (cf. Figure 5(b)):

$$\begin{aligned} \dot{x}_s &= -y_s + x_s(R_c^2 - x_s^2 - y_s^2) \\ \dot{y}_s &= x_s + y_s(R_c^2 - x_s^2 - y_s^2) \end{aligned} \quad (6)$$

where (x_s, y_s) corresponds to the position of the robot according to the center of the convergence circle (characterized by an R_c radius). Figure 5 shows that the circle of "$R_c = 1$" is a periodic orbit. This periodic orbit is called a limit-cycle. Figure 5(a) and 5(b) show the shape of equations (5) and (6) respectively. They show the direction of trajectories (clockwise or counter-clockwise) according to (x_s, y_s) axis. The trajectories from all points (x_s, y_s) including inside the circle, move towards the circle.

The proposed control law which permits to follow these trajectories is an orientation control, the robot is controlled according to the center of its axle, i.e., while taking (l_1, l_2) = (0, 0) (cf. Figure 4). The desired robot orientation θ_d is given by the differential equation of the limit-cycle (5) or (6) as:

$$\theta_d = \arctan(\frac{\dot{y}_s}{\dot{x}_s}) \quad (7)$$

and the error by

$$\theta_e = \theta_d - \theta \quad (8)$$

We control the robot to move to the desired orientation by using the following control law:

$$w = \dot{\theta}_d + K_p \theta_e \quad (9)$$

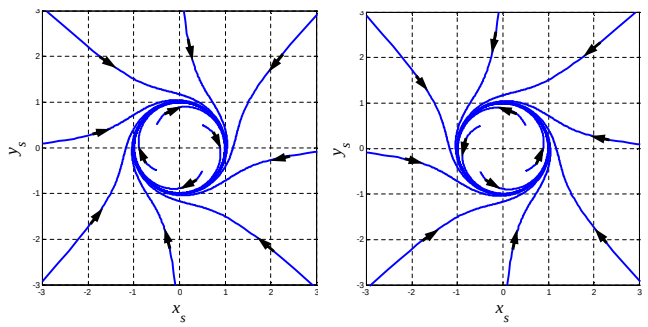
with K_p a constant > 0, $\dot{\theta}_e$ is given then by:

$$\dot{\theta}_e = -K_p \theta_e \quad (10)$$

Let's consider the following Lyapunov function

$$V_2 = \tfrac{1}{2} \theta_e^2 \quad (11)$$

$\dot{V}_2$ is equal then to $\theta_e \dot{\theta}_e = -K_p \theta_e^2$ which is always strictly negative (so, asymptotically stable). It is to note that the nominal speed of the robot v when this controller is active is a constant.



(a) Clockwise (b) Counter-Clockwise

Fig. 5. Shape possibilities for the used limit-cycles

IV. Orbital obstacle avoidance algorithm

In what follows, the overall methodology to achieve the proposed obstacle avoidance algorithm will be given. The algorithm is developed according to stimuli-response principle. To implement this kind of behavior it is important to:

- detect the obstacle to avoid (cf. Section II),
- give the direction of the avoidance (clockwise or counter-clockwise),
- define an escape criterion which defines if the obstacle is completely avoided or not yet.

All these different steps must be followed and applied while guaranteeing that: the robot trajectory is safe, smooth and avoids undesirable situations as deadlocks or local minima ; and that the stability of the applied control law is guaranteed. The necessary steps to carry out the obstacle avoidance algorithm (2) are given below:

1) For each sample time, obtain the distance D_{ROi} and perpendicular distance D_{PROi} for each potentially disturbing obstacle "i" (i.e., $D_{PROi} \leq R_{Ii}$) (cf. Figure 1),
2) Among the set of disturbing obstacles (which can constrain the robot to reach the target), choose the closer to the robot (the smallest D_{ROi}). This specific obstacle has the following features: radius R_{Oi} and (x_{obst}, y_{obst}) position,
3) After the determination of the closest constrained obstacle, we need to obtain four specific areas (cf. Figure 6) which give the robot behavior: clockwise or counter-clockwise obstacle avoidance ; repulsive or attractive phase (cf. Algorithm 2). To distinguish between these 4 areas we need to:
   - define a specific reference frame which has the following features (cf. Figure 6):
     - the X_O axis connects the center of the obstacle (x_{obst}, y_{obst}) to the center of the target. This axis is oriented towards the target,
     - the Y_O axis is perpendicular to the X_O axis and it is oriented while following trigonometric convention.
   - apply the reference frame change of the position robot coordinate $(x, y)_A$ (given in absolute reference frame) towards the reference frame linked to the obstacle $(x, y)_O$. The transformation is achieved while using the following homogeneous transformation:

$$\begin{pmatrix} x \\ y \\ 0 \\ 1 \end{pmatrix}_O = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 & x_{obst} \\ \sin\alpha & \cos\alpha & 0 & y_{obst} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} \begin{pmatrix} x \\ y \\ 0 \\ 1 \end{pmatrix}_A \quad (12)$$

Once all necessary perceptions are obtained, one can apply the proposed orbital obstacle avoidance strategy given by Algorithm 2. To obtain the set-points, it is necessary to obtain the radius "R_c" and the direction "clockwise or counter-clockwise" of the limit-cycle to follow. The position (x_O, y_O) gives the configuration (x, y) of the robot according to obstacle reference frame. The definition of this specific reference frame gives an accurate means to the robot to know what it must to do. In fact, the sign of x_O gives the kind of behavior which must be taken by the robot (attraction or repulsion). In repulsive phase, the limit-cycle takes different radii to guarantee the trajectory smoothness. The sign of y_O gives the right direction to avoid the obstacle. In fact, if $y_O \geq 0$ then apply clockwise limit-cycle direction else apply counter-clockwise direction. This choice permits to optimize the length of robot trajectory to avoid obstacles. Nevertheless, this direction is forced to the direction taken just before if the obstacle avoidance controller was already active at $(t - \delta T)$ instant (cf. Section V-B).

Input: All the features of the closest obstacle
Output: Features of the limit-cycle trajectory to follow

//I) Obtaining the radius "R_c" of the limit-cycle
if $x_O \leq 0$ **then**
  $R_c = R_{Ii} - \xi$ (**Attractive phase**)
  {*with ξ a small constant value as $\xi \ll$ Margin (cf. Section II) which guarantees that the robot do not navigate very closely to the R_{Ii} radius (which causes the oscillations of the robot (cf. Figure 9))*}
else
  {*Escape criterion: go out of the obstacle circle of influence with smooth way*}
  R_c = R_c + ξ (**Repulsive phase**)
end
//II) Obtaining the limit-cycle direction
if *obstacle avoidance controller was active at $(t - \delta T)$ instant* **then**
  Apply the same direction already used, equation (5) or (6) is thus applied.
  {*This will permit to avoid several conflicting situations (cf. Rule 2 below)*}
else
  {*The limit-cycle set-point is given by:*}
  $\dot{x}_s = sign(y_O)y_s + x(R_c^2 - x_s^2 - y_s^2)$
  $\dot{y}_s = -sign(y_O)x_s + y(R_c^2 - x_s^2 - y_s^2)$
end

Algorithm 2: Obstacle avoidance algorithm

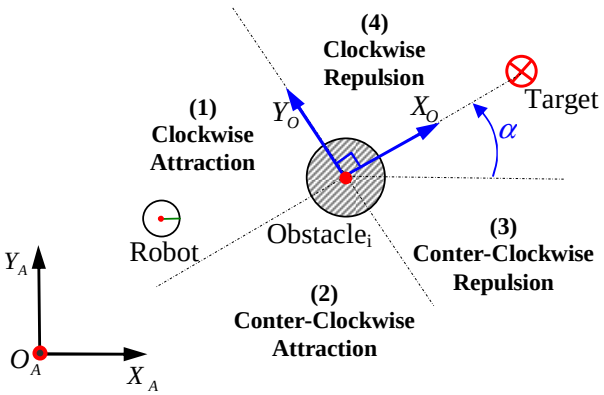


Fig. 6. The 4 specific areas surrounding the obstacle to avoid

V. Conflicting situations management

The good performance of proposed algorithm 2 need to manage some conflicting situations which are due to local minima or dead ends. The rules used to avoid these situations are given below.

A. Rule 1 - What obstacle to avoid?

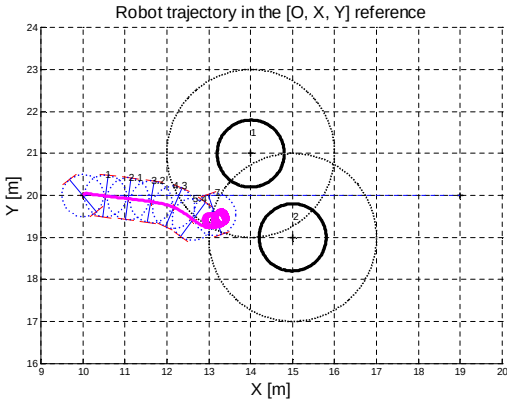
if *Two or more constrained obstacles have the same value of the distance* D_{ROi} *(cf. Figure 1)* **then**
  the robot will choose to avoid the one with the smallest D_{PROi}
end
if *It is already the same* D_{PROi} **then**
  the robot will choose the smallest obstacle D_{TOi} (cf. Figure 1)
end
if *It is already the same* D_{TOi} **then**
  choose arbitrary one of these obstacles
end

Algorithm 3: Rule 1

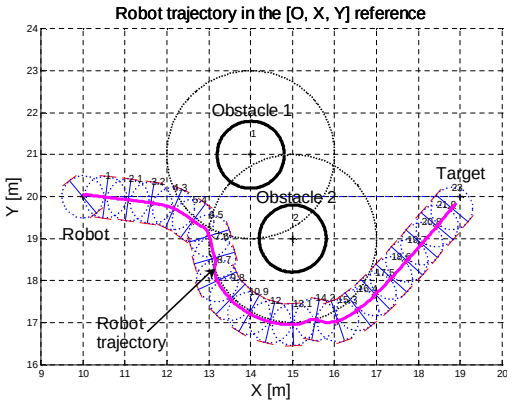
B. Rule 2 - How to avoid local minima and dead ends?

As given in Algorithm 2 (line 9 and 10) the direction of the limit-cycle can be compelled to avoid conflicting situations. This case is given for example when the robot must avoid two or more obstacles with an overlapped region. Figure 7(a) shows what happens to the robot when it do not follows this rule. In Figure 7(b) the robot continues to avoid the obstacle 2 in counter-clockwise according to the rule 2 instead of avoiding it in clockwise direction. Therefore, with this short memory information on the antecedent direction of the avoided obstacle, the robot can perform efficiently its navigation while avoiding this conflicting situation. In [24] authors use, in the same above situation, the definition of a virtual obstacle which contains all the overlapped obstacles, but this method need more time to achieve the obstacles skirting. This is due to the more important distance covered by the robot. In fact, we can easily suppose that when there are two or more overlapped obstacles that the new equivalent virtual obstacle will have a bigger radius than each individual obstacle and this radius will increase according to the furthest obstacles. To illustrate this case, let's take the specific example where a lot of overlapped obstacles are in a straight line. The equivalent virtual obstacle will be given by a very big circle which is not at all justifiable in that obstacles configuration. Moreover, the applied method given in [24] is, in our opinion, less reactive in the sense that it needs more information on the positions of all overlapped obstacles (even if the obstacle doesn't immediately disturb the navigation of the robot), whereas ours permits switching from one obstacle to another according to only reactive rules (cf. Section III-A).

Otherwise, figure 8 gives the robot trajectory when the obstacles are disposed as *U-shape* [28]. This obstacle configuration leads generally to dead end but it is not the case with algorithm 2.



(a) Without imposing the direction, the robot falls in a local minima



(b) While imposing the direction

Fig. 7. Influence of the rule 2

C. Rule 3 - How to avoid trajectory oscillations?

Figure 9 shows the efficiency of the proposed algorithm 2 to avoid the trajectory oscillations when the robot skirts the obstacle. Instruction codes 1 to 7 of Algorithm 2 permits to the robot to do not oscillate between the position where $D_{ROi} \leq R_{Ii}$ (activation of "obstacle avoidance" controller) and $D_{ROi} \geq R_{Ii}$ (activation of "attraction to the target" controller).

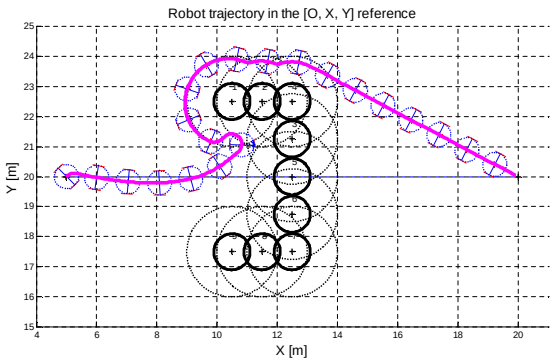
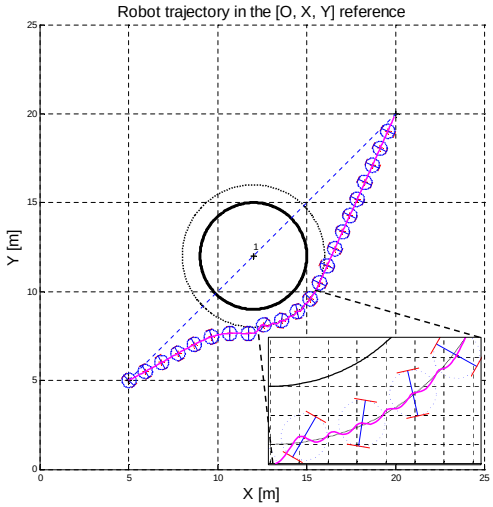
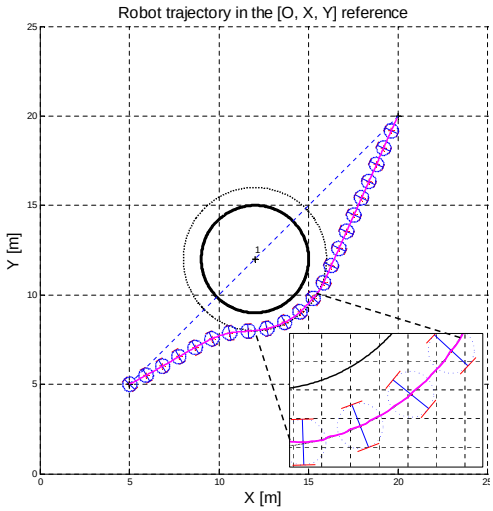


Fig. 8. Experimentation with U-shape obstacles



(a) With oscillations



(b) Without oscillations

Fig. 9. Avoidance of trajectory oscillations when Algorithm 2 is used

VI. Simulations results

Figure 10 shows the progress value of Lyapunov functions attributed to each controller $V_i|_{i=1..2}$ (cf. Figure 2) when the navigation is performed (cf. Figure 3(a)). These functions decrease asymptotically to the equilibrium point. The demonstration of the stability of the overall proposed structure of control is given in [15].

Otherwise, to demonstrate the efficiency of the proposed obstacle avoidance algorithm, a statistical survey was made while doing a large number of simulations in different cluttered and unstructured environments (cf. Figure 11(b)). We did 1000 simulations with every time 25 obstacles with different positions in the environment. All simulations permits to the robot to reach the target in finite time. These simulations prove also the gain in time given when the orbital method is applied (cf. Figure 11(b)) instead of the one which activates the obstacle avoidance controller only when the robot is inside of the circle of influence (cf. Figure 11(a)). For

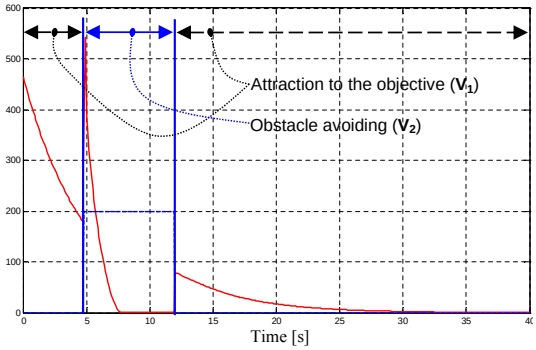
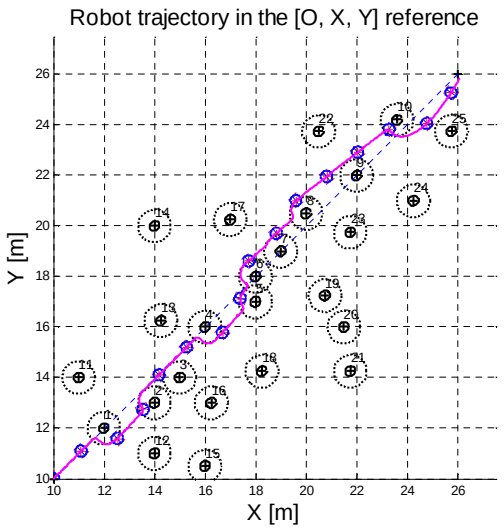
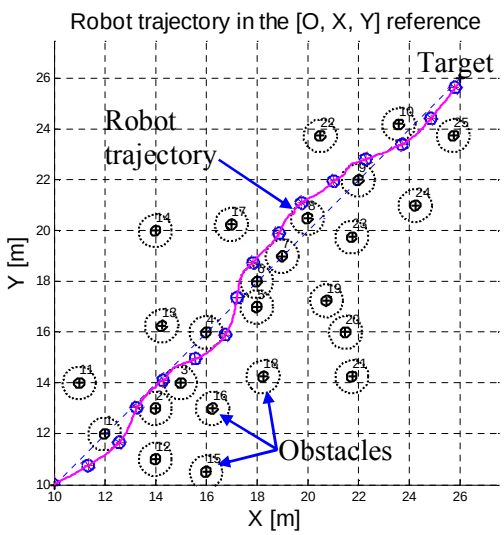


Fig. 10. Evolution of Lyapunov functions for the two used controllers during the robot navigation.

these two simulations (cf. Figure 11(a) and 11(b)) the gain in time is of 8% and the mean time of the 1000 simulations



(a) Without orbital algorithm



(b) With orbital algorithm

Fig. 11. Smooth trajectory obtained with the proposed orbital algorithm

gives an improvement of 6%. The trajectories given by the proposed algorithm are smoother (cf. Figure 11(b)) than those without (cf. Figure 11(a)).

VII. Conclusion and further work

In this paper, an obstacle avoidance algorithm based on orbital limit-cycle trajectories is proposed. This algorithm was embedded in an on-line behavioral control architecture and permits for a mobile robot to navigate in cluttered environments with safe and reliable way. In addition to the use of limit-cycles, the algorithm uses specific reactive rules which allows to the robot to avoid deadlocks, local minima and oscillations. These simple rules are efficient and permits to the proposed algorithm to do not becomes more and more complex. In other terms, the proposed control structure is open and flexible in the sense that it can manage a lot of other conflicts situations while only adding simple reactive rules. Otherwise, the stability proof of the overall control architecture is given. Statistical survey in different environments proves the efficiency and the flexibility of the control. The proposed algorithm allows also to reduce the time needed to reach the target. In fact, according to this algorithm, robot anticipates the collisions with obstacles according to smooth local trajectory modifications. Future work will first test the proposed control architecture on the CyCab vehicle [20]. The second step is to adapt the proposed structure of control to more complex tasks like the navigation in highly dynamical environment.

References

[1] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *The International Journal of Robotics Research*, vol. 5, pp. pp.90–99, Spring 1986.

[2] R. C. Arkin, "Motor schema-based mobile robot navigation," *International Journal of Robotics Research*, vol. 8, no. 4, pp. pp.92–112, 1989.

[3] A. Elnagar and A. Hussein, "Motion planning using maxwell's equations," in *IEEE/RSJ International Conference on Intelligent Robots and System, Lausanne, Switzerland*, 2002.

[4] A. Saffiotti, E. Ruspini, and K. Konolige, "Robust execution of robot plans using fuzzy logic," in *in Fuzzy Logic in Artificial Intelligence : IJCAI'93 Workshop*, Springer-Velag, Ed., Chambéry-France, 1993, pp. 24–37.

[5] O. Motlagh, T. S. Hong, and N. Ismail, "Development of a new minimum avoidance system for a behavior-based mobile robot," *Fuzzy Sets and Systems*, 2008.

[6] C. Ordonez, E. G. C. Jr., M. F. Selekwa, and D. D. Dunlap, "The virtual wall approach to limit cycle avoidance for unmanned ground vehicles," *Robotics and Autonomous Systems*, vol. 56, no. 8, pp. pp.645–657, 2008.

[7] L. P. Zapata R., Cacitti A., "Dvz-based collision avoidance control of non-holonomic mobile manipulators," *JESA, European Journal of Automated Systems*, vol. 38(5), pp. 559–588, 2004.

[8] J. Minguez, F. Lamiraux, and J.-P. Laumond, *Handbook of Robotics*, 2008, ch. Motion Planning and Obstacle Avoidance, pp. pp.827–852.

[9] J.-C. Latombe, *Robot Motion Planning*. Kluwer Academic Publishers, Boston, MA, 1991.

[10] E. Rimon and D. E. Koditschek, "Exact robot navigation using artficial potential flelds," *IEEE Transactions on Robotics and Automation*, vol. 8(5), pp. 501–518, Oct. 1992.

[11] C. Belta, V. Isler, and G. J. Pappas, "Discrete abstractions for robot motion planning and control in polygonal environments," *IEEE Transactions on Robotics*, vol. 21(5), pp. 864–874, Oct. 2005.

[12] D. C. Conner, H. Choset, and A. Rizzi, "Integrated planning and control for convex-bodied nonholonomic systems using local feedback," in *Proceedings of Robotics: Science and Systems II*. Philadelphia, PA: MIT Press, August 2006, pp. 57–64.

[13] M. Egerstedt and X. Hu, "A hybrid control approach to action coordination for mobile robots," *Automatica*, vol. 38(1), pp. 125–130, 2002.

[14] J. Toibero, R. Carelli, and B. Kuchen, "Switching control of mobile robots for autonomous navigation in unknown environments," in *IEEE International Conference on Robotics and Automation*, 2007, pp. 1974–1979.

[15] L. Adouane, "An adaptive multi-controller architecture for mobile robot navigation," in *10th IAS, Intelligent Autonomous Systems*, Baden-Baden, Germany, July 23-25 2008, pp. 342–347.

[16] L. Adouane and N. Le Fort-Piat, "Behavioral and distributed control architecture of control for minimalist mobile robots," *Journal Européen des Systèmes Automatisés*, vol. 40, no. 2, pp. pp.177–196, 2006.

[17] A. Stuart and A. Humphries, *Dynamical systems and numerical analysis*. Cambridge University Press, 1996.

[18] H. K. Khalil, *Frequency domain analysis of feedback systems*, P. Hall, Ed. Nonlinear Systems: Chapter7, 3 edition, 2002.

[19] M. Yerry and M. Shephard, "A modified quadtree approach to finite element mesh generation," *Computer, Graphics and Applications*, 1983.

[20] C. Pradalier, J. Hermosillo, C. Koike, C. Braillon, P. Bessière, and C. Laugier, "The cycab: a car-like robot navigating autonomously and safely among pedestrians," *Robotics and Autonomous Systems*, vol. 50, no. 1, pp. 51–68, 2005.

[21] W. H. Huang, B. R. Fajen, J. R. Fink, and W. H. Warren, "Visual navigation and obstacle avoidance using a steering potential function," *Robotics and Autonomous Systems*, vol. 54, no. 4, pp. 288–299, April 2006.

[22] H. Zhang, S. Liu, and S. X. Yang, "A hybrid robot navigation approach based on partial planning and emotion-based behavior coordination," in *International Conference Intelligent Robots and Systems*, Beijing, China, October 2006, pp. 1183–1188.

[23] R. A. Brooks, "A robust layered control system for a mobile robot," *IEEE Journal of Robotics and Automation*, vol. RA-2, pp. pp.14–23, March 1986.

[24] D.-H. Kim and J.-H. Kim, "A real-time limit-cycle navigation method for fast mobile robots and its application to robot soccer," *Robotics and Autonomous Systems*, vol. 42(1), pp. 17–30, 2003.

[25] L. Adouane, "Hybrid and safe control architecture for mobile robot navigation," in *9th Conference on Autonomous Robot Systems and Competitions*, Portugal, May 2009.

[26] J.-P. Laumond, *La robotique mobile*. Hermès, 2001.

[27] M. S. Jie, J. H. Baek, Y. S. Hong, and K. W. Lee, "Real time obstacle avoidance for mobile robot using limit-cycle and vector field method," *Knowledge-Based Intelligent Information and Engineering Systems*, pp. 866–873, October 2006.

[28] M. Wang and J. N. Liua, "Fuzzy logic-based real-time robot navigation in unknown environment with dead ends," *Robotics and Autonomous Systems*, vol. 56, no. 7, pp. pp.625–643, 2008.