Procedia Computer Science

Volume 51, 2015, Pages 423–432

ICCS 2015 International Conference On Computational Science

Adaptive Autonomous Navigation using Reactive Multi-agent System for Control Law Merging

Baudouin Dafflon[1], Jose Vilca[3], Franck Gechter[2], and Lounis Adouane[3]

[1] Université Lyon2 Lumière, DISP-LAB, 160 avenue de l'université, Bron, France,
name.surname@univ-lyon2.fr
[2] UTBM, IRTES-SET, 4 rue Mieg, Belfort, France.
name.surname@utbm.fr
[3] Institut Pascal, Blaise Pascal University, Clermont-Ferrand, France.
name.surname@univ-bpclermont.fr

Abstract

This paper deals with intelligent autonomous navigation of a vehicle in cluttered environment. We present a control architecture for safe and smooth navigation of a Unmanned Ground Vehicles (UGV). This control architecture is designed to allow the use of a single control law for different vehicle contexts (attraction to the target, obstacle avoidance, etc.) [4]. The reactive obstacle avoidance strategy is based on the limit-cycle approach [2]. To manage the interaction between the controllers according to the context, the multi-agent system is proposed. Multi-agent systems are an efficient approach for problem solving and decision making. They can be applied to a wide range of applications thanks to their intrinsic properties such as self-organization/emergent phenomena. Merging approach between control laws is based on their properties to adapt the control to the environment. Different simulations on cluttered environments show the performance and the efficiency of our proposal, to obtain fully reactive and safe control strategy, for the navigation of a UGV.

Keywords: Multi-agent systems, Autonomous vehicles, Hybrid architecture, Obstacle avoidance

1 Introduction

One of the main motivation to use Unmanned Ground Vehicles (UGVs) is the decrease of the traffic congestion of vehicles in urban areas, with correlated pollution, noise and time waste. So as to obtain such a transportation system, an efficient and reliable automatic navigation capability is required with the some criteria which must guarantee the safety and comfort of passengers [11].

This paper deals with the navigation of urban vehicle in cluttered environment. The navigation task consists in reaching a defined target while avoiding detected obstacles detected from real-time sensor measurements. To ensure the vehicles ability to accomplish a reactive navigation, it is proposed to explore behavioral control architectures originally proposed by Brooks [7]. This kind of control

Selection and peer-review under responsibility of the Scientific Programme Committee of ICCS 2015

doi:10.1016/j.procs.2015.05.262

architecture breaks the complexity of the overall task by dividing it into several basic tasks [1]. Each basic task is accomplished with its corresponding controller.

An important issue for successful autonomous navigation is the obstacle avoidance ability. This function permits to prevent robot collision, thus ensuring vehicle safety. Many reactive approaches can be found in literature, such as obstacle avoidance using vortex fields [12] and orbital trajectories [2]. This last approach is built on circular limit-cycle differential equations in [18, 16, 2]. Circular limit cycles are more stable than vortex fields and always converge to periodic orbits. This work uses elliptical trajectories that were presented in [3]. Therefore, more generic and efficient obstacle avoidance is performed, even with different obstacle shapes, for instance, long walls.

Most of the algorithms found in literature deal with only one fixed obstacle. However, they brings interesting properties such as reliability and continuity in control law. Dealing with several obstacles and with their potential moves is a much harder issue which requires an important embedded computational power so as to recompute at each time-step the correct trajectory under continuity constraint or a decrease in term of performances and precision. The goal of this paper is to propose a method using a reactive multi-agent system to merge command laws of the control architecture (attraction to target controller and the obstacle avoidance controllers).

Multi-agent systems are an efficient approach for problem solving and decision making. They can be applied to a wide range of applications thanks to their intrinsic properties and features such as simplicity, flexibility, reliability, self-organization/emergent phenomena, low cost agent design and adaptation capacity. It has been shown that reactive multi-agent system are efficient to tackle complex problems [14], cooperation of situated agents/robots [15], data fusion and problem/game-solving [13]. In this context, our proposal consists in decision making by evaluating emergent properties of agent's organization.

The paper is structured as follow: in the next section, the control architecture for the navigation of a UGV is introduced. The model of the UGV and its controllers are also detailed. The multi-agent system and its proprieties applied to autonomous navigation are described in Section 3. Simulations showing the efficiency of our proposal are detailed in Section 4. Finally, conclusion and future works are given in Section 5.

2 Control architecture

The control architecture for a safe and smooth autonomous navigation of UGV is shown in Fig. 1. It is designed for a UGV modelled as a tricycle robot. This architecture aims to manage the interactions among elementary controllers while guaranteeing the stability of the overall control [6, 5]. The global navigation framework is operated by the *Hierarchical action selection* block that selects the elementary controller (*Target reaching* or *Obstacle avoidance*) according to the context of the environment. Each elementary controller (cf. Fig. 1) provides as output ($\mathbf{O}_{AT}$ or $\mathbf{O}_{OA}$) a Control Input $\mathbf{I}_{SP}$ to the *Control law* block.

In this work, a single control law for the UGV (tricycle robot) is used [20]. It considers the vehicle postures and velocities. This control law allows the UGV to reach a static or dynamic target with a desired orientation and velocity (cf. subsection 2.2.3). The inputs of the control law (posture errors between the vehicle and its assigned target) are provided by the elementary controllers (cf. subsection 2.2). The control law is synthesized according to Lyapunov theorem (more details are given in [20]). The main blocks of the architecture are detailed below.

The *Sensor Information* block incorporates the propriocetive and exteroceptive sensors such as range sensor, cameras, odometers and RTK-GPS. Its goal is to capture information related to the robot environment, mainly potential obstacles [8, 10]. In the sequel, we assume that the UGV has a RTK-GPS and a LIDAR range sensor.

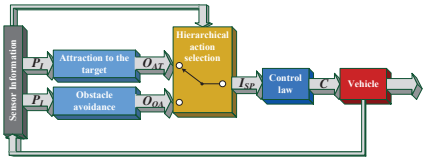

Figure 1: Control architecture embedded in the UGV for autonomous navigation [20].

The control architecture uses a *Hierarchical action selection* mechanism to manage the switches between the two elementary controllers (Behavior-based approach), *Target reaching* and *Obstacle avoidance* blocks, according to the formation parameters and environment perception. The hierarchical action selection mechanism activates the *Obstacle avoidance* block as soon as it detects at least one obstacle which can hinder the future vehicle movement toward its dynamic virtual target (more details are given in [3]). It allows to anticipate the activation of obstacle avoidance controller and to decrease the time to reach the assigned target (static or dynamic). In order to provide the enough overall details of the presented control architecture, the following subsections present the UGV model and the elementary controllers.

2.1 Vehicle modeling

We assume that the UGV evolves in asphalt road and in cluttered urban environment with relatively low speed (less than $v_{max} = 2\ m/s$). Hence, the use of kinematic model (which relies on pure rolling without slipping) of the UGV is sufficient. The kinematic model of the UGV is based on the well-known tricycle model [19]. The two front wheels are replaced by a single virtual wheel located at the center between the front wheels. The equations of UGV model can be written as (cf. Fig. 2):

$$\begin{cases} \dot{x} &= v\cos(\theta) \\ \dot{y} &= v\sin(\theta) \\ \dot{\theta} &= v\tan(\gamma)/l_b \end{cases} \tag{1}$$

where (x, y, θ) is the UGV posture in the global reference frame $X_G Y_G$. v and γ are respectively the linear velocity and the orientation of the vehicle front wheel. l_b is the wheelbase of the vehicle.

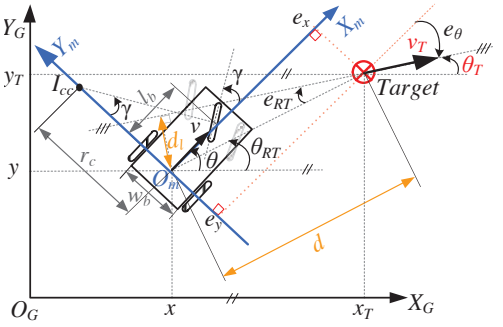

Figure 2: UGV and target configuration variables in Cartesian reference frames (local and global).

2.2 Elementary controllers

Each elementary controller generates the control inputs $\mathbf{I}_{SP}$ (posture errors (e_x, e_y, e_θ) and velocities v_T) of the *Control law* block (cf. Fig. 1).

2.2.1 Target reaching controller

The target set-point modeling is defined as a point with non-holonomic constraints (cf. Fig. 2). For static target reaching (*point stabilization*, i.e., to reach a specific point with a given orientation), v_T is not necessarily equal to zero; v_T is then considered as a desired velocity value for the vehicle when it reaches the desired target posture (x_T, y_T, θ_T).

Before to remind the used control law [20], let us describe the following notations (cf. Fig. 2):

- I_{cc} is the instantaneous center of curvature of the vehicle trajectory, $r_c = l_b/\tan(\gamma)$ is the radius of curvature and $c_c = 1/r_c$ is the curvature.
- (e_x, e_y, e_θ) are the errors w.r.t local frame $(X_m Y_m)$ between the vehicle and the target postures.
- θ_{RT} and d are respectively the angle and distance between the target and vehicle positions.
- e_{RT} is the error related to the vehicle position (x, y) w.r.t the target orientation.

This controller guides the vehicle towards the static target. It is based on the posture control of the UGV w.r.t. the target (represented by errors variables (e_x, e_y, e_θ) in Fig. 2). These errors are computed w.r.t. the local reference frame $X_m Y_m$ and they are given by:

$$\begin{cases} e_x = & \cos(\theta)(x_T - x) + \sin(\theta)(y_T - y) \\ e_y = & -\sin(\theta)(x_T - x) + \cos(\theta)(y_T - y) \\ e_\theta = & \theta_T - \theta \end{cases} \tag{2}$$

The error function e_{RT} is added to the canonical error system (2) (cf. Fig. 2). Let us now write d and θ_{RT} as (cf. Fig. 2):

$$d = \sqrt{(x_T - x)^2 + (y_T - y)^2} \tag{3}$$

$$\begin{cases} \theta_{RT} = & \arctan\left((y_T - y)/(x_T - x)\right) & \text{if } d > \xi \\ \theta_{RT} = & \theta_T & \text{if } d \leq \xi \end{cases} \tag{4}$$

where ξ is a small positive value ($\xi \approx 0$). The error e_{RT} is defined as (cf. Fig. 2):

$$e_{RT} = \theta_T - \theta_{RT} \tag{5}$$

Furthermore, the velocity set-point v_T of the static target is defined by the designer according to the task. Finally the posture errors and velocities $(e_x, e_y, e_\theta, v_T)$ are the input of the *Control law* block (cf. subsection 2.2.3).

2.2.2 Obstacle avoidance controller

Different methods can be found in the literature for obstacle avoidance [17, 21]. One of them is the limit-cycle method, the UGV avoids reactively the obstacle if it tracks accurately limit-cycle trajectories as detailed in [3]. The main ideas behind this controller are briefly detailed below:

The differential equations of the elliptic limit-cycles are:

$$\dot{x}_s = \quad m(By_s + 0.5Cx_s) + x_s(1 - Ax_s^2 - By_s^2 - Cx_s y_s) \tag{6}$$
$$\dot{y}_s = \quad -m(Ax_s + 0.5Cy_s) + y_s(1 - Ax_s^2 - By_s^2 - Cx_s y_s) \tag{7}$$

with $m = \pm 1$ according to the avoidance direction (clockwise or counter-clockwise). (x_s, y_s) corresponds to the position of the UGV according to the center of the ellipse. The variables A, B and C are given by:

$$A = (\sin(\Omega)/b_{lc})^2 + (\cos(\Omega)/a_{lc})^2 \tag{8}$$
$$B = (\cos(\Omega)/b_{lc})^2 + (\sin(\Omega)/a_{lc})^2 \tag{9}$$
$$C = (1/a_{lc}^2 - 1/b_{lc}^2)\sin(2\Omega) \tag{10}$$

where a_{lc} and b_{lc} characterize respectively the major and minor elliptic semi-axes and Ω gives the ellipse orientation.

In our case, the controller can be written as an orientation control. We consider thus $e_x = 0$ and $e_y = 0$ in (2) (cf. Fig. 2), i.e, the vehicle position is at each sample time in the desired position. The limit-cycle propriety allows to avoid the obstacles. The desired vehicle orientation is given by the differential equation of the limit-cycle (6) and (7):

$$\theta_d = \arctan(\dot{y}_s/\dot{x}_s) \tag{11}$$

Furthermore, the linear velocity of the UGV is decreased for safe avoidance when the obstacle avoidance controller is activated, e.g, $v_T = v_{max}/2$.

2.2.3 Control law

The used control law is designed according to Lyapunov stability analysis [20]. The desired vehicle linear velocity v and its front wheel orientation γ that make the errors (e_x, e_y, e_θ) converge always to zero can be chosen as:

$$v = v_T \cos(e_\theta) + v_b \tag{12}$$
$$\gamma = \arctan(l_b \left[r_{c_T}^{-1} \cos^{-1}(e_\theta) + c_c\right]) \tag{13}$$

where c_c is given by:

$$v_b = K_x (K_d e_x + K_l d \sin(e_{RT}) \sin(e_\theta) + K_o \sin(e_\theta) c_c) \tag{14}$$
$$c_c = \frac{d^2 K_l \sin(e_{RT}) \cos(e_{RT})}{r_{c_T} K_o \sin(e_\theta) \cos(e_\theta)} + \frac{K_{RT} \sin^2(e_{RT})}{\sin(e_\theta)\cos(e_\theta)} + \frac{K_d e_y - K_l d \sin(e_{RT}) \cos(e_\theta)}{K_o \cos(e_\theta)} + K_\theta \tan(e_\theta) \tag{15}$$

$\mathbf{K} = (K_d, K_l, K_o, K_x, K_{RT}, K_\theta)$ is a vector of positive constants which must be defined by the designer according to the desired convergence toward the assigned target (more details are given in [20]).

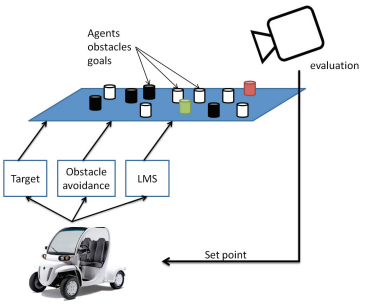

Figure 3: Global architecture

3 Hierchical action selection using a reactive multi-agent system

3.1 Global overview

Different solutions can be chosen to combine the elementary controllers. The easiest one is to use a hard switch which selects the application of the one controller according to the distance to the nearest obstacle. This solution provides good results in term of reliability and command law continuity. Nevertheless, it does not ensure adaptive behaviour and a good level of comfort for passengers.

The aim of this section is to introduce an adaptive system to combine *target reaching* and *obstacles avoidance* behaviours depending on the distribution of the obstacles perceived by the vehicle. The proposed switching system between controllers is based on a multi-agent system where the interpretation of agency on the organizational level provides a merged command. The proposed approach is an extension of the model presented in [9]. It is based on a situated virtual environment where data, provided by sensors, agents and target position are interacting together. The interactions between system entities are based on Newtonian laws.

The merging process is made as follow (Figure 3):

- The agents virtual environment is built using the perception of obstacles that have to be avoided and integrates the position of the target relatively to current vehicle position. The dynamics of this environment is linked to the changes that may occur in vehicle perception.
- Agents, considered as elementary particles, interact with vehicle target position and obstacles representatives.
- Agents organisation is then measured by an external observer taking into account geometrical and dynamical aspects. This measure is then translated into a merge command law that takes into account both navigation goal and obstacle distribution.

This system is detailed in the following subsections.

3.2 Model description

3.2.1 Agents Environment

The agents environment is the key component in such approaches. Its role is to link vehicles world (real or simulated) with the agent world. It integrates both the data provided by the two controllers presented above and the information furnished by the sensors. Basically, the data provided by the controllers are

combined so as to produce an attractive spot for agents. The combination is made taking into account the distance of the nearest obstacle using the following equation:

$$T(\vec{x},y) = f(d)\vec{OA} + (1-f(d))\vec{F} \tag{16}$$

where T: target position in vehicle referential, OA : obstacles avoidance command, F: path following command

The function f is define as follow: $f(x) = (1+0.5.x^2)^{-1}$ with x the distance to the nearest obstacle. The value of $f(x)$ is high when the obstacle is close to the vehicle, consequently more importance is paid on obstacle avoidance.

As opposed to the treatment applied to the command laws, the positions of perceived obstacles are directly integrated into the virtual environment as aggregates of repulsive spots.

3.2.2 Agents and Interactions

Agents are considered as small mass particles evolving in a force field. The force field is obtained combining repulsion forces induced by obstacles representatives and by agents themselves and attraction forces induced by the target to reach. Agents environment perception is realized through a circular frustum. A projection of the target position is made so as to be able keep it on frustum borders when out of range This projection allows agents to always know the target direction in the environment.

As explained before, interactions as composed of two categories: attractions and repulsions.

- Interaction between agents and target (Attraction): The attraction force generated by the target is computed as a linear force defined by:

$$\vec{F} = \beta_g m \vec{A_i T} \tag{17}$$

- Interaction between agents (Repulsion): The repulsion between agents is generally introduced to ensure a homogeneous exploration of the environment avoiding false agents grouping. This repulsion is made by classical Newtonian force in r^{-2}. If A_i and A_j are two agents located in P_i and P_j, the repulsion force is given by:

$$\vec{Fr}_{ij} = \alpha m_i m_j \frac{\vec{P_i.P_j}}{\|\vec{P_i.P_j}\|^3} \tag{18}$$

- Interaction between agents and obstacles (Repulsion): This interaction shares the same formulation as agents repulsion. This force could be generalized in a 2-dimensional space by:

$$\begin{cases} Fo_i^X = \sum_o \left(\Delta_o \cdot m \cdot m_o \frac{(x_i - x_o)}{((y_i-y_o)^2+(x_i-x_o)^2)^{3/2}} \right) \\ Fo_i^Y = \sum_o \left(\Delta_o \cdot m \cdot m_o \frac{(y_i - y_o)}{((y_i-y_o)^2+(x_i-x_o)^2)^{3/2}} \right) \end{cases} \tag{19}$$

3.2.3 Agent population evaluation

The decision process is based on the evaluation of the distribution of agents. The system evaluates the repartition of agents population and defines a center of mass. The result of this observation is noted $\overrightarrow{P_{dir}(x,y)}$ and corresponds to a vector from vehicle and center of mass of agent's population $P_{mean}(x,y)$

$\overrightarrow{P_{dir}(x,y)}$ is a new command to control the vehicle. The angle between $\overrightarrow{P_{dir}(x,y)}$ and local X axis is a steering angle adopted by the vehicle. The length of $\overrightarrow{P_{dir}(x,y)}$ corresponds to the speed command reference.

4 Validation

In many cases, when a simulation is done, one have to make a conclusion from what have happened. In this context, metrics have been defined to record some parameters during simulation time. They are useful to exploit post-simulation results.

- Steering angle metrics: This metrics records the steering angle command of the vehicle.
- Speed metric : This metric records the speed command of the vehicle.
- Physical integrity metric : Physical integrity is the closest distance to the obstacles. Time to collision is also evaluated using the current velocity vector.

As previously said, this paper presents a hybrid control architecture and multi-agent system. To illustrate our approach, simulations were made in Matlab® and in Janus[1] for the multi-agent part. The scenario is divided into two parts. In first, hard switch between obstacle avoidance and target reaching controllers is used. In second, multi-agent system adapt command taking account data environment. Scene is composed by four obstacles placed on vehicle path and simulation is stopped at the first collision.

4.1 Simulation results

Simulation were made with one single obstacle. The result obtained were similar whether the hard switch or the multi-agent were used. Consequently, we decided to focus on the results obtained with several obstacles where the results obtained are more interesting in term of command continuity. This simulation allows to increase the complexity of the environment. The vehicle navigates while avoiding the obstacles (clockwise and counter-clockwise sense).

- **Hard switch.**
  Figure 4 shows the trajectory of the vehicle using a hard switch between the controllers. The switches occur when the hinder obstacles are detected (e.g., obstacles 3 and 4). Figure 5 (top left) shows the evolution of the velocity and steering angle commands using the hard switch between the controllers. We can observe the discontinuity in the commands which can damage the actuators and can be uncomfortable for the passengers.

  Figure 5 (top right) shows the distance to the four obstacles using the hard switch between the controllers. We can observe that the vehicle navigation is safe.

- **Multi-agents switch.**
  Figure 4 (bottom left) shows the trajectory of the vehicle using a multi-agents switch. The gradual coverage of obstacle avoidance law allows a smoother path

  Figure 5 (Bottom right) shows the evolution of velocity and steering angle during the simulation. We can see that the maximum speed is more important and that the steering angles are smoother when multi-agents merge is used.

  We can summarize these simulations: the soft switch allows to move nearer obstacles has a greater speed while ensuring better security.

[1] http://www.janus-project.org/

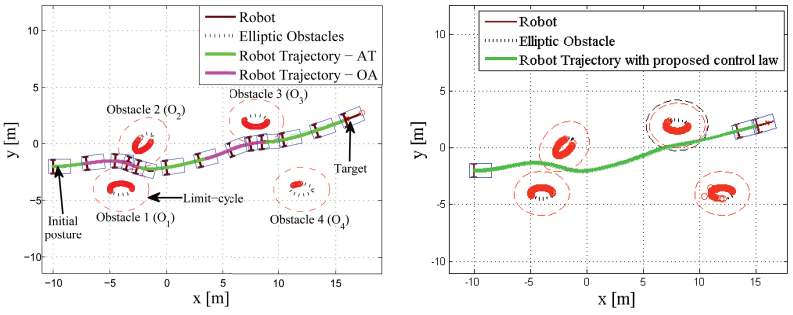

Figure 4: Trajectory of the vehicle: Hard switch (Left) and Multi-agent switch (right)

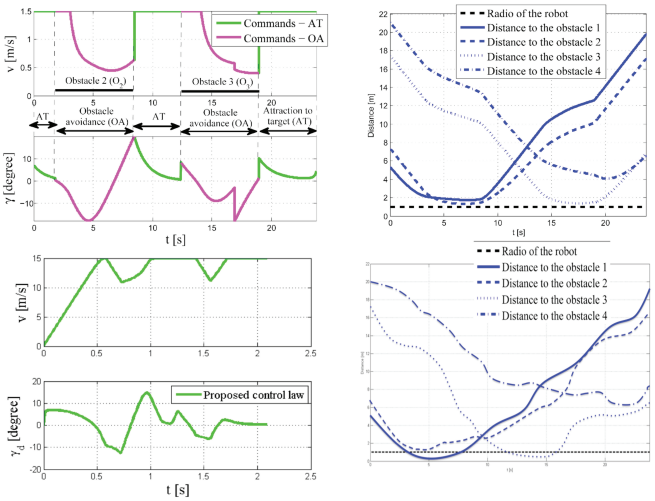

Figure 5: **Top left**: Commands of vehicle (Hard switch), **Top right**: Distance to the obstacle (Hard switch), **Bottom left**: Commands of vehicle (Multi-agents), **Bottom right**: Distance to the obstacle (multi-agents)

5 Conclusions

In this paper, an approach of global control architecture was presented. Originally based on a hard switch, we propose a merging controls laws to adapt vehicle behaviour to its environment. To cope with the navigation of UGV in cluttered environment, a single control law is embedded in the UGV allowing the simplification of the used control architecture for autonomous navigation. The obstacle avoidance based on the limit-cycle guarantees the safe navigation in cluttered environment. Merging system proposes an adaptive hierarchical action selection using reactive multi-agent system for control laws merging. Merge system is based on reactive multi-agent system. Environment data are provided by vehicle sensors and used to anticipate obstacles. Multi-agent system could be see as a security system when an obstacle avoidance controller failure. This method was successfully tested in simulation and results obtained encourage us to test it using actual laboratory vehicles.

Those works are done with the support of the French ANR (National Research Agency) through the ANR-VTT SafePlatoon 6 project (ANR-10-VPTT-011).

References

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

[2] L. Adouane. Orbital obstacle avoidance algorithm for reliable and on-line mobile robot navigation. In *9th Conference on Autonomous Robot Systems and Competitions*, Portugal, May 2009.

[3] L. Adouane, A. Benzerrouk, and P. Martinet. Mobile robot navigation in cluttered environment using reactive elliptic trajectories. In *18th IFAC World Congress*, August 2011.

[4] Lounis Adouane. An adaptive multi-controller architecture for mobile robot navigation. In *10th IAS, Intelligent Autonomous Systems*, pages 342–347, Baden-Baden, Germany, July 23-25 2008.

[5] A. Benzerrouk, L. Adouane, and P. Martinet. Altruistic distributed target allocation for stable navigation in formation of multi-robot system. In *10th International IFAC Symposium on Robot Control (SYROCO'12)*, Dubrovnik - Croatia, 5-7, September 2012.

[6] Ahmed Benzerrouk, Lounis Adouane, and Philippe Martinet. Lyapunov global stability for a reactive mobile robot navigation in presence of obstacles. In *ICRA'10 International Workshop on Robotics and Intelligent Transportation System, RITS10*, Anchorage-Alaska, May 7th 2010.

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

[8] J. Clark and R. Fierro. Mobile robotic sensors for perimeter detection and tracking. {*ISA*} *Transactions*, 46(1):3 – 13, 2007.

[9] B. Dafflon, F. Gechter, P. Gruer, and A. Koukam. Vehicle platoon and obstacle avoidance: a reactive agent approach. *IET Intelligent Transport Systems*, (3):257–264, 2013.

[10] A. Das, R. Fierro, V. Kumar, J. Ostrowski, J. Spletzer, and C. Taylor. A vision-based formation control framework. *IEEE Transaction on Robotics and Automation*, 18(5):813–825, 2002.

[11] P. Daviet and M. Parent. Platooning for small public urban vehicles. In Oussama Khatib and J.Kenneth Salisbury, editors, *Experimental Robotics IV*, volume 223 of *Lecture Notes in Control and Information Sciences*, pages 343–354. Springer Berlin, 1997.

[12] A. De Luca and G. Oriolo. Local incremental planning for nonholonomic mobile robots. In *IEEE International Conference on Robotics and Automation*, volume 1, May 1994.

[13] G. DiMarzo-Serugendo, A. Karageorgos, O.F. Rana, and F. Zambonelli. Engineering self-organising systems: Nature-inspired approaches to software engineering. *Lecture notes in Atificial intelligence*, 2977:299, 2004.

[14] M. El-Zaher, F. Gechter, P. Gruer, and M. Hajjar. A new linear platoon model based on reactive multi-agent systems. *23rd IEEE International Conference on Tools with Artificial Intelligence ICTAI*, 2011.

[15] Jean-Pierre Georgé, Marie-Pierre Gleizes, FranciscoJ. Garijo, Victor Noël, and Jean-Paul Arcangeli. Self-adaptive coordination for robot teams accomplishing critical activities. In *Advances in Practical Applications of Agents and Multiagent Systems*, volume 70, pages 145–150. Springer, 2010.

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

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

[18] 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*, 42(1):17–30, 2003.

[19] A. De Luca, G. Oriolo, and C. Samson. Feedback control of a nonholonomic car-like robot. In J.-P. Laumond, editor, *Robot Motion Planning and Control*, pages 171–253. Springer-Verlag, 1998.

[20] J. Vilca, L. Adouane, Y Mezouar, and P. Lébraly. An overall control strategy based on target reaching for the navigation of an urban electric vehicle. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'13)*, Tokyo - Japan, November 2013.

[21] R. Zapata, A. Cacitti, and P. Lepinay. Dvz-based collision avoidance control of non-holonomic mobile manipulators. *JESA, European Journal of Automated Systems*, 38(5):559–588, 2004.