This article can be cited as A. Soriano, E. J. Bernabeu, A. Valera and M. Valles, Distributed Collision Avoidance Method Based on Consensus among Mobile Robotic Agents, International Journal of Imaging and Robotics, vol. 15, no. 1, pp. 80-90, 2015.

Distributed Collision Avoidance Method Based on Consensus among Mobile Robotic Agents

Ángel Soriano, Enrique J. Bernabeu, Ángel Valera and **Marina Vallés**

Instituto U. de Automática e Informática Industrial
Universitat Politècnica de Valencia, Camino de Vera s/n
46022 Valencia, Spain;
Email: {ansovi, ebernabe, giuprog, mvalles}@ai2.upv.es

ABSTRACT

This paper presents a new methodical approach to the problem of collision avoidance of mobile robots taking advantages of multi-agents systems to deliver solutions that benefit the whole system. The proposed method has the next phases: collision detection, obstacle identification, negotiation and collision avoidance. In addition of simulations with virtual robots in a 2D and 3D space, an implementation with real mobile robots has been developed in order to validate the proposed algorithm. The robots are based on Lego NXT, and they are equipped with a ring of proximity sensors for the collisions detections. The platform for the implementation and management of the multi-agent system is JADE.

Keywords: Avoiding collision method, robotic agents, mobile robots, avoidance collision method, jade platform.

Mathematics Subject Classification: 93C83, 93C95

Computing Classification System: J.7

1. INTRODUCTION

The area of artificial intelligence (AI) has expanded considerably in recent years. It not only dominates the area of games versus computers, but nowadays it applies in many sectors like databases management or web pages. As it is well known, the main topic of AI is the concept of intelligent agent defined as an autonomous entity which observes through sensors and acts upon an environment using actuators (Russell et al., 2003). This definition is very close to services that a robot can provide, so the concept of agent often is related with robots, (Bruce et al., 1997), (Van Leeuwen, 1995), (Michalewics, 1996).

On the other hand, detecting and avoiding a collision is a previous step for overcoming the motion planning problem (Cherubini et al., 2013), (Swingler et al., 2013). In fact, collision detection has been inherently connected with the motion-planning algorithms from the very beginning (Purcaru et al., 2013). Current planning algorithms require the collision detection of mobile and nondeterministic obstacles (Saudi et al., 2013).

This article can be cited as A. Soriano, E. J. Bernabeu, A. Valera and M. Valles, Distributed Collision Avoidance Method Based on Consensus among Mobile Robotic Agents, International Journal of Imaging and Robotics, vol. 15, no. 1, pp. 80-90, 2015.

Continuous collision detection (CCD) techniques are the most effective when dealing with multiple mobile agents or robots. CCD algorithms basically make a return if a collision between the motion of two given objects is presented or not; and if a collision is going to occur then, the instant in time of the first contact is returned, (Bernabeu, 2009), (Bernabeu et al., 2001), (Choi et al., 2006), (Redon et al., 2002), and (Van den Bergen, 2005).

In this paper, collision detection strategies of autonomous mobile robots based on (Bernabeu et al., 2001) are combined with strategies based on artificial intelligence to offer a new method of collision avoiding management.

The method is divided into three basic concepts which are merged in this paper: obstacle detection by a mobile robot, the concept of abstraction robotic agent as a software agent within MAS, and distributed artificial intelligence as a method of communication and negotiation between these software agents.

Nowadays, there are many sensors on the market that allow robots to know if there is an obstacle that stands between them and its trajectory, and where is that obstacle. This process is usually local, i.e. it is performed inside a robot. In the case of two mobile robots at the same scenario, each one represents an obstacle to the other, but neither is aware of it because it is handled as a local process. The concept of robotic agent in a multi-agent robotic system is proposed as a next level or upper layer to fix it and to manage a more intelligent solution.

Multi-agent robot systems (MARS) represent a complex distributed system, consisting of a large number of agents-robots cooperating for solving a common task. In this case, each agent of MARS represents a real physical mobile robot that informs its software agent of all it perceives. The ability of communication, cooperation and coordination between the agents, allows conversations, negotiations and agreements, which are the basis of the algorithm is presented.

A preliminary version of this paper is appeared in (Soriano et al., 2013). The main difference of this paper with respect to (Soriano et al., 2013) is an improvement over the simulation and test of the correct functioning of the method, which will be used to extend it in the future.

2. AVOIDING COLLISION METHOD

The aim of this section is reviewing a CCD methodology in (Bernabeu et al., 2001) for obtaining the instant in time when two robots or agents in motion will be located at their maximum-approach positions while they are following straight-line trajectories.

The mentioned maximum approach is also calculated. Therefore, if the involved robots do not collide while they are following their respective motions, then their minimum separation is returned. Otherwise, their maximum penetration is computed as a minimum translational distance (Cameron et al., 1986). A remarkable aspect is that both the instant in time and the corresponding minimum separation or maximum approach are computed without stepping any involved trajectory.

Some collision avoiding configurations for the involved robots or agents are directly generated from the computed instant in time and maximum penetration. These collision-free configurations are determined in accordance with a given coordination between the robots or agents.

2.1. Obtaining the Instant in Time and the Maximum Approach

Consider two robots or agents in motion each one enveloped or modelled by a circle. Let A be a circle in motion whose start position at time t_s is $A(ts)=(cA(ts),rA)$. Where $cA(ts)\in\Re^2$ is the A's centre at t_s and $r_A\in\Re$ is its radius. A is following a straight-line trajectory whose final position at t_g is given by $A(t_g)=(c_A(t_g),r_A)$. Let $v_A\in\Re^2$ be the A's velocity for the time span $[t_s,t_g]$.

Let B be a second circle in motion whose start and goal positions at the respective instants in time t_s and t_g are $B(t_s)=(c_B(t_s),r_B)$ and $B(t_g)=(c_B(t_g),r_B)$. The B's velocity for the time span $[t_s,t_g]$ is $v_B\in\Re^2$.

All the infinite intermediate positions of the mobile circle A for $t\in[t_s,t_g]$ while A is in motion is parameterized by λ with $\lambda\in[0,1]$, as follows:

$$A(\lambda)=\left\{(c_A(\lambda),r_A):\, c_A(\lambda)=c_A(t_s)+\lambda\cdot(c_A(t_g)-c_A(t_s))\;;\; t=t_s+\lambda(t_g-t_s)\;;\; \forall\lambda\in[0,1]\right\} \quad (1)$$

Note that the positions $A(\lambda)$ and $A(t)$, with $t=t_s+\lambda(t_g-t_s)$, are equal for all $t\in[t_s,t_g]$ and $\lambda\in[0,1]$. All the infinite intermediate positions of the mobile circle B are analogously parameterized for $\lambda\in[0,1]$ as indicated in (1).

Observing equation (1) is easy to conclude that the maximum approach d_M between in-motion circles A and B will be obtained by finding the parameter $\lambda_c\in[0,1]$ that minimizes $\|c_A(\lambda)-c_B(\lambda)\|-(r_A+r_B)$

Let $\lambda_c\in[0,1]$ be the parameter that minimizes (2). λ_c is obtained by minimizing by computing the distance from the origin point O to the straight-line $c_A(\lambda)-c_B(\lambda)$.

Note that $c_A(\lambda)-c_B(\lambda)$ for all $\lambda\in[0,1]$ is really a segment whose extreme points are respectively $c_0=c_A(t_s)-c_B(t_s)$ and $c_1=c_A(t_g)-c_B(t_g)$. Then, the parameter $\lambda_c\in[0,1]$ is obtained by projecting O onto mentioned segment, $O^{\perp}$, as

$$\lambda_c=-\frac{c_0\cdot(c_1-c_0)}{\|c_1-c_0\|^2} \quad \text{with } \lambda_c\in[0,1]. \text{ Then } O^{\perp}=c_0+\lambda_c\cdot(c_1-c_0). \quad (2)$$

Once λ_c is obtained, d_M and the associated instant in time t_M are computed as

$$d_M=\|c_A(\lambda_c)-c_B(\lambda_c)\|-(r_A+r_B) \qquad t_M=t_S+\lambda_c(t_g-t_S). \quad (3)$$

If d_M is negative, then d_M holds a penetration distance and, then A and B will collide with the maximum penetration d_M at t_M.

This article can be cited as A. Soriano, E. J. Bernabeu, A. Valera and M. Valles, Distributed Collision Avoidance Method Based on Consensus among Mobile Robotic Agents, International Journal of Imaging and Robotics, vol. 15, no. 1, pp. 80-90, 2015.

2.2. Determining Avoiding Collision Configurations

In case of collision, the positions where *A* and *B* present their maximum penetration are, as mentioned, $c_A(\lambda_c)$ and $c_B(\lambda_c)$ respectively with $\lambda_c \in [0,1]$, $d_M<0$, $t_M \in [t_s,t_g]$. One of these positions can be minimally translated in order to bring both circles into contact by using the unit vector $\hat{v}_{MTD}=O^{\perp}/\|O^{\perp}\|$, with $\|\hat{v}_{MTD}\|=1$,

Let $A_f(t_M)$ and $B_f(t_M)$ be the mention collision-free configurations,

$$\begin{aligned} A_f(t_M) &= \left(c_{Af}(t_M), r_A\right): \; c_{Af}(t_M) = c_A(t_M) - \delta \cdot \alpha \cdot d_M \cdot \hat{v}_{MTD} \\ B_f(t_M) &= \left(c_{Bf}(t_M), r_B\right): \; c_{Bf}(t_M) = c_B(t_M) + \delta \cdot (1-\alpha) \cdot d_M \cdot \hat{v}_{MTD} \; . \end{aligned} \tag{4}$$

where $c_A(t_M)=c_A(t_s)+\lambda_c(c_A(t_g)-c_A(t_s))$ and $c_B(t_M)=c_B(t_s)+\lambda_c(c_B(t_g)-c_B(t_s))$. Parameter $\delta \geq 1$ is a safety threshold. If $\delta=1$, then configurations $c_{Af}(t_M)$ *and* $c_{Bf}(t_M)$ will be in contact. Finally, parameter $\alpha \in [0,1]$ configures the degree of motion modification applied to each mobile robot or agent. In this way, if $\alpha=1$, then $c_B(t_M)$ *and* $c_{Bf}(t_M)$ are equal and, consequently, mobile robot or agent *B* do not change its current motion. A graphical example is shown in Fig. 1.

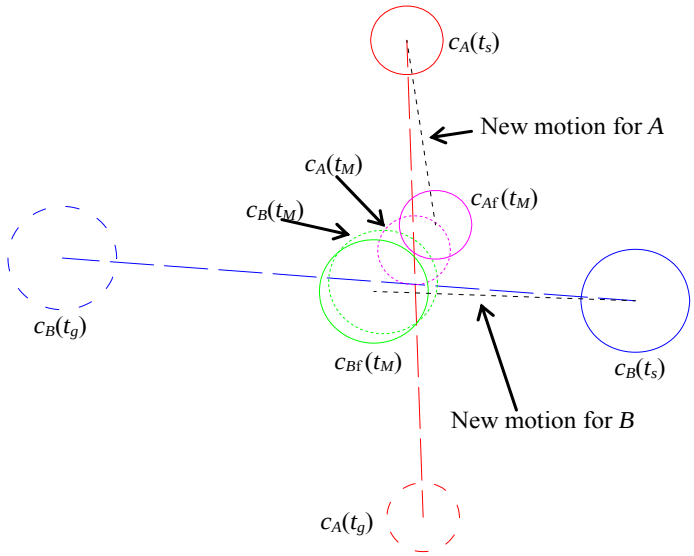


Figure 1. Avoiding collision configurations with $\alpha=0.7$ and $\delta=1.03$.

3. HYBRID CONTROL COLLISION AVOIDANCE

The implementation of the collision avoidance proposed methodology has six phases (see Fig. 2). A scenario where multiple robots follow a path infinite straight line between two target points is considered. These two points are alternated when they are achieved. All robots have their representation as a software agent in the MAS which encompasses the whole system, so there is no moving object within the scene that is not a software agent.

In phase 1 (**detection**), the local system (each robot) has defined a detection object area. If the local system detects an obstacle that may be a threat of collision (from now threat-object), it calculates the position of threat-object in the global scenario and it sends to the agent who represents the local system in MAS to manage it.

When an agent receives the position of a threat-object (from now threat-position) by the local system, it must identify what kind of threat it is. To know this, in the **obstacle identification** phase, the agent detects the threat (from now detector-agent), consults the other agents to know who is located within that area of threat. If there is not any agent within that area it is identified as a static object threat and directly the collision detection phase is performed. Otherwise, the threat-agent is identified through communication among agents and the next phase starts.

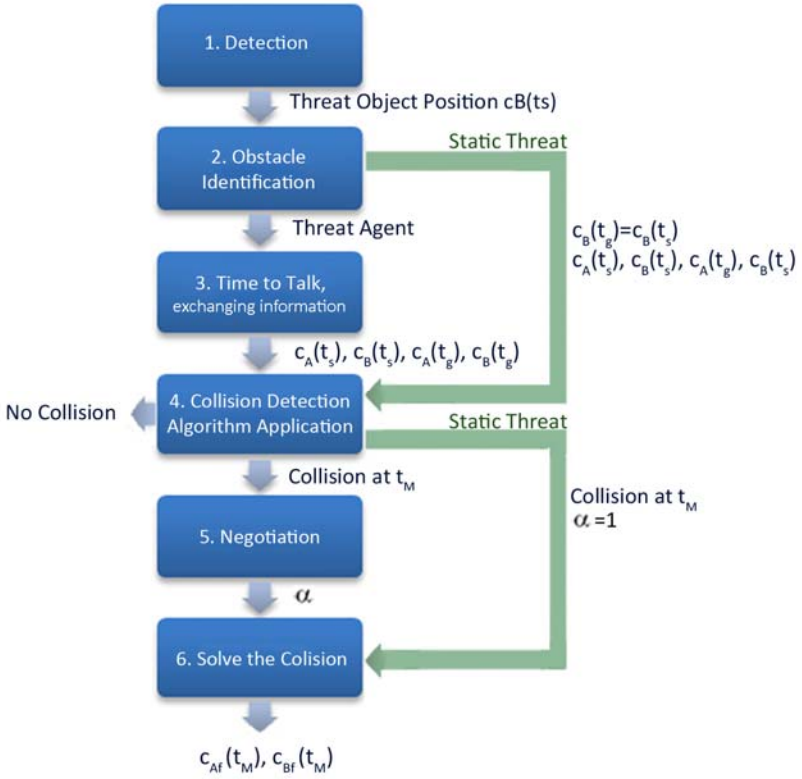


Figure 2. Phases of the proposed methodology.

When the two involved agents in a possible threat have been identified, the communication between them is used to obtain the information needed to apply the detection algorithm presented in 2.1. In the **time to talk, exchanging information** phase, the inputs of the algorithm are four: the positions of each of the agents involved in that instant ($c_A(t_s)$, $c_B(t_s)$) and the target positions where they will be at time t_g ($c_A(t_g)$, $c_B(t_g)$). This time t_g must be the same for the two robots therefore, to calculate it, the agents communicate to each other to know which one reaches its destination before. Who plans to take more time to reach their destination calculates an intermediate destination from its current

This article can be cited as A. Soriano, E. J. Bernabeu, A. Valera and M. Valles, Distributed Collision Avoidance Method Based on Consensus among Mobile Robotic Agents, International Journal of Imaging and Robotics, vol. 15, no. 1, pp. 80-90, 2015.

trajectory and the arrived time of the other agent to its destination. In this way the two agents shared the time it takes to reach their destination.

In the **collision detection** phase, the input requirements to implement collision detection algorithm are: current position coordinates of detector-agent ($c_A(t_s)$), its destination, ($c_A(t_g)$), current position of threat-object ($c_B(t_s)$) and its destination ($c_B(t_g)$). If in the Phase 2, the threat-object was identified as a static-threat, the target is the same as the initial position ($c_B(t_s)= c_B(t_g)$). Therefore the inputs are applied to the algorithm and it returns the probability of collision with the threat-object. In case there is no collision, threat is discarded and the method ends but if a collision is detected, the method informs to detector-agent the time of maximum penetration (t_M). The next step is to avoid the collision by the method described in section 2.2. If the object is a static-threat, the detector- agent should take over the entire cost of the collision avoidance (α=1) and jump to Phase 6. Otherwise the negotiations between the two agents involved are opened to decide how much charge is allocated to each.

To decide the load percentage (α) that each robot will have in the collision avoidance, in the **negotiation** phase the two agents communicate with each other and exchange parameters such as priority, the difficulty of maneuvering, maximum speed, etc., which define the easiness or availability that each agent offers to change its trajectory and avoid collision. Once each agent agreed with the selection of α, the detector-agent runs the last method described below.

The detector-agent, by the method 2.2, computes the two new positions that the robot should be achieve at time t_M to avoid collision in the **solve the collision** phase. The threat-agent receives, from the detector-agent, the avoidance position ($c_{Bf}(t_M)$) and the time in which must be achieve. Both change their trajectories to go to the new destination partial ($c_{Af}(t_M)$, $c_{Bf}(t_M)$) at the right time. Once it's reached, the collision is resolved, each robot continues its original path and the method ends.

The method only involves two agents in the agreement. When a third object intervenes, for now the scheme of Fig. 3 is performed.

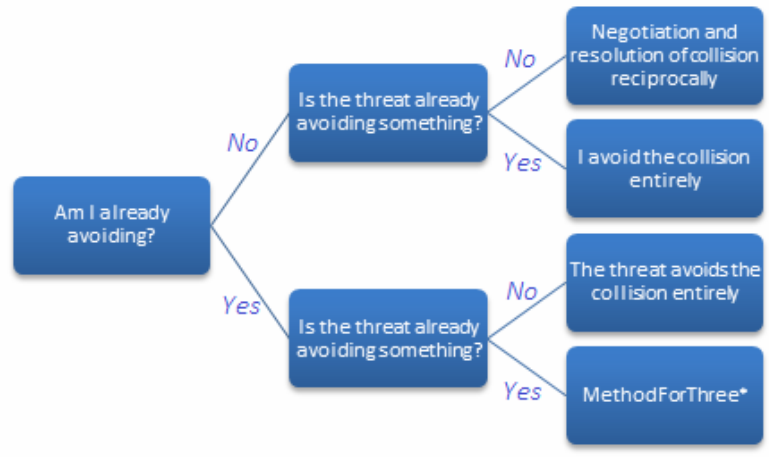


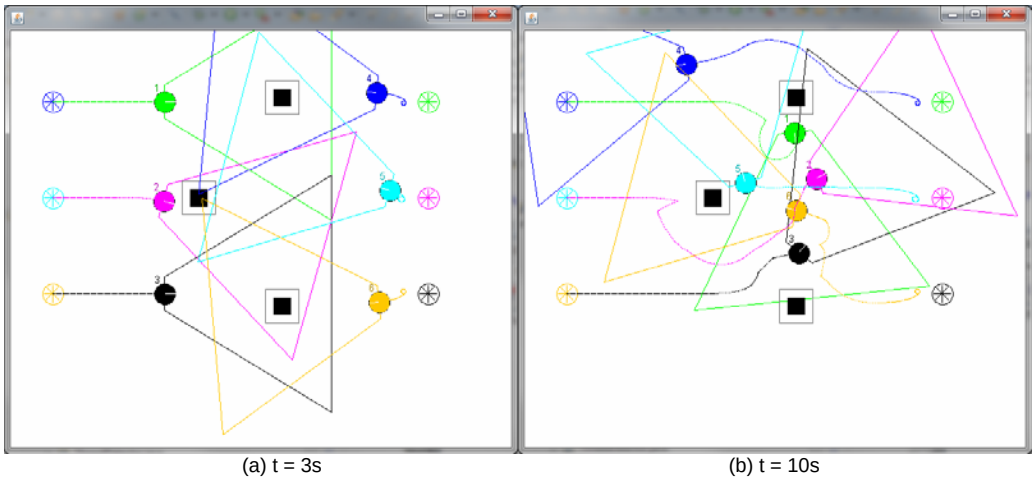
Figure 3. Performance binary tree.

This article can be cited as A. Soriano, E. J. Bernabeu, A. Valera and M. Valles, Distributed Collision Avoidance Method Based on Consensus among Mobile Robotic Agents, International Journal of Imaging and Robotics, vol. 15, no. 1, pp. 80-90, 2015.

The critical point occurs when an agent (a1) which is already avoiding a collision with other agent (a2) detect that exists a new collision with a third agent (a3), who is already avoiding other agent too. MethodForThree* is a provisional method that progressively reduces the speed of the agents a1 and a2, and may even stop them completely until a3 is no longer a threat to a1. Collision detection by agreements involving more than two agents is also being developed.

4. SIMULATION WITH VIRTUAL ROBOTS

Different scenarios with mobile robots have been simulated in order to test the effectiveness of this method. A GUI developed with Java Swing libraries, allows visually verify the correct operation of the algorithm, emulating the movements of the virtual robots modelled from the actual model of a LEGO NXT. Figure 4 shows the execution obtained in two instants (t=3s, t=10s). There are six robots (circles of different colors) that they must arrive to the opposite location (marked by the same color star). The figure also shows the detection area (a trapezoid in front of each robot), three static objects (black squares) and the path described by each robot for the first seconds of the simulation.



(a) t = 3s (b) t = 10s

Figure 4. Collision avoidance simulation with six robots.

Furthermore, for a more detailed visualization of the simulation, VREP software (http://www.coppeliarobotics.com) has been connected to the multi-agent platform. VREP is a 3D robot simulator with integrated development environment. It is based on a distributed control architecture where each object or model can be individually controlled via an embedded script, a plugin, a ROS node or a remote API client. It also allows navigation on the environment through the use of the mouse, making possible to visualize the scenario from anywhere at any time.

To integrate the visualization of simulation in VREP with the multi-agent platform, each robotic agent creates a behavior that opens a socket connection with an instance of VREP program and sends regularly the position of the managed robot to VREP. Thus, whenever an agent updates its position,

This article can be cited as A. Soriano, E. J. Bernabeu, A. Valera and M. Valles, Distributed Collision Avoidance Method Based on Consensus among Mobile Robotic Agents, International Journal of Imaging and Robotics, vol. 15, no. 1, pp. 80-90, 2015.

that change is reflected in VREP. Figure 5 shows, from two different points of view, the first few seconds of the 3D simulation of a scenario with two static objects (white boxes) and six LEGO NXT robots running the collision avoidance algorithm presented. The traveled paths until that moment have been marked out in a yellow outline.

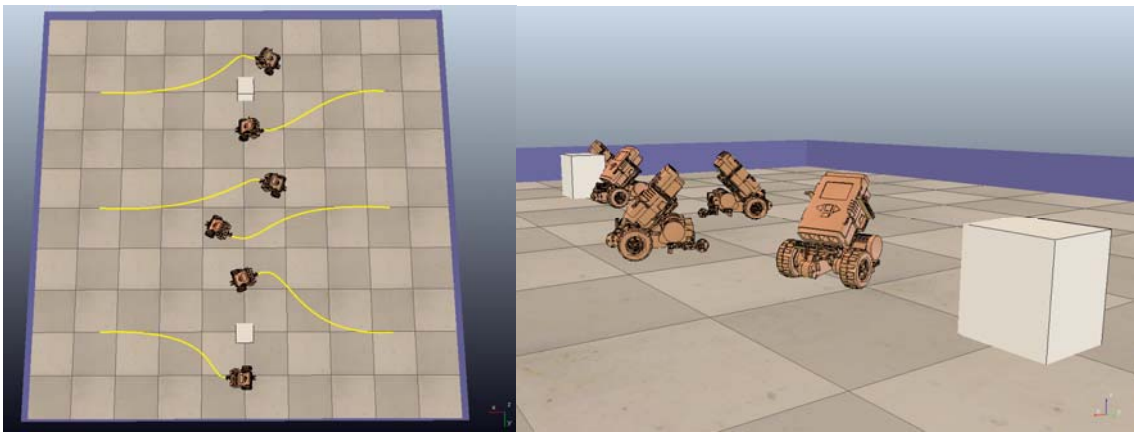


Figure 5. Simulation in VREP of a scenario with two static objects and six robots.

Once the method has been verified through simulation, an experimental deployment with real robots is described in the following section.

5. PRACTICAL IMPLEMENTATION WITH MOBILE ROBOTS

A practical implementation with mobile robots has been developed in order to test the robustness of the presented algorithm. The mobile robots used are LEGO Mindstorms NXT (http://mindstorms.lego.com) and the platform for the management of MAS chosen was JADE (http://jade.tilab.com).

Two LEGO differential wheeled mobile robots have been built (Campion et al., 1996). Each robot has defined two destinations points. In order to achieve the trajectory, a control strategy based on a pure pursuit algorithm (Wallace et al., 1985) was implemented in the robots. The robots have been equipped with a ring of proximity sensors to detect possible obstacles.

Robots are connected to their software agents (computers) via Bluetooth and those computers are part of a network that forms the overall MAS through JADE. The connection diagram is presented in Fig. 6a. Each robot carries a triangle to detect its position from an overhead camera located at the top of the scenario. This camera is also used to monitoring and minimizing odometry problems.

These robots have multiple threads running different functional modules. Each of them has one module to control the robot trajectory, a second one for detection that manages the IR sensors and a third one for communication that receives and sends information to the software agent (see Fig. 6b).

This article can be cited as A. Soriano, E. J. Bernabeu, A. Valera and M. Valles, Distributed Collision Avoidance Method Based on Consensus among Mobile Robotic Agents, International Journal of Imaging and Robotics, vol. 15, no. 1, pp. 80-90, 2015.

While IR sensors does not detect anything, the robot follows its fixed trajectory, but when something is detected by IR, the communication module informs to the software agent and expects a solution to the possible collision from MAS. If the solution leads to a new destination for the robot, the communication module receives the new destination and sends it to the control module for change the path.

The management of the agents in JADE is simple. When a software agent receives the position of a detected threat, the agent asks everyone if anyone is located in the threat area. Thus, if other robot is the threat, it's identified as the threat agent and they exchange their destinations and speeds to verify if there will be a collision or not. If finally there is it, they negotiate the way to avoid it and send the new destinations and speeds to their respective robots.

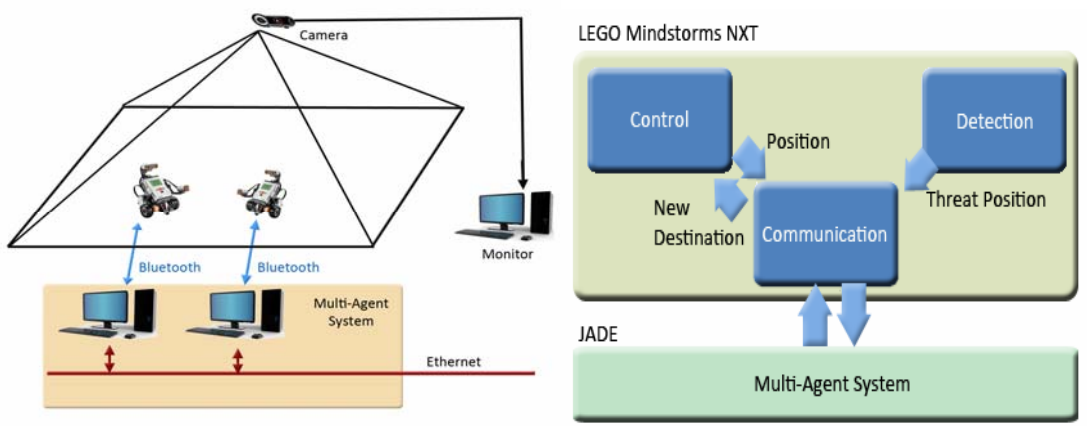


Figure 6. Control architecture and modules connection scheme.

In http://idecona.ai2.upv.es, a video demonstration of practical experiment with Lego robots (*Multimedia* folder, at videos multimedia gallery, with name *Collision avoidance of mobile robots*) and two compiled versions of the platform that allow the simulation with robots (*Desarrollos de Software* folder, at Results option, Project menu) can be obtained.

6. DISCUSSION AND CONCLUSION

A collision avoidance method that takes advantages and benefits of MAS has been presented in this work. This method is located one level above the traditional methods of obstacle avoidance where the management is performed locally and the possible communications between the local systems are solved functionally. The application of techniques provided by the area of artificial intelligence to the robotic area opens a wide range of possibilities that offers more natural results and gives human characteristics of communication like negotiation between robots. This work has succeeded in unifying concepts of agent theory with concepts from the area of mobile robotics, providing more intelligence to robots and offering solutions that otherwise cannot be provided. The methodology has been tested both in simulations and in real executions with mobile robots.

The kinematic configuration of the used agents is holonomic, then considering only linear trajectories might be acceptable. However, as a future work, the collision detection using another kind of movements, like natural Splines and Bezier curves are being considered.

ACKNOWLEDGEMENTS

This work has been partially funded by the Ministerio de Ciencia e Innovación (Spain) under research projects DPI2011-28507-C02-01 and DPI2010-20814-C02-02.

REFERENCES

Bernabeu E. J., 2009, *Fast generation of multiple collision-free and linear trajectories in dynamic environments.* IEEE Trans. Robotics **25**(4). 967-975

Bernabeu E. J., Tornero J., Tomizuka M., 2001, *Collision prediction and avoidance amidst moving objects for trajectory planning applications.* Proceedings of the IEEE Int. Conf. Robot. Automat., pp. 3801-3806.

Bruce, K.B., Cardelli, L., Pierce, B.C., 1997, *Comparing Object Encodings*. In: Abadi, M., Ito, T. (eds.): Theoretical Aspects of Computer Software. Lecture Notes in Computer Science, Vol. 1281. Springer-Verlag, Berlin Heidelberg New York, 415–438

Cameron S. and Culley R. K., 1986 *Determining the minimum translational distance between two convex polyhedra.* Proceeding of the IEEE Int. Conf. Robot. Automat., pp. 591-596.

Campion G., Bastin G., and Dandrea-Novel B., 1996, *Structural properties and classification of kinematic and dynamic models of wheeled mobile robots. Robotics and Automation*, IEEE Transactions on, **12**(1):47–62.

Cherubini A., Chaumette F., 2013, *Visual navigation of a mobile robot with laser-based collision avoidance*, International Journal of Robotics Research, vol. 32, no. 2, pp. 189-205.

Choi Y-K., Wang W., Liu Y. and Kim M-S., 2006, *Continuous collision detection for two moving elliptic disks.* IEEE Trans. Robotics **22**(2). 213-224.

Java Agent Development Framework. http://jade.tilab.com

LEGO Mindstorms home page. http://mindstorms.lego.com

Michalewicz, Z., 1996, *Genetic Algorithms + Data Structures = Evolution Programs*. 3rd ed. Springer-Verlag, Berlin Heidelberg New York

Purcaru C., Precup R.-E., Iercan D., Fedorovici L.-O., David R.-C., Dragan F., 2013, *Optimal robot path planning using gravitational search algorithm*, International Journal of Artificial Intelligence, vol. 10, no. S13, pp. 1-20.

Redon S., Kheddar A. and Coquillart S., 2002, *Fast continuous collision detection between rigid bodies.* Computer Graphic Forum, **21**(3). 279-288.

Russell, S., Norvig, P., 2003, *Artificial Intelligence: A modern approach.*

Saudi A., Sulaiman J., 2013, *Indoor path planning for mobile robot using LBBC-EG*, International Journal of Imaging and Robotics, vol. 11, no. 3, pp. 37-45.

Soriano A., Bernabeu E. J., Valera A., Vallés M., 2013, *Collision Avoidance of Mobile Robots using Multi-Agent Systems*. The International Symposium on Distributed Computing and Artificial Intelligence.

Swingler A., Ferrari S., 2013, *On the duality of robot and sensor path planning*, Proceedings of 52nd IEEE Conference on Decision and Control, Florence, Italy, pp. 984-989.

Van den Bergen G., 2005, *Continuous collision detection of general convex objects under translation.* Morgan Kauffmann Publishers. Game Developers Conf. http://www.dtecta.com/interesting. (2005)

Van Leeuwen, J. (ed.), 1995, *Computer Science Today*. Recent Trends and Developments. Lecture Notes in Computer Science, Vol. 1000. Springer-Verlag, Berlin Heidelberg New York

V-REP. Virtual Robot Experimentation platform. http://www.coppeliarobotics.com

Wallace, R., Stentz A., Thorpe C., Moravec H., Whittaker W., Kanade T, 1985, *First Results in Robot Road-Following*.