The 2009 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 11-15, 2009 St. Louis, USA

Traffic Control for a Swarm of Robots: Avoiding Group Conflicts

Leandro Soriano Marcolino and Luiz Chaimowicz

Abstract— **A very common problem in the navigation of robotic swarms is when groups of robots move into opposite directions, causing congestion situations that may compromise performance. In this paper, we propose a distributed coordination algorithm to alleviate this type of congestion. By working collaboratively, and warning their teammates about a congestion risk, robots are able to coordinate themselves to avoid these situations. We executed simulations and real experiments to study the performance and effectiveness of the proposed algorithm. Results show that the algorithm allows the swarm to navigate in a smoother and more efficient fashion, and is suitable for large groups of robots.**

I. Introduction

Large groups of robots have been receiving much attention in recent years. Generally called *swarms*, these systems employ a large number of simple agents to perform a great range of tasks. Generally, a swarm of robots must work in a distributed fashion and use limited communication resources. Due to these characteristics, new algorithms to control and coordinate these large groups of robots have been developed.

One of the difficulties encountered in the navigation of a swarm is congestion: a large number of robots moves towards the same region of the environment at the same time, causing conflicts that waste time and resources. This problem can appear when the congestion region is a target for many robots, for example during *waypoint navigation*, or when groups of robots move in opposite directions and face each other while navigating. This second case, in particular, appears very often. For example, one group might be moving from a base to a target, while other group is returning from the target to the base. Particularly, we observed these problems in [1], where congestion situations often happened during the navigation of a swarm of robots.

Although there are many works dealing with traffic control or collision avoidance, they generally are not appropriate for the context presented here. Traffic control algorithms usually are developed for structured environments where robots navigate in delimited lanes and meet at specific intersections, where it is necessary to choose which one will pass first. Collision avoidance algorithms generally are developed for and tested only in a small group of robots. When large groups of robots are considered, it is very hard to negotiate free paths for all of them in a distributed fashion using regular collision avoidance algorithms. In this case, the congestion problems persist.

This work is partially supported by Fapemig and CNPq. The authors would like to thank Renato Garcia for his help with the localization system and the development of the epuck driver.

The authors are with the Vision and Robotics Laboratory (VeRLab), Computer Science Department, Federal University of Minas Gerais, Brazil. emails: {soriano,chaimo}@dcc.ufmg.br

Therefore, we can see that developing new solutions for traffic control of a large number of agents are of great importance to improve the navigation of a swarm, decreasing the waste of time and resources caused by congestion situations. The objective of this paper is to investigate and develop methodologies to control the traffic of a swarm of robots, in the case where groups of robots move in opposite directions in unstructured environments. We propose a distributed algorithm that allows the robots to warn their teammates and dynamically change their trajectories in order to avoid congestion. We perform a series of simulations and real experiments in different scenarios to show the effectiveness and efficiency of the proposed approach. In a companion paper [2], we also developed a solution for the congestion situation where many robots have a common target.

II. Related Work

When navigating a large number of robots, the combined configuration space can be very complex. Therefore, a common approach is to control the robots in a decentralized way, mixing gradient descent techniques with local repulsion forces [3], [4]. However, this can lead to congestion situations, decreasing the efficiency of the system. Specifically, in a recent work [1], we noticed that many robots got "stuck on traffic" when groups moved in opposite directions. These conflicts delayed the navigation and compromised performance. It is important, therefore, to coordinate the robots in order to enable the swarm to have a better navigation.

The traffic control problem is an important research topic. In [5], it is characterized as a *resource conflict* problem and the importance of its study is emphasized. Works dealing with traffic control started to appear in the late 1980s. In [6], for example, many policies are presented to avoid the congestion of robots in a factory. In [7] traffic rules are shown to navigate a group of robots. In general, works on this area assume that the robots navigate in delimited lanes (like streets or roads). These lanes meet in intersections, where congestion may happen. The traffic control, in general, is executed only at these intersections.

More recent works can be found both in the cooperative robotics field and in the multi-agent systems field. Some works use a manager agent to administrate the traffic at intersections where congestion may happen, as in [8]. A similar approach, in the robotics field, can be seen in [9], where a sensor network is used to coordinate the traffic of a group of robots. Others are working in manager free scenarios, as in [10], which presents a completely distributed algorithm that, based on a spatial temporal pattern, coordinates the movement of robots into intersections or junctions.

However, these methods do not solve the proposed problem, as they assume a structured environment, in which there are fixed locations where the robots may encounter. Besides, they focus in selecting which robot will pass first in the intersection or junction, which is not what we need to solve.

In [11], a mechanism is proposed to avoid congestion in crowd simulations. The authors propose an approach in which agents plan early to avoid congestion, enabling smoother trajectories than when using local repulsion forces. The method, however, is too centralized to be used with a swarm of robots.

Instead of dealing with traffic control, there are works that tries to find more efficient approaches to collision avoidance than using local repulsion forces. In [12], an algorithm is proposed in which robots coordinate their velocities in order to avoid a collision. The coordination may entail not only the robots directly involved in the probable collision, but the robots in the neighborhood as well, which might have to change their velocities to help the robots involved. Other works that deal with collision avoidance are [13], [14], [15], [16]. However, avoiding collisions do not necessarily mean avoiding congestions. Even with a good collision avoidance behavior, the system can still become cluttered and inefficient. Besides, in general these works do not show cases with a large number of robots.

As can be seen, although there are many works dealing with traffic control and collision avoidance, to the best of our knowledge there is no algorithm that deals directly with the proposed problem, where large groups of robots move in opposite directions in an unstructured environment and must coordinate themselves in a distributed, robust and fault-tolerant fashion. The main contribution of this paper is a decentralized coordination algorithm that allows a swarm of robots to prevent congestion in these situations using only local sensing and communication, without assuming the use of delimited lanes nor needing an external infra-structure to control the traffic.

III. Methodology

We are going to describe our algorithm considering a conventional potential field approach, since this is the most common method in swarm navigation: robots are attracted by the goal and repelled by their neighbors in order to avoid collisions among the group.

Thus, given a fully actuated robot i, with dynamic model given by $\dot{\mathbf{q}}_i = \mathbf{v}_i$, $\dot{\mathbf{v}}_i = \mathbf{u}_i$, where $\mathbf{q}_i = [x_i, y_i]^T$ is the pose of the robot i, $\mathbf{u}_i$ is the control input and $\mathbf{v}_i$ is the velocity vector, the following control law is used:

$$\mathbf{u}_i = k_1 \frac{f(\mathbf{ta}_i)}{||f(\mathbf{ta}_i)||} - k_2 \sum_{j \in N_i} \frac{1}{\mathbf{q}_j - \mathbf{q}_i} - k_3 \dot{\mathbf{q}}_i \qquad (1)$$

The constants k_1, k_2 and k_3 are positive. The first term is the attraction force of the robot towards the target: function f calculates the vector that points towards the target of the robot i, $\mathbf{ta}_i$. The second term represents the local repulsion forces. The robots in the neighborhood of robot i are represented by the set N_i. We define as a neighbor every robot that is within a certain limit, δ, of distance from robot i. The third term is a damping force, used to improve stability, mainly in simulations.

The coordination algorithm works as follows: we assume that every robot, i, is able to know the direction of its target, d_i. Besides, we assume that robots are able to communicate locally with all the robots that are within a maximum distance α ($\alpha \geq \delta$). The general idea of the algorithm is that the first robots to notice the risk of congestion should warn their teammates, allowing them to avoid the problem.

We will call as a teammate of robot i every robot within a maximum distance α that has a target in the same direction (d_i). In our algorithm, robots are able to send messages exclusively to their teammates. This can be easily implemented, for example, by putting d_i in every message and ignoring the messages that are supposed to be read by other groups.

We modeled our solution as a simple finite state machine. A robot starts in the *normal* mode, following Equation 1. Upon realization of a congestion risk, it changes its mode to *deviating*, and dynamically adapts its trajectory in order to avoid the congestion. When the risk has been successfully avoided, the robot returns to the *normal* mode.

We consider that a robot is able to detect the presence of another when the distance between them is lower than δ. Every time that a robot, i, detects the presence of another, j, it sends a message saying the direction of its target. If $d_i \neq d_j$, the robot that received a message, j, is able to perceive the imminent risk of congestion. In a similar way, j is also going to send a message to i informing its target direction, allowing both robots to notice this risk. In order to decrease the number of messages, each robot can send only one message informing its direction at every ϵ iterations. This initial phase of the algorithm can be seen in Figure 1(a).

The robots that noticed the risk of congestion send a message to their teammates, as can be seen in Figure 1(b). Each robot, upon receiving this message, relays it to its teammates, as shown in Figure 1(c). In this way, the information of a congestion risk is sent through the swarm and each group can deviate appropriately, as shown in Figure 1(d).

As soon as a robot notices the risk of congestion, whether it found a robot of the other group or received a warning from a teammate, it deviates from the local where the congestion could happen. To do this, the robot uses the direction of its target as a basis. It can be specified, for example, that each robot will deviate in the counterclockwise direction, therefore the group that goes from west to east will deviate to the south, while the group that goes from east to west will deviate to the north. The controller of a deviating robot, therefore, can be given by:

$$\mathbf{u}_i = k_1 \frac{g(\mathbf{ta}_i)}{||g(\mathbf{ta}_i)||} - k_2 \sum_{j \in N_i} \frac{1}{\mathbf{q}_j - \mathbf{q}_i} - k_3 \dot{\mathbf{q}}_i, \qquad (2)$$

where g is a vector that will guide the robot in the direction of the target, while at the same time will force it to deviate in the appropriate direction. In order to construct a model

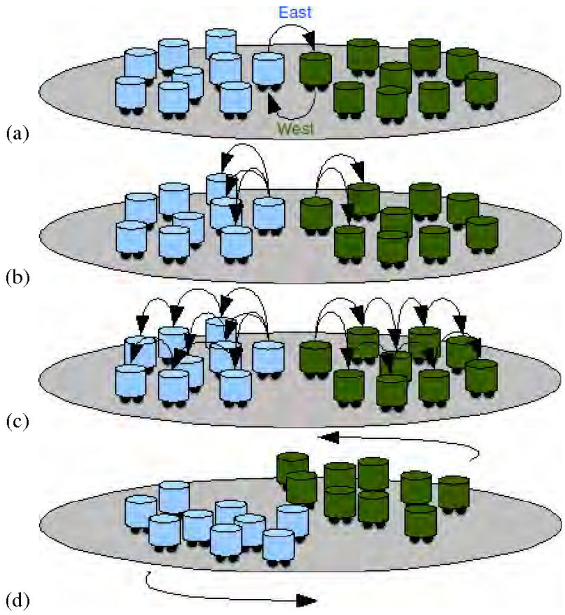


Fig. 1. Steps of the execution of the proposed coordination algorithm.

independent of global positions, we will state the equation of g in a coordinate system D with origin at the point where the robot started to deviate. To simplify the equation it will be considered that the robot moves in the x axis and it will deviate in the positive direction of the y axis. The equations for the other cases can be easily derived. The robot will move a certain distance in the y axis before returning to its regular controller, given in Equation 1. Therefore, function g can be given by:

$$g(\mathbf{ta}_i) = k_4(f(\mathbf{ta}_i)_x) + k_5(\mathbf{q}_{iy} - \phi), \qquad (3)$$

where k_4 and k_5 are positive constants with $k_4 < k_5$, ϕ is a positive value, which specifies the distance to be traveled in the y axis while deviating, and all vectors are expressed in the coordinate system D. When the distance effectively traveled in the y axis is close to ϕ, the robot returns to its normal controller, given by Equation 1. The value of ϕ must be estimated empirically or by an experimental evaluation, as will be discussed in Section IV-C. Note that it is not necessary to know the global position of the robot, only its relative distance with respect to the point where it started to deviate.

IV. Experiments

We ran a series of simulations and real experiments to study the performance and feasibility of the proposed algorithm. For the simulations, we used the Player/Stage framework [17], a well known framework for robotics programming and simulation. We simulated differential drive robots, based in the P2DX model, equipped with laser. The real experiments were performed using a dozen *e-puck robots*. The e-puck is a small-sized (7cm diameter) differential drive robot that is very suitable for swarm experimentation [18]. Each robot is equipped with a ring of 8 IR sensors that allows proximity sensing and a group of colored LEDs to indicate robot status. Local processing is performed by a dsPIC microprocessor and a bluetooth wireless interface allows robot to robot communication and remote control. Figure 2 shows the robots used in the experiment.

Fig. 2. Dozen *e-puck* robots used in the experiments.

In order to simulate differential drive robots, we adapted the controllers of Equations 1 and 2 to work with non-holonomic robots, following the approach presented in [19]. This same approach was used to control the e-pucks.

A. Simulations

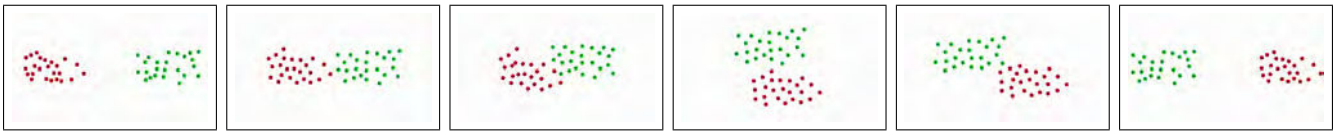
We ran simulations with two groups of 24 robots moving in opposite directions. In Figure 3 we can see the execution using only collision avoidance (NotCoord). In Figure 4, the result using the coordination algorithm (Coord) can be seen. As can be visually observed, the behavior of the robots is better using the proposed algorithm. The navigation is smoother, the groups are more cohesive and the risk of collisions is smaller. In the navigation using only collision avoidance (Figure 3), on the other hand, the groups mixed and the risk of collisions is high. We can see an agglomeration of robots in the central region, characterizing a congestion situation. We can also observe that while some robots are able to reach the target, others are still locked in the congestion region, wasting resources.

We also ran a series of simulations to evaluate the efficiency and the scalability of the algorithm. We varied the number of robots per group, and measured the execution time and the number of messages sent. For every fixed number of robots, we ran the simulation 20 times and computed the arithmetic mean of the result. The robots of a group were randomly positioned in a pre-specified area before every execution. As a measure of time, we used the number of iterations necessary for the last robot to overpass a fixed point in its direction of movement. We used the following values for the main constants: $\delta = 2m, \alpha = 3m, \epsilon = 50, k_1 = 2.5, k_2 = 0.8, k_3 = 10, k_4 = 0.1, k_5 = 1, \phi = 4m$.

In Figure 5 we can see the time used by both algorithms. As can be observed, the convergence time of the NotCoord algorithm increased monotonically when the number of robots got higher, while using the proposed algorithm, the number of iterations remained practically constant even for larger groups. Moreover, the Coord algorithm had a smaller convergence time than the NotCoord algorithm with gains of up to 40% for large groups. We executed a *t-test* that showed that the Coord algorithm was better in all analyzed points with 95% level of confidence. We also computed the standard deviation of the results, which showed that the proposed algorithm has a smaller deviation from the mean.

(a) (b) (c) (d) (e) (f)

Fig. 3. Execution with two groups using only local repulsion forces.



(a) (b) (c) (d) (e) (f)

Fig. 4. Execution with two groups using coordination algorithm.

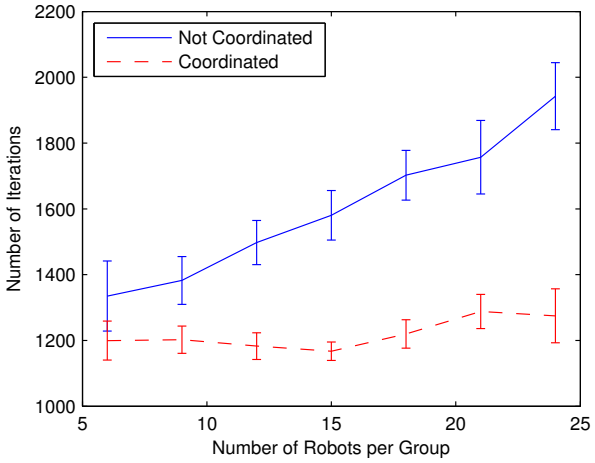


Fig. 5. Time used by both algorithms. The bars show the confidence interval of the results, with 95% level of confidence.

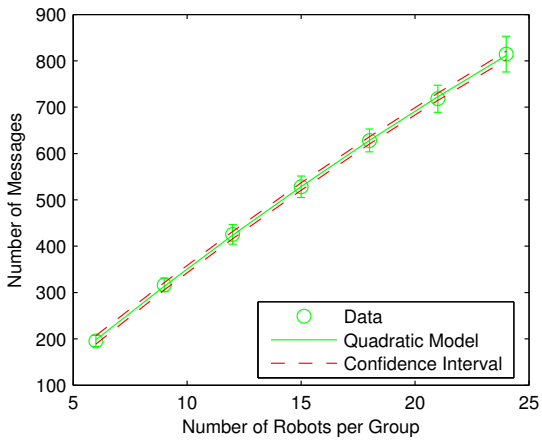


Fig. 6. Number of messages sent for a varying number of robots. The bars show the confidence interval of the results. Both the confidence interval of the points and of the regression correspond to a level of confidence of 95%.

In Figure 6 we can see the number of messages used by the proposed algorithm for a varying number of robots. The best linear model found was $y = 34.0875x + 6.7446$, with a coefficient of determination (R^2) of 0.9978; while the best quadratic model, shown in the figure, was $y = -0.2969x^2 + 42.9942x - 49.3679$, with $R^2 = 0.9998$. Although the best model was a quadratic function, we can see that the quadratic term was a small negative number. This result shows that the algorithm scales well and is suitable for large groups.

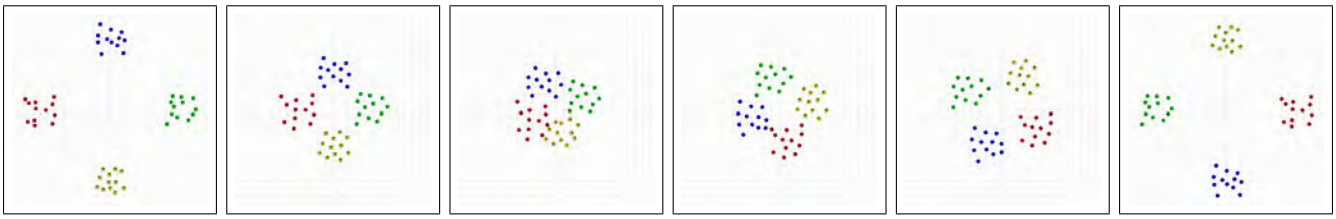
We also ran simulations with four groups of 12 robots. The result can be seen in Figure 7. As can be observed, the algorithm also worked well in that situation. The robots circulated around the region where a congestion could happen, and every group was able to reach the specified destination.

B. Real Robots

As mentioned, we also tested the proposed algorithm using twelve *e-puck* robots. These experiments are important to show the feasibility of the algorithm in real scenarios, with all the uncertainties caused by sensing and actuation errors, communication failures, etc.

To simplify the implementation, we used a localization system specifically designed for swarm localization in indoor environments [20], although, as mentioned, the algorithm does not depend on global localization. Also, as the IR sensors of the e-pucks have a very small range, we implemented a virtual sensor based on the localization system to detect neighbors.

A sequence of snapshots of an execution with two groups of robots can be seen in Figure 8, while with four groups can be seen in Figure 9 (a short video of the experiments is accompanying the paper). E-pucks with all LEDs on are in the *deviating* state. The graphs in the bottom depict the robots' position and states: robots following their normal controller, given by Equation 1 (+); robots following the deviation controller, given by Equation 2 (∘), and robots that, after deviating, returned to their normal controller (×). We used the following values for the main constants: $\delta = 0.3m, \alpha = 0.3m, \epsilon = 25, k_1 = 0.2, k_2 = 0.02, k_3 = 5, k_4 = 0.1, k_5 = 1$. In the two groups case, we used $\phi = 0.18m$, while in the four groups case we used $\phi = 0.25m$.



(a) (b) (c) (d) (e) (f)

Fig. 7. Execution with four groups using the coordination algorithm.

As can be seen, using the proposed algorithm the robots were able to complete the task in a smooth manner in both scenarios. In the two executions the total time was about 2 minutes. We also ran the same scenarios using only local repulsion forces, which needed about 4.5 minutes in the two groups case and 6 minutes in the four groups case. The convergence time gain was of 55% in the former case and 66% in the latter. Therefore, these proof of concept experiments indicate that the algorithm can work well to coordinate a swarm of robots, allowing them to efficiently navigate into opposite directions.

C. Choice of Parameters

It is important to discuss some aspects concerning the selection of parameters used in the algorithm. This might help designers that decide to try the proposed method. When robots coming from different directions realize the presence of each other, they must have enough time to deviate before they encounter. Therefore, the parameters of the system must be selected in order to facilitate this to happen.

The speed of the robots (given mainly by the constant k_1), must be adjusted considering the communication velocity, because robots that are fast have a higher inertia. The sensing and communication ranges (given by constants δ and α) also need to be selected considering the speed of the robots. The algorithm can work with groups having different velocities, but faster robots must have a greater sensing and communication range, so that slower ones will realize the congestion risk sooner and will have enough time to deviate. It is also possible to work with constants k_4 and k_5 to change the balance between deviating and reaching the specified destination. This can lead to a safer system, but with a performance cost. In the experiments we used k_5 an order of magnitude higher than k_4, as this configuration led to a good balance between the forces applied to the robots.

Another important parameter is the ϕ constant, that controls the height of the deviation trajectory. If it is low, one group will not be able to completely avoid the others. If it is high, the robots will move more than necessary, wasting time and resources. It is necessary to find a good compromise point, which can be done by experimental evaluation. In our simulations we realized that with four groups of robots it is better to use a ϕ constant slightly higher than with two groups, as it is a more difficult situation.

V. Conclusions

In this work, we proposed an algorithm to control the traffic of a swarm of robots, avoiding congestion situations when large groups of robots move in opposite directions. The proposed algorithm is based on a simple idea: robots that perceive the possibility of collision warn their teammates through local communication and the group changes its behavior to avoid this situation. In spite of being simple, the algorithm presented very good results in terms of performance and scalability for the studied scenarios. This happens because, as in nature, robots take advantage of being part of a group, and their collaborative work allows them to avoid the congestion earlier than they could using only their local sensing. We performed several simulations and real experiments which demonstrated that the proposed algorithm was successful in avoiding congestions and improving navigation efficiency. Moreover, the algorithm showed a tendency to scale well as the population increases, which is very important when dealing with swarms.

The situations presented in this paper are still very specific. There are still lot of work to do in order to generalize this algorithm for other situations. In this sense, this work is also important since it forms the basis for new and exciting future works. We would like to observe its behavior for a large number of groups or when groups do not have opposite directions, but encounter with different angles. Situations where the number of robots in each group is very different or have different velocities must also be investigated. It would also be interesting to explore the case where the groups of robots encounter in different time intervals. By investigating these situations and other related cases we can achieve a robust and efficient navigation system for a swarm of robots.

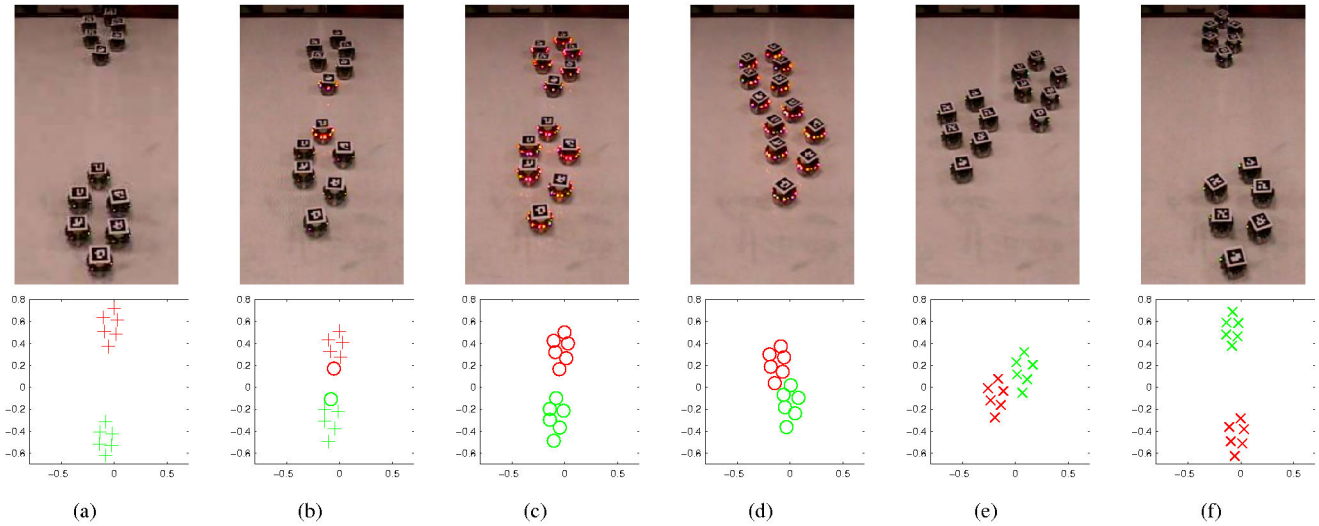
References

[1] L. S. Marcolino and L. Chaimowicz, "No robot left behind: Coordination to overcome local minima in swarm navigation," in *Proceedings of the 2008 IEEE International Conference on Robotics and Automation*, 2008, pp. 1904–1909.

[2] ——, "Traffic control for a swarm of robots: Avoiding target congestion," in *Proceedings of the 2009 IEEE International Conference on Intelligent Robots and Systems*, 2009.

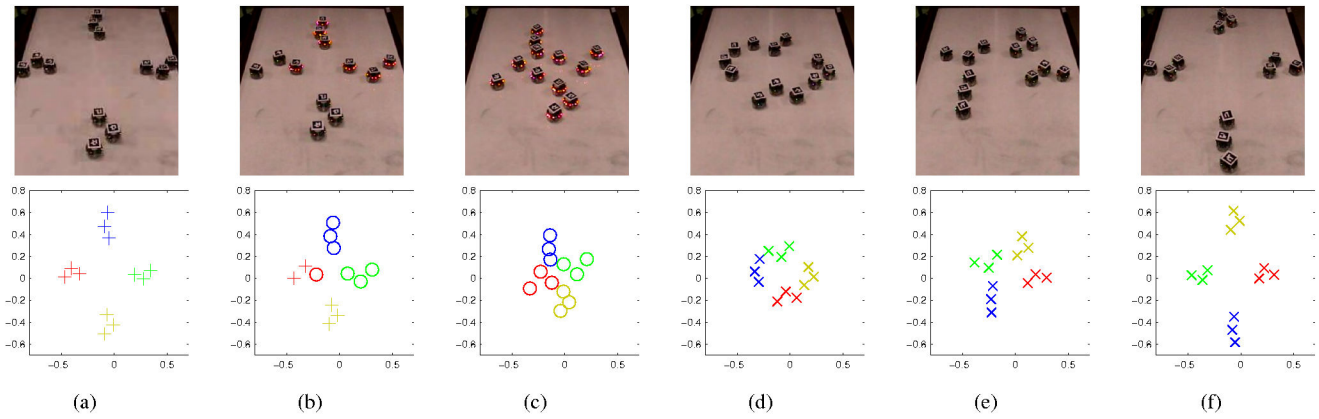
[3] R. Bachmayer and N. E. Leonard, "Vehicle networks for gradient descent in a sampled environment," in *Proceedings of the 41st IEEE Conference on Decision and Control (CDC-02)*, 2002, pp. 112–117.

[4] M. A. Hsieh, L. Chaimowicz, and V. Kumar, "Decentralized controllers for shape generation with robotic swarms," *Robotica*, vol. 26, pp. 691–701, September 2008.



(a) (b) (c) (d) (e) (f)

Fig. 8. Real execution with two groups using the coordination algorithm.



(a) (b) (c) (d) (e) (f)

Fig. 9. Real execution with four groups using the coordination algorithm.

[5] U. Y. Cao, A. S. Fukunaga, and A. B. Kahng, "Cooperative mobile robotics: Antecedents and directions," *Autonomous Robots*, vol. 4, no. 1, pp. 7–23, March 1997.

[6] D. Grossman, "Traffic control of multiple robot vehicles," *Journal of Robotics and Automation*, vol. 4, pp. 491–497, 1988.

[7] S. Kato, S. Nishiyama, and J. Takeno, "Coordinating mobile robots by applying traffic rules," in *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, 1992.

[8] K. Dresner and P. Stone, "Multiagent traffic management: an improved intersection control mechanism," in *AAMAS '05: Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*. New York, NY, USA: ACM, 2005, pp. 471–477.

[9] K. Viswanath and K. M. Krishna, "Sensor network mediated multi robotic traffic control in indoor environments," in *Proceedings of the International Conference on Advanced Robotics*, 1997.

[10] Y. Ikemoto, Y. Hasegawa, T. Fukuda, and K. Matsuda, "Zipping, weaving: Control of vehicle group behavior in non-signalized intersection," in *Proceedings of the 2004 IEEE International Conference on Robotics and Automation*, New Orleans, USA, 2004, pp. 4387–4391.

[11] A. Treuille, S. Cooper, and Z. Popović, "Continuum crowds," in *SIGGRAPH '06: ACM SIGGRAPH 2006 Papers*. New York, NY, USA: ACM, 2006, pp. 1160–1168.

[12] K. M. Krishna and H. Hexmoor, "Reactive collision avoidance of multiple moving agents by cooperation and conflict propagation," in *Proceedings of the 2004 IEEE International Conference on Robotics and Automation*, 2004, pp. 2141–2146.

[13] J. S. J. H. Penders, "Spatial conflicts among robot agents," in *Proceedings of the AAAI'99 Workshop on Agents' Conflicts*, Orlando, 1999.

[14] M. Jger and B. Nabel, "Decentralized collision avoidance, deadlock detection, and deadlock resolution for multiple mobile robots," in *Proceedings of the 2001 IEEE International Conference on Intelligent Robots and Systems, IROS 2001*, Maui, USA, 2001, pp. 1213–1219.

[15] A. Yasuaki and M. Yoshiki, "Collision avoidance method for multiple autonomous mobile agents by implicit cooperation," in *Proceedings of the 2001 IEEE International Conference on Intelligent Robots and Systems, IROS 2001*, Maui, USA, 2001, pp. 1207–1212.

[16] C. Cai, C. Yang, Q. Zhu, and Y. Liang, "Collision avoidance in multi-robot systems," in *Proceedings of the 2007 IEEE International Conference on Mechatronics and Automation*, Harbin, China, 2007, pp. 2795–2800.

[17] B. Gerkey, R. T. Vaughan, and A. Howard., "The player/stage project: Tools for multi-robot and distributed sensor systems." in *Proceedings of the 11th International Conference on Advanced Robotics*, Coimbra, Portugal, June 2003, pp. 317–323.

[18] C. M. Cianci, X. Raemy, J. Pugh, and A. Martinoli, "Communication in a Swarm of Miniature Robots: The e-Puck as an Educational Tool for Swarm Robotics," in *Simulation of Adaptive Behavior (SAB-2006), Swarm Robotics Workshop*, ser. Lecture Notes in Computer Science (LNCS), 2007, pp. 103–115.

[19] A. De Luca and G. Oriolo, "Local incremental planning for nonholonomic mobile robots," in *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, 1994.

[20] R. Garcia, P. Shiroma, L. Chaimowicz, and M. Campos, "A framework for swarm localization," in *Proceedings of VIII SBAI - Brazilian Symposium on Intelligent Automation*, October 2007, in Portuguese.