Remote pursuer-evader experiments with mobile robots in the Automatic Control Telelab

Marco Casini, Andrea Garulli, Antonio Giannitrapani, Antonio Vicino

Dipartimento di Ingegneria dell'Informazione e Scienze Matematiche
Università di Siena
Via Roma, 56, 53100 Siena, Italy
E-mail: {casini,garulli,giannitrapani,vicino}@ing.unisi.it

Abstract: The use of remote labs for teaching control systems and robotics is gaining increasing attention in the control community. One of the key advantages of remote labs is the possibility to easily include new real experiments within the existing setup and share them with students through the Internet. In this paper, a new feature of the mobile robot setup embedded in the Automatic Control Telelab is presented. A pursuer-evader setting has been developed, involving two mobile robots based on the LEGO Mindstorms NXT Technology. Students can choose which role to play (pursuer or evader) and design the best control strategy to achieve the desired objective. Several experiments are reported to demonstrate the flexibility of the proposed environment.

Keywords: remote labs, mobile robots, pursuer-evader games

1. INTRODUCTION

In recent years, remote laboratories are becoming increasingly popular among educational professionals of various disciplines. Teaching subjects like automatic control systems and mobile robotics is expected to highly benefit from the adoption of ICT tools in terms of cost and time savings. As a matter of fact, remote labs freely accessible through the Internet allow students to experiment with real, possibly expensive, physical processes from anywhere in the world. At the same time, they provide a valuable support to instructors who can arrange real experiments with minimal effort, also when dealing with large classes (Dormido [2004], Gomes and Bogosyan [2009]).

A distinctive feature of remote labs, when compared to other technological teaching aids like simulation environments or virtual laboratories, is the preservation of the interaction with real experiments. On the one hand, this stimulates the interest of students, who can see the evolution of real devices during the experiment thanks to live video streaming usually provided by these kind of labs. On the other hand, experimenting with real processes is much more challenging and instructive, since one has to face a number of real-world issues which are often neglected in simulation. The latter is of paramount importance when dealing with mobile robots or autonomous agents. From low-level control laws to high-level motion coordination algorithms, experimental validation on real robots is crucial to assess their effectiveness in real operating conditions. To this purpose, a number of experimental setups for teaching and research in the mobile robotics field have been recently proposed, see e.g. Marshall et al. [2006], Mastellone et al. [2008], Taskin and Chernova [2013].

An interesting problem involving a group of autonomous agents is the so called pursuer-evader (PE) game. In its most basic setting, an agent (pursuer) has to catch an escaping agent (evader). Depending on the domain, PE algorithms have been proposed for agents moving over discrete graphs (Parsons [1976], Bopardikar et al. [2008]) or robots moving in a continuous space (Lim et al. [2004], Gerkey et al. [2006], Stipanović et al. [2010]).

In this paper, a remote lab suitable for PE experiments with mobile robots is presented. Intuitive graphical interfaces allow users to specify a number of experiment parameters and to visualize the experiment evolution online. Predefined algorithms can be selected or user-defined PE strategies can be tested, by uploading a suitably formatted Matlab function. The resulting framework features high versatility, while at the same time being extremely easy to use. This work extends previous experimental setups focused on testing motion coordination strategies for a team of robots (Casini et al. [2011]) and obstacle avoidance algorithms (Casini et al. [2012]). The developed setup is worldwide freely accessible through the Automatic Control Telelab, a remote lab developed at the University of Siena (Casini et al. [2004]).

The paper is organized as follows. In Section 2 a brief overview of the hardware and software architecture of the proposed experimental setup is presented. Section 3 starting from basic works on PE games, highlights some peculiarities of the proposed setup which affect the admissible PE strategies. User interaction through graphical interfaces is described in Section 4. Some experiments performed by high-school and undergraduate students are reported in Section 5, whereas some conclusions and future developments are drawn in Section 6.

Fig. 1. LEGO Mindstorms mobile robot.

2. SETUP DESCRIPTION

The Automatic Control Telelab (ACT) of the University of Siena (`http://act.dii.unisi.it`) is a remote lab for teaching purposes in the field of control systems, system identification and mobile robotics. Currently, it offers five different physical processes available for remote experiments, ranging from a simple DC motor to a 2 DoF helicopter model. Through a web server, users can select a predefined experiment or test ad-hoc control laws. A web camera constantly provides visual feedback of the ongoing experiment and at the end all relevant data can be downloaded for offline analysis.

Recently, the ACT has been enriched with a team of four mobile robots, built with the LEGO Mindstorms technology (The LEGO Group [2009]). The robots (see Figure 1) have a differential drive, with a PI controller running onboard the NXT brick to control wheel speeds. A pair of wide angle cameras mounted on the lab ceiling are used to extract in real-time the pose of each robot, thanks to the markers placed to the top of each vehicle (see Figure 1). A server running on a desktop PC constantly gathers the robot poses, computes the desired vehicle speeds on the basis of the control laws to be tested and send them to the robots through a bluetooth connection. A high fidelity Matlab simulator of the robot team is also available, speeding up the design of the control laws. This way, users can design the control strategy offline, without even connecting to the ACT, and then validate their algorithms on the real robots. An additional tool, extremely useful for the analysis of a carried out experiment, is the so called "player", which allows one to graphically reproduce the evolution of the robots during an experiment from the downloaded data. Further details on the hardware and software architecture can be found in Casini et al. [2011].

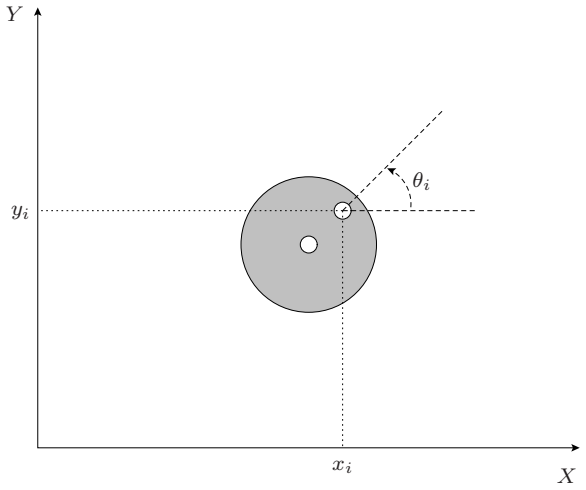

Fig. 2. The robot pose.

3. PURSUER-EVADER GAMES

The PE problem was first introduced in the context of differential game theory with the objective of determining necessary conditions for two moving agents to collide (Isaac [1965]). Since then, a number of different versions of the problem have been addressed. For example, in Parsons [1976] the agents are supposed to move in a discrete space so that the problem can be cast as a search on a graph. Issues arising from sensing limitations of the agents are considered e.g. in Bopardikar et al. [2008]. When dealing with mobile robots, the motion domain is naturally continuous and several additional aspects can be taken into account. In Lim et al. [2004], the kinematic constraints which often limit the maneuverability of real vehicles are considered. In Gerkey et al. [2006], a PE strategy for agents equipped with sensors featuring limited field of view is presented. In Stipanović et al. [2010], guaranteed strategies ensuring capture (or evasion) for agents with nonlinear dynamics are studied.

In order to exploit the developed experimental setup for designing and testing PE strategies, the following observations are in order. The robots feature a unicycle-like kinematics. Denoting by $p_i(t) = [x_i(t)\ y_i(t)\ \theta_i(t)]'$ the position and orientation of the i-th robot at time t (see Figure 2), then the robot pose evolves according to the model

$$\begin{aligned} \dot{x}_i(t) &= v_i(t)\cos(\theta_i(t)), \\ \dot{y}_i(t) &= v_i(t)\sin(\theta_i(t)), \qquad i = 1, \ldots, N, \\ \dot{\theta}_i(t) &= \omega_i(t), \end{aligned}$$

where v_i and ω_i are the linear and angular speed, respectively. The above motion model is quite common in practice and suitable for the control laws proposed in Lim et al. [2004], Stipanović et al. [2010]. Conversely, the boundedness of the available free space poses some limitations on the feasible PE strategies. For example, in this case it seems reasonable to assume that the evader is faster than the pursuer, which is the opposite of what is usually done when moving in free space like in Lim et al. [2004].

4. USER INTERFACE

This section describes the graphical user interfaces (GUIs) which allow a user to perform remote pursuer-evader games. Through the "control interface" (Fig. 3), users may choose the role to play (pursuer, evader or both), the difficulty level, and the experiment maximum duration. At this stage, the Matlab function containing the implemented algorithm must be uploaded. The input parameters of the Matlab function are the pose of each robot, the experiment time and the sampling time, while the output parameters are the linear and angular speed of the chosen player. The algorithm of the opponent is predefined, and it is automatically chosen by the system depending on the role selected by the user. In case students want to test simultaneously pursuer and evader algorithms, they have to upload a Matlab function containing both algorithms. In all cases, suitable templates are provided to help the implementation of the designed algorithms.

It is worthwhile to notice that the behaviour of the designed algorithm can be preliminary checked in simulation by means of the provided simulator. Although the actual result may differ from the simulated one due to disturbances, unmodeled dynamics, etc., students are encouraged to test their file before uploading it, to verify that the syntax is correct and the robot behaviour is acceptable.

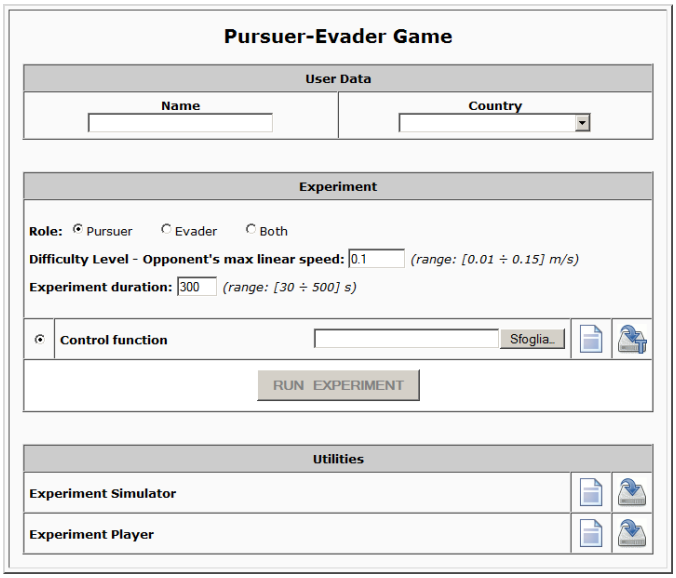

Fig. 3. The "control interface".

Once the Matlab file containing the algorithm has been uploaded, a check on the correctness of the user function is done, If no errors are detected, the "PEG interface" opens (Fig. 4). This interface has been developed as a Java applet and allows one to start/stop the remote experiment and to observe the robot trajectories. A real-time graphical representation, numerical values and online camera allow students to increase the sense-of-presence in the lab.

The experiment ends either when the pursuer "catches" the evader (pursuer wins) or when the time is out (evader wins). In both cases, data containing the full information about the experiment can be downloaded to perform offline analysis.

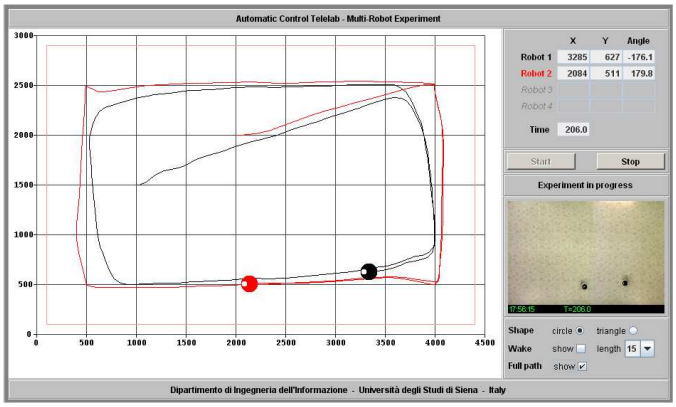

Fig. 4. The "PEG interface".

```
function [Vel]=ACT_Pursuer(Time,Pose,Ts)
   v_max=0.075; % max linear speed (m/s)
   w_max=1;     % max angular speed (rad/s)
   dX=Pose(1,2)-Pose(1,1);
   dY=Pose(2,2)-Pose(2,1);
   phi=Pose(3,1);       % actual heading
   phides=atan2(dY,dX); % desired heading
   V=v_max;
   W=w_max*(wrapToPi(phides-phi));
   if abs(W)>w_max, W=w_max*sign(W); end
   Vel=[V ; W];  % linear and angular speed
end
```

Fig. 5. Example 1. The Matlab function implementing the pursuer algorithm.

5. EXPERIMENTAL TESTS

The pursuer-evader game is a framework which allows one to create several educational scenarios suited for students ranging from high-school to graduate and post-graduate. In fact, depending on the given experiment and the chosen role, one can devise several tasks with different complexity; moreover, for a given task, the difficulty can be adjusted by varying the speed ratio between players. In the following, two examples with increasing complexity are reported to show the potential of the proposed facility as a tool for control education on mobile robotics.

5.1 Example 1.

It was asked to three groups of high-school students to design an evader algorithm able to survive a pursuer. The pursuer algorithm was designed to point towards the evader at each time step, by setting the linear velocity to its maximum (see Fig. 5 for the Matlab code). To facilitate the task, a predefined function able to move the robot into a desired pose was available to students. So, their algorithm had to compute as output a pose reference instead of a velocity reference.

The algorithm designed by the three groups which best performed in terms of rate of success, forces the evader to move to four places near the vertexes of the workspace, depending on the actual position of the pursuer at a given time step. It is not allowed to move towards non adjacent vertexes (Fig. 6). Despite the simplicity of the proposed strategy, this algorithm turns out to work well, and it is able to let the evader escape when the speed of the pursuer

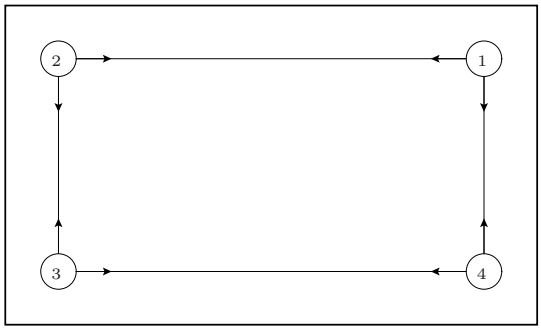

Fig. 6. Example 1. The evader strategy implemented by students.

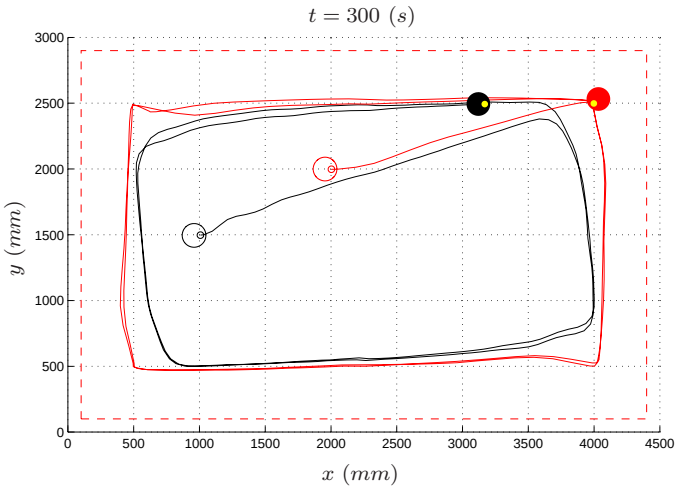

Fig. 7. Example 1. Pursuer (black), evader (red), speed ratio 50%.

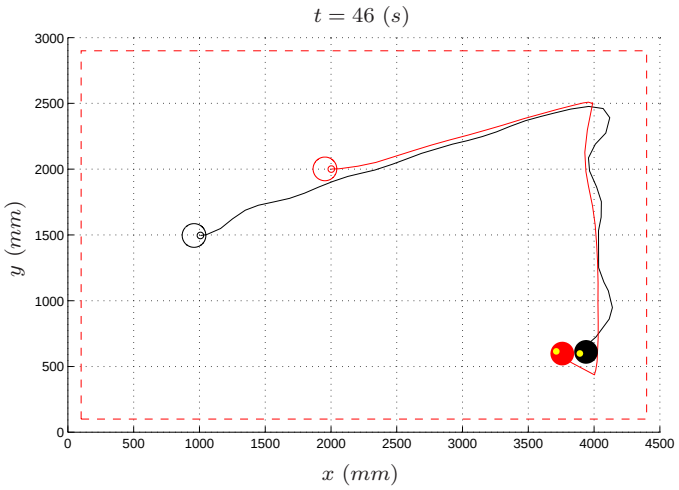

Fig. 8. Example 1. Pursuer (black), evader (red), speed ratio 80%.

is less than 70% with respect to the speed of the evader. The length of the experiment is fixed to 300 seconds. In Fig. 7 and 8, the path of the two vehicles are reported for a speed ratio of 50% (evader wins) and 80% (pursuer wins). In these figures, empty and filled circles denote the initial and final poses of the robots, respectively. Robot paths are denoted by solid lines.

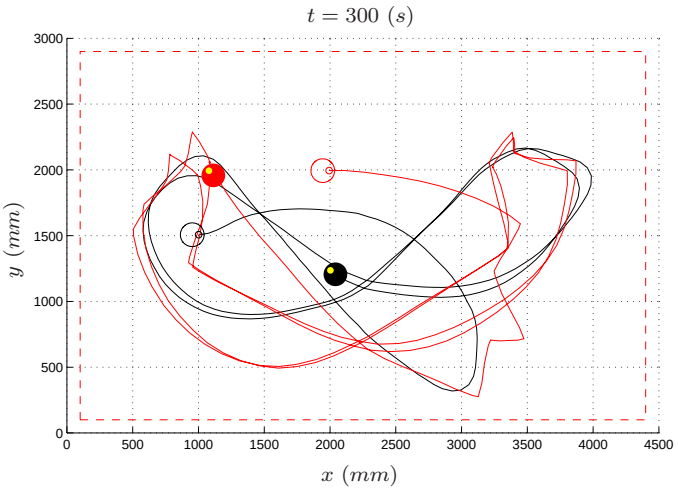

Fig. 9. Example 2. Pursuer (black), evader (red), speed ratio 50%.

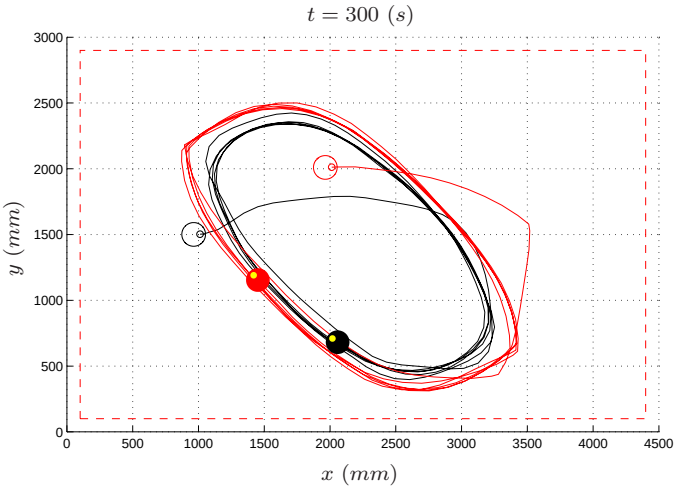

Fig. 10. Example 2. Pursuer (black), evader (red), speed ratio 80%.

5.2 Example 2.

In this example, a more sophisticated control strategy is used for the evader. The algorithm of the pursuer is the same as in Example 1, while the evader strategy has been designed by an undergraduate student and it is based on the potential field method (Khatib [1986]), which tries to drive the robot according to an artificial repulsive potential field generated by the pursuer and the workspace boundaries.

Figures 9 and 10 depict the path of two experiments where the speed ratio is set to 50% and 80%, respectively. Different from Example 1, in both cases the evader is able to escape the pursuer. Additional experiments show that, under the same initial conditions, the pursuer can catch the evader only if the speed ratio is at least 92%.

5.3 Comparison with simulations

From an educational point of view, it is interesting to compare the real experiments to the results obtained through the provided simulator. By comparing the simulation (Fig. 11) and the actual experiment (Fig. 7) of Example 1 (speed ratio 50%), one can state that the

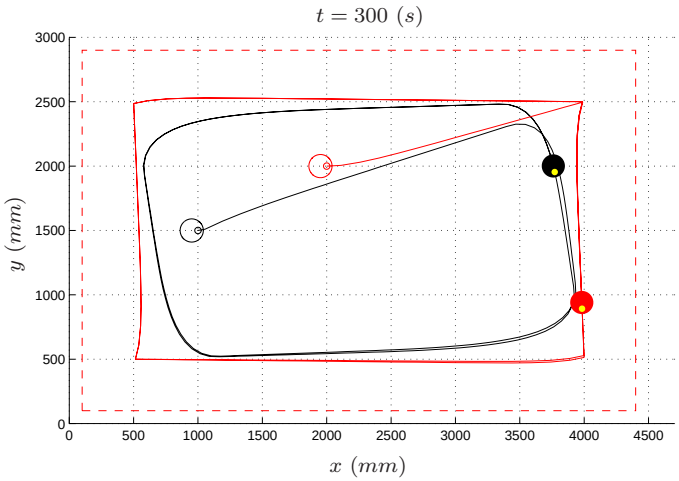

Fig. 11. Simulated experiment for Example 1 with speed ratio 50%.

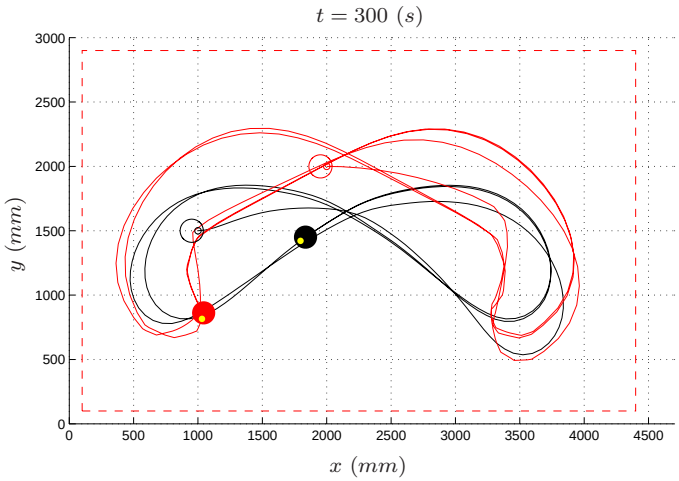

Fig. 12. Simulated experiment for Example 2 with speed ratio 50%.

two trajectories are similar, and the virtual (simulated) experiment provides a good approximation of reality.

A different situation arises by comparing the simulation (Fig. 12) and the real experiment (Fig. 9) of Example 2 (speed ratio 50%). In this case, due to the complex behaviour of the robot algorithms, the two trajectories differ significantly. This is a typical situation where a virtual experiment is not able to reproduce a real one. In this case, student experiences on real devices result to be very instructive.

6. CONCLUSIONS

Pursuer-evader games are an intriguing topic which involves a number of key concepts in control and robotics, ranging from mobile robot trajectory design to optimal control and game theory. Embedding pursuer-evader experiments in the Automatic Control Telelab has proven to be stimulating for students, who can easily devise and test strategies for both players, not only in simulation but also on real robots based on the LEGO Mindstorms NXT technology. The proposed setup is interesting also for researchers, who are able to evaluate and compare control laws available in the literature, in a realistic scenario in which the limitations arising from the mobile robot sensors and actuators and from the finite dimension of the environment play a role and must be taken into account. The extension of the proposed framework to games including structured environments (e.g., workspaces with virtual obstacles), and multiple pursuers and evaders is under development. A further extension of this lab will involve the student competition facility reported in Casini et al. [2005]. In fact, purser-evader games are perfect for creating student challenges; here, students may compete in a "pursuer vs evader" game, design pursuer algorithms able to catch a predefined evader in the least amount of time, or may devise evader strategies able to survive a predefined pursuer as long as possible.

REFERENCES

S.D. Bopardikar, F. Bullo, and J.P. Hespanha. On discrete-time pursuit-evasion games with sensing limitations. *IEEE Transactions on Robotics*, 24(6):1429–1439, 2008.

M. Casini, D. Prattichizzo, and A. Vicino. The Automatic Control Telelab: A web-based technology for distance learning. *IEEE Control Systems Magazine*, 24(3):36–44, 2004.

M. Casini, D. Prattichizzo, and A. Vicino. A student control competition through a remote robotics lab. *IEEE Control Systems Magazine*, 25(1):56–59, 2005.

M. Casini, A. Garulli, A. Giannitrapani, and A. Vicino. A LEGO Mindstorms multi-robot setup in the Automatic Control Telelab. In *Proc. of the 18th IFAC World Congress*, pages 9812–9817, Milan, Italy, 2011.

M. Casini, A. Garulli, A. Giannitrapani, and A. Vicino. A remote lab for multi-robot experiments with virtual obstacles. In *Proc. of the 9th IFAC Symposium on Advances in Control Education*, pages 354–359, Nizhny Novgorod, Russia, 2012.

S. Dormido. Control learning: present and future. *Annual Reviews in Control*, 28:115–136, 2004.

B.P. Gerkey, S. Thrun, and G. Gordon. Visibility-based pursuit-evasion with limited field of view. *The International Journal of Robotics Research*, 25(4):299–315, 2006.

L. Gomes and S. Bogosyan. Current trends in remote laboratories. *IEEE Transactions on Industrial Electronics*, 56(12):4744–4756, 2009.

R. Isaac. *Differential games: A mathematical theory with applications warfare and pursuit, control and optimization*. John Wiley and Sons, Inc., New York, 1965.

O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *The International Journal of Robotics Research*, 5(1):90, 1986.

S.H. Lim, T. Furukawa, G. Dissanayake, and H.F. Durrant-Whyte. A time-optimal control strategy for pursuit-evasion games problems. In *Proc. of the 2004 IEEE International Conference on Robotics and Automation*, pages 3962–3967, New Orleans, LA, USA, 2004.

J.A. Marshall, T. Fung, M.E. Broucke, G.M.T. D'Eleuterio, and B.A. Francis. Experiments in multirobot coordination. *Robotics and Autonomous Systems*, 54(3):265–275, 2006.

S. Mastellone, D.M. Stipanovic, C.R. Graunke, K.A. Intlekofer, and M.W. Spong. Formation control and collision avoidance for multi-agent non-holonomic systems:

Theory and experiments. *The International Journal of Robotics Research*, 27(1):107–126, 2008.

T Parsons. Pursuit-evasion in a graph. In Y. Alavi and D. Lick, editors, *Theory and applications of graphs*, pages 426–441. Springer-Verlag, 1976.

D.M. Stipanović, A. Melikyan, and N. Hovakimyan. Guaranteed strategies for nonlinear multi-player pursuit-evasion games. *International Game Theory Review*, 12 (01):1–17, 2010.

P. Taskin and S. Chernova, editors. *Special issue on robotics education*, volume 56(1) of *IEEE Transactions on education*, 2013.

The LEGO Group. The Lego Mindstorms home page, 2009. [Online]. Available: http://mindstorms.lego.com.