A fault-tolerant modular control approach to multi-robot perimeter patrol

Alessandro Marino, Lynne E. Parker, Gianluca Antonelli, Fabrizio Caccavale and Stefano Chiaverini

Abstract— **The use of large scale multi-robot systems is motivated by a number of desirable features, such as scalability, fault tolerance, robustness and lower cost with respect to more complex and specialized agents. This work is focused on a a behavior-based approach to the problem of the multi-robot border patrolling, in the framework of the Null-Space-based Behavioral control (NSB); it is based on two previous works of the same authors, where the feasibility of the approach is demonstrated. Namely, a few aspects of the approach, not yet tackled in previous works, are investigated: its robustness to faults of individual agents, its capability of managing large numbers of robots, the possibility of adding new tasks in the framework of the multi-robot patrolling problem. Along these directions, our approach has been validated in simulation with a large number of robots and sudden faults as well as experimentally on a team composed by three Pioneers 2-DX robots.**

Index Terms— **Behavioral control; Platoon of vehicles; Multi-robot systems; Border Patrol; Swarm Robotics.**

I. Introduction

In swarm robotics the achievement of the mission is the result of cooperation of independent agents forming the swarm. In this approach, the overall behavior is generally *emergent*, i.e., although each unit responds to external stimuli via simple behaviors and, generally, it has not a global cognition of the assigned mission, the couples stimuli-behaviors are organized in such a way the cooperation implicitly raises from the interaction among robots and with the environment. The main advantage of this approach is its robustness to faults of the individual robots, as no predefined role is established, and reorganization can be automatically achieved. Moreover, this approach is clearly modular, since new emergent behaviors at swarm level can be obtained and, in general, more complex missions could be accomplished, by properly defining new simple behaviors at the level of the single unit. There are, of course, some drawbacks connected to this approach. For example, there is not any formal method to generate, starting from the overall mission description, a set of elementary couples stimuli-behaviors, whose interaction leads to task accomplishment, especially in the presence of dynamically changing environments. Moreover, since the architecture decentralized and, in general, the swarm operates in dynamic environments, it is difficult to introduce performance indexes and compare these architectures with other approaches (e.g., deliberative centralized approaches). In fact, traditional benchmarking methodologies are based on the assumption of static environments. When dealing with swarms of autonomous robots, however, this concept is candidate to fail, since the environment, rather than the robots, is not repeatable (e.g., it is impossible to have the same environment conditions, the same obstacle positions, sensor readings and, when required, the same human interaction).

The patrolling problem, which has received considerable attention in the last years [7]-[6], can be considered as a major application field of swarm robotics. Due to the different requirements and environments, it is difficult to give an exact definition of robotic border patrol, in the sense that a patrolling mission may require different objectives to be fulfilled and may be subject to several constraints, depending on various conditions. In [5] an analysis of the main patrolling task issues and some multi-agent-based solutions are presented. The achievements in [5] have been further extended in [4]. In [14] and [10] graph-theory is used to find the optimal solution of a mathematical problem expressing a multi-robot surveillance problem. In [6] the authors analyze non-deterministic paths for a group of homogeneous mobile robots patrolling a frontier. However, in most of the above mentioned works the patrolling problem is approached from an analytical perspective, having in mind and centralized control solutions. Centralized approaches are not necessarily a good choice from a practical point of view [6], since analytical approaches are likely to make the patrolling algorithm predictable. On the other hand, a pure random motion of the robots is unlikely to be effective [5].

In this paper, a swarm architecture, in the framework of the Null-Space-based Behavioral (NSB) control approach [9], is adopted to cope with the multi-robot patrolling problem. The results are an extension of those obtained in two previous works of the same authors [15],[16]; the approach is characterized by the introduction of the concept of *action*, obtained by combining in a consistent way elementary behaviors; once a set of actions is defined, according to the requirements of the patrolling task, an *actions selection* mechanism selects the best action, according to a suitable criterion, based either on external stimuli or, eventually, on the internal state of the robotic swarm; the latter is often used to implement some form of learning and adaptivity. In detail, the work in [15],[16] is extended by considering:

- the presence of *friends* agents interacting with the patrolling swarm;

A. Marino and F. Caccavale are with the Dipartimento di Ingegneria e Fisica dell'Ambiente, Università degli Studi della Basilicata, Viale dell'Ateneo Lucano 10, 85100, Potenza, Italy, {alessandro.marino,fabrizio.caccavale}@unibas.it

L.E. Parker is with the Department of Electrical Engineering and Computer Science, The University of Tennessee, 1122 Volunteer Blvd, TN 37996-3450, Knoxville, USA, leparker@utk.edu

G. Antonelli and S. Chiaverini are with the Dipartimento di Automazione, Elettromagnetismo, Ingegneria dell'Informazione e Matematica Industriale, Università degli Studi di Cassino, Via G. Di Biasio 43, 03043, Cassino (FR), Italy, antonelli,chiaverini@unicas.it

- collision avoidance issues, even in the presence of a large number of robot in the swarm;
- robustness to the occurrence of faults affecting multiple robots in the swarm.

The approach is first tested in simulation, by using the Matlab and Player/Stage [18] environments; then, it is experimentally verified on a team of commercially available mobile robots, namely the Pioneer 2DX robots available at the Distributed Intelligence Laboratory of the University of Tennessee.

II. Control architecture

In Figure 1 a sketch of the adopted robot control architecture is reported. A set of elementary behaviors are defined to accomplish simple tasks. These behaviors are combined in more meaningful *actions* via the Null-space-based Behavioral (NSB) control approach [9], briefly reviewed in Appendix A. It is worth noticing that the use of the NSB approach to compose elementary behaviors ensures a suitable and predictable output, differently from other competitive or cooperative approaches. Once a complete set of actions is defined, the developers need to focus only on the actions selection mechanism via a suitably defined *supervisor*, rather than on the command fusion problem. Moreover, differently form other works, centralized control solutions are not considered here, due to the inherently weakness of grouping all the computational effort in one single machine, even if remote.

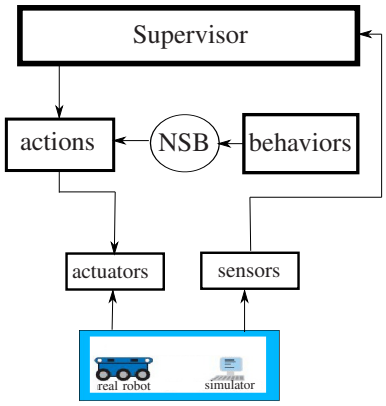

Fig. 1. The overall single-robot architecture.

III. NSB control for multi-robot patrolling

The architecture described in Section II has been adopted to design a control scheme for multi-robot patrolling problems [15],[16]. The following assumptions are adopted to develop the proposed solution:

- each robot can localize itself in the environment, at least with respect to the border or other agents;
- each robot knows or can estimate the geometric description of the border locally to its position;
- each robot is characterized by a visibility area, where it recognizes the presence of another patrolling robot, a friend or an hostile agent;
- each robot is characterized by its own safety area (contained in the visibility area), where other agents are not allowed to enter.
- it is forbidden to the robots any kind of explicit communications;
- each robot is autonomous, in the sense that it does not rely on a central computational unit; moreover, distributed control algorithms that need an explicit exchange of information (e.g., as those based on consensus) are not used;
- each robot is aware of the existence of other patrolling robots, friends and hostile agents;
- each robot does not know the total number of patrolling robots.

Clearly, these are very severe assumptions aimed at conferring an high degree of autonomy and robustness to the whole system.

A. Elementary Behaviors and Actions

In the case of the mono-dimensional border patrolling problem, a set of elementary behaviors is defined:

- `Reach Frontier`
- `Patrol Frontier Clockwise`
- `Patrol Frontier Counter-Clockwise`
- `Teammate Avoidance`
- `Friend Avoidance`

whose semantics and analytical expressions are given in Appendix B.

As motivated above, it is appropriate to compose the elementary behaviors into more complex behaviors; the latter are sometimes defined as *behavior sets* in the literature. Similar to the concept of behavior set, here a higher abstraction layer is introduced: the *action*. As shown in Figure 1 and Appendix C, an action is given by the proper composition, achieved via NSB, of several elementary behaviors and represents a macroscopic attitude of the robotic system. Only one single action can be active at once.

For the specific case of the border patrolling task, the following set of actions is obtained by combining the elementary behaviors defined above:

- `Action Reach Frontier`
- `Action Keep Going`
- `Action Patrol Clockwise`
- `Action Patrol Counter-Clockwise`
- `Action Teammate Avoidance`
- `Action Friend Avoidance`

According to the definitions in Appendix C, each action is given by elementary behaviors arranged in priority; e.g., the `Reach Frontier` action properly combines the elementary behaviors `Stay on Frontier` and `Teammate Avoidance`, depending on the sensed presence of other patrolling robots in the visibility range and the distance from the border. It is worth noticing that such actions require that each robot is able to recognize other agents and their nature (friends or teammates) and localize itself in the environment or with respect to the border.

B. The supervisor

The Supervisor in Figure 1 is in charge of selecting the next action to be executed, based on sensing information available to the single robot and the current state of the robot. Different solutions can be adopted; among them, a finite state automata (see [15]) allows to code in a simple way all the possible states and transitions between states. In each state, only one single action is active. Another choice is represented a Fuzzy Inference System in [16] that allows to use linguistic rules to code state transitions. Another advantage of using the latter is the possibility to command smooth transitions among the states to avoid discontinuities on robot velocities. Clearly, other paradigm can be adopted using, for example, motivational function and other bio-inspired architectures. In this paper a finite state automata is used; details on the automata can be found in [15].

IV. Case Studies

As stated in Section I, we are interested in a decentralized robust solution to the patrolling task. Execution of a patrolling mission requires an high level of autonomy and, depending on applications and sensors range, an high number of agents. Therefore, at any time instant, the number of patrolling robots can suddenly vary due to robot faults or battery charging, making difficult any optimal centralized strategy. Several simulations on closed and open border, with different sizes and shapes, have been carried out by using both Matlab [13] and Player/Stage [18] environments. They are briefly described in IV-A; videos are available at [3]. In Section IV-B, experimental results are discussed. Videos of experiments are available at [1]-[2].

A. Simulations

In the following, Matlab simulations are briefly described so as to show the algorithm behavior in the presence of large number of patrolling robots, robot faults and presence of friend agents. The team is composed by 60 robots, with a visibility and safety area equal to 20 m. The scalar gains in eqs. (8)–(10) (see Appendix B) have been chosen as $\lambda_{rf} = 12$, $\lambda_{cw} = 12$, $\lambda_{ccw} = 12$, $\lambda_{ta} = 15$ and $\lambda_{fa} = 15$. Figure 2 shows the robot positions at different time instants. Robots are represented by points surrounded by their visibility range and safety area (a continuous circle); while the continuous thick line represents the border to be patrolled. In the first frame, robots are randomly distributed and they are trying to reach the border activating the `Reach Frontier` action, while avoiding other teammates. In the second frame some robots have reached the border and they have started to perform the patrolling mission activating the actions `Keep Going` and `Patrol Clockwise` (or `Patrol Counterclockwise`). Friend agents, represented by cross markers surrounded by their safety area (a dash-dot circle of 20m radius), start to approach the border and the cross it (third frame) without collisions with patrolling robots. Then, in the fourth frame friends gain the center of the bordered zone; in this situation, patrolling robots are not more affected by friends motions. As can be noticed, the effective definition of the behaviors and actions allows collisions avoidance even in case of high robot density. Moreover, patrolling vehicles on the border don't allow other teammates approaching the border to influence their motion, thus preventing conflicting situations; at the same time, by properly selecting actions (i.e., `Patrol CW` and `Patrol CCW`) they keep themselves at the safety distance from other teammates patrolling the line.

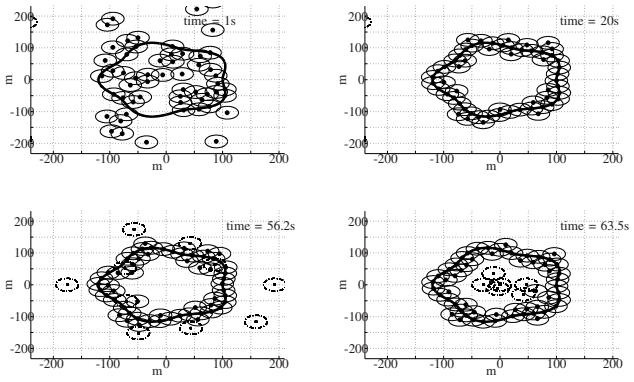

Fig. 2. Sample frames at different time instants of robots performing a patrolling mission.

It is useful to remark that the approach prevents robots from collisions. To this aim, in Figure 3 the minimum value over time of the distances among all the possible robot couples is depicted, i.e.:

$$d_{r,min}(t) = \min_{\forall i \neq j \ i,j \in Nr} \parallel p_{i,r}(t) - p_{j,r}(t) \parallel, \qquad (1)$$

where $p_{i,r}(t)$ is the position of i-th robot at instant t, and N_r is the set of patrolling robots.

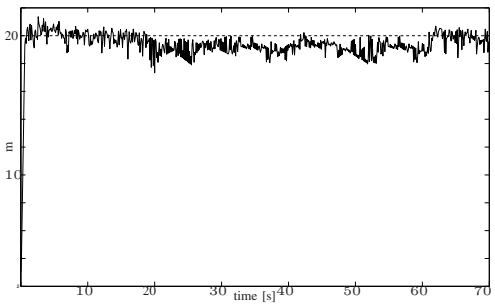

Fig. 3. Minimum of distances between all possible robot couples.

In the same way, in Figure 4 it is shown the minimum distance over time of all the possible couples robot-friend, i.e.:

$$d_{f,min}(t) = \min_{\forall i \in N_r, j \in N_f} \parallel p_{i,r}(t) - p_{j,f}(t) \parallel, \qquad (2)$$

where $p_{j,f}(t)$ is the position of j-th friend agent at instant t and N_f is the set of friend agents.

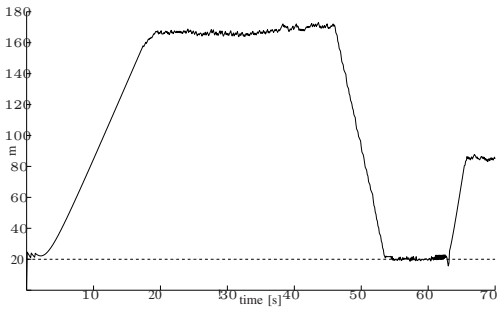

Fig. 4. Minimum of distances between all robot-friend couples.

Figures 3 and 4 clearly show that the control is capable of avoiding collisions, and thus the mission is safely performed.

Finally, the performance in the presence of multiple robot faults are tested. Namely, the same patrolling mission is executed, but some of the patrolling robots fail (first frame of Figure 5). These failures are virtually represented by robot disappearances from the scene. The last frame shows the new situation, where the robots, without any external influence, redistribute along the border, thus fulfilling the mission objective.

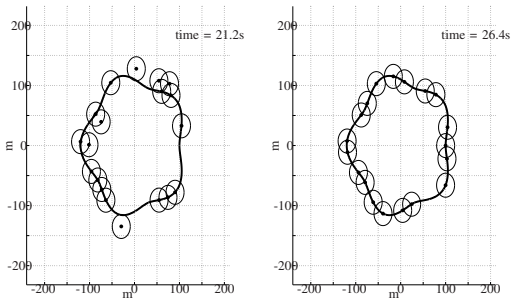

Fig. 5. In left plot several robots fails. In the right plot remaining robots automatically redistribute around the border.

B. Experiments

Experiments have been performed on a robot team composed by three Pioneer 2-DX robots (0.44 m long, 0.38 m wide, and 0.22 m tall); they are equipped with a two-wheel drive and a passive caster, two rings of sonars (8 front and 8 rear), a SICK laser range-finder, a pan-tilt-zoom color camera, on-board computation on a PC104 stack and Player control software [18].

Figure 6 shows a portion of the patrolled area and its representation in Player/Stage. The solid line represents the border, the triangles represent the robots together with their visibility range. In detail, the border is a closed line composed by segments joined by arcs; and its overall length is 51 m. The robots know the geometry of the border and approach it at a speed of 0.35 m/s ($\lambda_{rf} = 0.35$), patrol at a speed of 0.35 m/s ($\lambda_{cw} = \lambda_{ccw} = 0.35$) and escape other teammates at a speed of 0.35 m/s ($\lambda_{ta} = 0.35$). The localization in the environment is achieved by a pre-built map and a localization driver based on an adaptive particle filter ([12]) available in the Player control software. Visibility range and safety area are equal to 2.5 m. Moreover, when action `Keep Going` is active (see Appendix C), the robot can decide to invert its motion direction, every 30 s, according to a random variable.

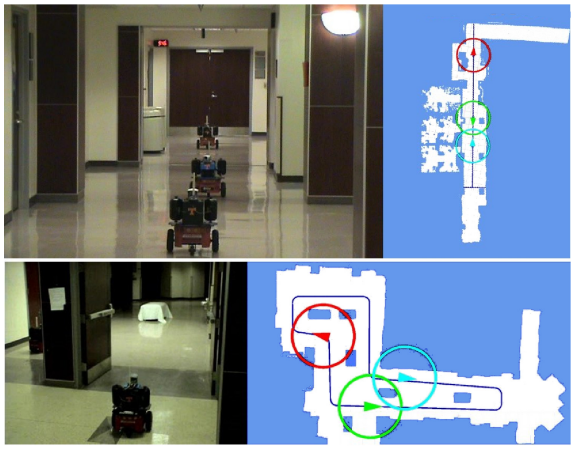

Fig. 6. Experiments scenarios. Left: pictures of the environments. Right: environments representations in the Player/Stage software, patrolled borders, patrolling robots (with their visibility areas).

Robots start patrolling asynchronously; in their initial position are close to the border, hence the `Keep Going` is active. During patrolling phases, robots are usually in the state `Keep Going` (`Clockwise` or `Counter-Clockwise`). When two robots encounter each other, one robot experiences a sudden transition `Keep Going` → `Patrol Clockwise`, while the other undergoes the dual tranistion `Keep Going` → `Patrol Counter-Clockwise`; after the interaction, they both return to the `Keep Going` state, proceeding in opposite directions. A sequence of the movements occurring in this situation is shown in Figure 7.

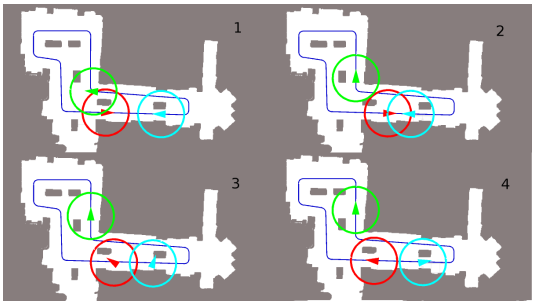

Fig. 7. Three robots team performing the patrol mission. The lower ones (in red and cyan) meet along the path and invert their motion directions. The arrows represent forward motion direction.

In order to assess performance of low level actions in reaching and staying on the border, the distance from the border for the three robots have been reported in Figure 8. The distances are within 0.1 m when no fault occurs; this value is acceptable for the experimental conditions and requirements. Peaks are reached during rotation movements due to sensor noise and, above all, to the neglected robot dynamics in the control law.

A fault occurs to robot number 2 after 28 minutes, then the robot is manually driven far from the border and reactivated at minute 41. Two faults occur to robot number 3, after 15 and 33 minutes. As can be noticed in Figure 8, thanks to the decentralized structure of the control law, faults do not affect the overall behavior of the swarm.

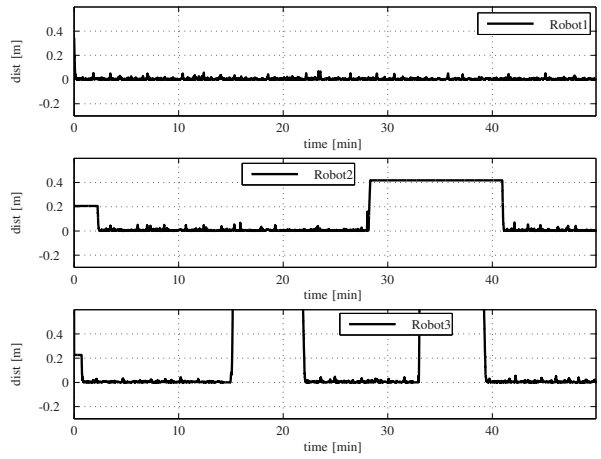

Fig. 8. Distance from the border.

APPENDIX

A. NSB review

In the following, a brief review of the NSB approach is provided. Let $\boldsymbol{\sigma} \in \mathbb{R}^m$ be the mathematical representation of the behavior to be implemented (often referred as task) and $\boldsymbol{p} \in \mathbb{R}^n$ be the vector of variables describing the system configuration; in general, they are related via the following model

$$\boldsymbol{\sigma} = \boldsymbol{f}(\boldsymbol{p}) \,, \tag{3}$$

with the corresponding differential relationship

$$\dot{\boldsymbol{\sigma}} \;=\; \frac{\partial \boldsymbol{f}(\boldsymbol{p})}{\partial \boldsymbol{p}} \boldsymbol{v} \;=\; \boldsymbol{J}(\boldsymbol{p}) \boldsymbol{v} \,, \tag{4}$$

where $\boldsymbol{J} \in \mathbb{R}^{m \times n}$ is the configuration-dependent task Jacobian matrix and $\boldsymbol{v} \in \mathbb{R}^n$ is the system velocity.

An effective way to generate motion references for the vehicles, $\boldsymbol{p}_d(t)$, starting from the desired behavioral function, $\boldsymbol{\sigma}_d(t)$, is to act at the differential level by inverting the (locally linear) mapping (4); in fact, this problem has been widely studied in robotics (see, e.g., [17] for a tutorial). A typical requirement is to pursue minimum-norm velocity in a closed loop version, leading to

$$\boldsymbol{v}_d = \boldsymbol{J}^{\dagger} \left(\dot{\boldsymbol{\sigma}}_d + \boldsymbol{\Lambda} \widetilde{\boldsymbol{\sigma}} \right) = \boldsymbol{J}^T \left(\boldsymbol{J} \boldsymbol{J}^T \right)^{-1} \left(\dot{\boldsymbol{\sigma}}_d + \boldsymbol{\Lambda} \widetilde{\boldsymbol{\sigma}} \right) \,, \tag{5}$$

where $\boldsymbol{\Lambda}$ is a suitable constant positive-definite matrix gain and $\widetilde{\boldsymbol{\sigma}} = \boldsymbol{\sigma}_d - \boldsymbol{\sigma}$ is the task error.

As described earlier, elementary behaviors are properly composed in more meaningful *Actions* by adopting a priority-based approach. Elementary behaviors are arranged in such a way that the index i related to the i-th behavior denotes its degree of priority (i.e., behavior 1 has the highest priority). In the case of 3 behaviors, according to [11], solution (5) is modified into

$$\boldsymbol{v}_d = \boldsymbol{v}_1 + \boldsymbol{N}_1 \boldsymbol{v}_2 + \boldsymbol{N}_{12} \boldsymbol{v}_3 \,, \tag{6}$$

where v_i is the velocity contribution (in the form (5)) due to the i-th behavior,

$$\boldsymbol{N}_1 = \left(\boldsymbol{I} - \boldsymbol{J}_1^{\dagger} \boldsymbol{J}_1 \right) \tag{7}$$

is the null space projection matrix of $\boldsymbol{J}_1$, and $\boldsymbol{N}_{12}$ is the null space projection matrix of the Jacobian obtained by stacking the Jacobians of the higher priority behaviors, i.e., $\boldsymbol{J}_{12} = \left[\boldsymbol{J}_1^T \; \boldsymbol{J}_2^T \right]^T$.

In this way, lower priority behaviors are executed only in their components not affecting higher priority behaviors; hence, differently from other command fusion approach, the output is predictable. Also, convergence to zero of task errors can be guaranteed for properly defined tasks [8]. On the other hand, a differentiable analytic expression of the defined behaviors is required, so as to compute the required Jacobians.

B. Elementary Behaviors definition

1) Reach Frontier: Given the robot position $\boldsymbol{p}_r \in R^2$ and the border B, $\boldsymbol{p}_B \in R^2$ is the closest point to $\boldsymbol{p}_r$ belonging to B. The behavior reach frontier is simply defined as:

$$\begin{cases} \sigma_{rf} = \| \boldsymbol{p}_r - \boldsymbol{p}_B \|, \; \sigma_{rf,d} = 0, \\ \boldsymbol{J}_{rf} = \boldsymbol{r}_{rf}^T, \; \boldsymbol{J}_{rf}^{\dagger} = \boldsymbol{r}_{rf}, \; \boldsymbol{N}_{rf} = \boldsymbol{I}_2 - \boldsymbol{r}_{rf} \boldsymbol{r}_{rf}^{\mathrm{T}}, \\ \boldsymbol{v}_{rf} = \lambda_{rf} \boldsymbol{r}_{rf} \left(-\sigma_{rf} \right), \end{cases} \tag{8}$$

where $\boldsymbol{r}_{rf} = \left(\boldsymbol{p}_r - \boldsymbol{p}_B \right) / \| \boldsymbol{p}_r - \boldsymbol{p}_B \|$, $\boldsymbol{J}_{rf}$ is the task Jacobian, $\boldsymbol{I}_2 \in R^{2 \times 2}$ is the identity matrix, $\boldsymbol{N}_{rf}$ is the null-space projection matrix and λ_{rf} is a positive scalar gain. It is worth noticing that computation of $\boldsymbol{p}_B$ may require proper approximations [6].

2) Patrol Frontier Clockwise: Given the border B and a point $\boldsymbol{p}_B$ belonging to B, $\boldsymbol{r}_{cw}$ is the unit vector tangent to the border in $\boldsymbol{p}_B$ and oriented in the clockwise direction of the border. The behavior is then directly defined as:

$$\begin{cases} \boldsymbol{v}_{cw} = \lambda_{cw} \boldsymbol{r}_{cw}, \\ \boldsymbol{N}_{cw} = \boldsymbol{I}_2 - \boldsymbol{r}_{cw} \boldsymbol{r}_{cw}^{\mathrm{T}}, \end{cases} \tag{9}$$

where $\boldsymbol{r}_{cw}$ plays the role of the task Jacobian, $\boldsymbol{N}_{cw}$ is the null-space projection matrix and λ_{cw} is a positive scalar gain.

3) Patrol Frontier Counter-Clockwise: This case is formally similar to the previous with the obvious difference to properly orient the vector tangent to the border in the counter-clockwise direction.

4) Teammate Avoidance: Given the robot position, $\boldsymbol{p}_r$, the obstacle position closest to the robot, $\boldsymbol{p}_t$, and the safety distance, d_s, the behavior teammate avoidance is defined as:

$$\begin{cases} \sigma_{ta} = \| \boldsymbol{p}_r - \boldsymbol{p}_t \|, \; \sigma_{ta,d} = d_s, \\ \boldsymbol{J}_{ta} = \boldsymbol{r}_{ta}^T, \; \boldsymbol{J}_{ta}^{\dagger} = \boldsymbol{r}_{ta}, \; \boldsymbol{N}_{ta} = \boldsymbol{I}_2 - \boldsymbol{r}_{ta} \boldsymbol{r}_{ta}^{\mathrm{T}}, \\ \boldsymbol{v}_{ta} = \lambda_{ta} \boldsymbol{r}_{ta} \left(d_s - \sigma_{ta} \right), \end{cases} \tag{10}$$

where $\boldsymbol{r}_{ta} = \left(\boldsymbol{p}_r - \boldsymbol{p}_t \right) / \| \boldsymbol{p}_r - \boldsymbol{p}_t \|$, $\boldsymbol{J}_{ta}$ is the task Jacobian, $\boldsymbol{N}_{ta}$ is the null-space projection matrix and λ_{ta} is a positive scalar gain.

5) Friend Avoidance: A *friend* is an agent that moves independently from other agents and that is allowed to cross the border. Therefore, when a friend tries to cross the border, patrolling agents should keep a desired distance from it without affecting its motion. Given the robot position $\boldsymbol{p}_r$, the friend position $\boldsymbol{p}_f$ and a safety distance d_s, the behavior friend avoidance is defined as:

$$\begin{cases} \sigma_{fa} = \|\boldsymbol{p}_r - \boldsymbol{p}_f\|, \ \sigma_{fa,d} = d_s, \\ \boldsymbol{J}_{fa} = \boldsymbol{r}_{fa}^T, \ \boldsymbol{J}_{fa}^\dagger = \boldsymbol{r}_{fa}, \ \boldsymbol{N}_{fa} = \boldsymbol{I}_2 - \boldsymbol{r}_{fa}\boldsymbol{r}_{fa}^{\mathrm{T}}, \\ \boldsymbol{v}_{fa} = \lambda_{fa}\boldsymbol{r}_{fa}\left(d_s - \sigma_{fa}\right), \end{cases} \quad (11)$$

where K_{fa} is positive definite diagonal matrix and N_{fa} is the null-space projector matrix.

C. Actions Definition

The elementary behaviors defined in Appendix B are the basis to build the actions defined in Section III-A. In the following, details of the actions defined for the patrolling problem are provided.

1) Reach Frontier (ARF): This action allows the robot to reach the border when it is far from it. In this case, the definition of the action simply coincides with the elementary behavior `Reach Frontier`:

$$\boldsymbol{v}_{Arf} = \boldsymbol{v}_{rf}, \quad (12)$$

2) Patrol Clockwise (APCW): This action allows the robot to stay on the border, while covering it in the clockwise direction. This action is obtained by combining the `Reach Frontier` and the `Patrol Frontier Clockwise` behaviors in the NSB sense:

$$\boldsymbol{v}_{Apcw} = \boldsymbol{v}_{rf} + \boldsymbol{N}_{rf}\boldsymbol{v}_{cw}, \quad (13)$$

i.e., `Reach Frontier` is the higher priority behavior.

3) Patrol Counter-Clockwise (APCCW): This case is formally similar to the previous, with the obvious difference to properly consider `Patrol Frontier Counter-Clockwise` behavior.

4) Keep Going (AKG): This action allows the robot to stay on the border, while covering it in the clockwise or counter-clockwise direction. This action is obtained by combining the `Reach Frontier` and the `Patrol Frontier Clockwise` (`Patrol Frontier Counter-Clockwise`) behaviors in the NSB sense:

$$\boldsymbol{v}_{Akg} = \boldsymbol{v}_{rf} + \boldsymbol{N}_{rf}\boldsymbol{v}_{p}, \quad (14)$$

where $\boldsymbol{v}_p$ is the vector tangent to the border in the closest point belonging to the border. It can be oriented clockwise or counter-clockwise, according to the current state.

5) Teammate Avoidance (ATA): When a teammate vehicle enters the safety area of another robots, it needs to avoid the teammate while trying to stay on the border or to reach it; in this way it can restart the patrol mission once the teammate-vehicle is far enough. This action can be obtained combining the behaviors `Teammate Avoidance` and *Reach Frontier* in the NSB sense:

$$\boldsymbol{v}_{Ata} = \boldsymbol{v}_{ta} + \boldsymbol{N}_{ta}\boldsymbol{v}_{rf}. \quad (15)$$

Since `Reach Frontier` is the secondary behavior, only its velocity components that do not conflict with the primary behavior will be executed.

6) Friend Avoidance (AFA): This action is not mathematically different from the action `Teammate Avoidance`. When a friend vehicle enters the safety area of a patrolling robot, the latter should not affect the motion of the former, while trying to stay on the border or to reach it. This action can be obtained combining the behaviors `Friend Avoidance` and `Reach Frontier` in the NSB sense:

$$\boldsymbol{v}_{Afa} = \boldsymbol{v}_{fa} + \boldsymbol{N}_{fa}\boldsymbol{v}_{rf}. \quad (16)$$

Also in this case, `Reach Frontier` is the secondary behavior, only its velocity components that do not conflict with the primary behavior will be executed.

REFERENCES

[1] http://www.cs.utk.edu/dilab/movies/Videoclosedborder.avi.
[2] http://www.cs.utk.edu/dilab/movies/Videoopenborder.avi.
[3] http://www.cs.utk.edu/dilab/movies/Videosimulation.avi.
[4] G. Ramalho H. Santana P. Tedesco T. Menezes A. Almeida and V. Corruble. Recent advances on multi-agent patrolling. *Proceedings of the Brazilian Symposium on Artificial Intelligence*, 2004.
[5] G. Ramalho J.D. Zucker A. Drogoul A. Machado. Multi-agent patrolling: an empirical. pages 155–170, 2002.
[6] N. Agmon, S. Kraus, and G. A. Kaminka. Multi-robot perimeter patrol in adversarial settings. In *Proceedings 2008 IEEE International Conference on Robotics and Automation*, pages 2339–2345, Pasaena, CA, May 2008.
[7] M. Ahmadi and P. Stone. A multi-robot system for continuous area sweeping tasks. In *Proceedings 2006 IEEE International Conference on Robotics and Automation*, pages 1716–1723, Orlando, Florida, May 2006.
[8] G. Antonelli. Stability analysis for prioritized closed-loop inverse kinematic algorithms for redundant robotic systems. In *Proceedings 2008 IEEE International Conference on Robotics and Automation*, pages 1993–1998, Pasadena, CA, May 2008.
[9] G. Antonelli and S. Chiaverini. Kinematic control of platoons of autonomous vehicles. *IEEE Transactions on Robotics*, 22(6):1285–1292, Dec. 2006.
[10] Y. Chevaleyre. Theoretical analysis of the multi-agent patrolling problem. *Procedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology*, pages 302–308, 2004.
[11] S. Chiaverini. Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators. *IEEE Transactions on Robotics and Automation*, 13(3):398–410, 1997.
[12] D. Fox, W. Burgard, F. Dellaert, and S. Thrun. Monte carlo localization: Efficient position estimation for mobile robots. In *Proc. of the National Conference on Artificial Intelligence*, 1999.
[13] http://www.mathworks.com.
[14] A. Kolling and S. Carpin. Multi-robot surveillance: an improved algorithm for the graph-clear problem. In *Proceedings 2008 IEEE International Conference on Robotics and Automation*, pages 2360–2365, Pasaena, CA, May 2008.
[15] A. Marino, L. Parker, G. Antonelli, and F. Caccavale. Behavioral control for multi-robot perimeter patrol: A finite state automata approach. In *Proceedings 2009 IEEE International Conference on Robotics and Automation*, Kobe, J, May 2009.
[16] A. Marino, L. Parker, G. Antonelli, and F. Caccavale. A fuzzy-based multiple robots border patrol. In *Proceedings 2009 17th Mediterranean Conference on Control and Automation.*, Thessaloniki, Greece, June 2009.
[17] B. Siciliano. Kinematic control of redundant robot manipulators: A tutorial. *Journal of Intelligent Robotic Systems*, 3:201–212, 1990.
[18] R. T. Vaughan, B. Gerkey, and A. Howard. The Player/Stage project: Tools for multi-robot and distributed sensor systems. In *In Proceedings of the 11th International Conference on Advanced Robotics, (ICAR)*, pages 317–323, Coimbra, Portugal, June 2003.