# ISocRob — Intelligent Society of Robots*

Rodrigo Ventura, Pedro Aparício, Carlos Marques, Pedro Lima, and Luís Custódio

Instituto de Sistemas e Robótica/Instituto Superior Técnico (ISR/IST),
Av. Rovisco Pais – 1, 1049-001 Lisboa
PORTUGAL
{yoda,aparicio,pal,lmmc}@isr.ist.utl.pt

**Abstract.** The SocRob project was born as a challenge for multidisciplinary research on broad and generic approaches for the design of a cooperative robot society, involving Control, Robotics and Artificial Intelligence researchers. In this paper the basic aspects of last year implementation as well as the improvements made meanwhile are briefly recalled and presented. Naturally, a special emphasis is given here to the novel solutions proposed for this year implementation, the results obtained and the expected future developments.

## 1 Introduction

The Artificial Intelligence and the Intelligent Control groups of the ISR/IST have started almost two years ago a joint project on Cooperative Robotics, denominated SocRob, to foster research on methodologies for the definition of functional, hardware and software architectures to support intelligent autonomous behavior and evaluate performance of a group of *real* robots, either as a society and as individuals.

The utilization of real robotic agents to perform on real environments, for instance, a robotic soccer game, raises several new questions and perspectives that turn the development of a multi-agent system a much more difficult and challenging problem [5].

The robots used by the ISocRob team were developed from scratch, so that both conceptual and implementation issues were considered [1]. For the 1999 competition, some adjustments and improvements were made both on hardware and software components: new robot wheels, a kicker device, development of a self-localization system, a friendly man-machine interface, and a new software framework based on the multi-agent system paradigm.

## 2 Hardware and Software Description

Each robot hardware is divided in four main blocks: sensors, main processing unit, actuators and communications. Currently, from the hardware architecture standpoint, the population is composed of homogeneous mobile robots.

* This work has been supported by a grant from the Fundação Calouste Gulbenkian and the Portuguese Foundation for Science and Technology (ISR/IST programmatic funding).

### 2.1 Hardware

The sensors and actuators available in each robot are those mentioned in last year's team report [1]. In terms of sensors, the novelty for this year was the development of a self-localization system (a pose sensor), but unfortunately this sensor was not available at the time of RoboCup'99. In terms of actuators, it has been developed a new kicking device similar for all robots.

**Pose sensor** Depending on the type of application involved, each robot of the society needs to regularly update its current pose (position and orientation) with respect to a reference frame (e.g., located in the field center). This may be accomplished based on the triangulation principle using for instance a *convex mirror for full scene image* system based on a vision camera and a mirror with a special geometry. Since the RoboCup environment has available a sufficient number of visual landmarks, the SocRob project team decided to experiment the "mirror" solution. The idea is to allow robots, using only one vision camera, to acquire images from the mirror, appropriately positioned above the robot, in order to obtain a global view of the environment. If images are sufficiently broad to include three different and static beacons (e.g., goal plus two field corners), robots may apply the triangulation principle to determine their position.

**Kicker** The kicking ability enables soccer players to move the ball into places that otherwise would not be accessible. The kicker device is divided in two main parts: electronic and mechanical structures. The kicker electronics is composed of a micro-controller, an IR beam circuit and a power actuator. The micro-controller runs the control program and generates a signal modulation to be used in the IR beam. This signal consists of a square wave, rated 40kHz that is fed to the amplifier powering the IR led's. The 40KHz detector output is also directly connected to the controller. If an object is obstructing the beam the demodulator delivers a 0V constant signal, otherwise should a 40kHz IR beam be received, a 5V constant signal is obtained. The controller output is connected to the circuitry that drives the servo-motor. This solution can be seen as an instinctive reaction when the robot senses the ball. However this behavior can be disabled by the processor unit in order the robot to perform different type of actions. The kicker mechanics is based on a automobile door opening servo-motor, which when powered with opposite polarities moves a piston in opposite directions. The piston course is approximately 3cm.

### 2.2 Software

In what concerns software components, the two main improvements made this year were the development of a man-machine interface and a multi-agent software framework for programming each one of the robots. Instead of having a set of procedures to implement each of tasks needed (*e.g.*, motors control, image processing, communications, behaviour control, etc.), it was implemented a new software framework based on the multi-agent paradigm. For each task it is created a specific (micro-)agent[1] prepared to perform the task. All micro-agents are implemented using the concept of *thread* for multitasking programming and communicate with each other through a common, but distributed, repository of information (blackboard).

The man-machine interface allows to observe the behaviour of all robots using telemetric information (encoder data), images acquired from the frontal camera and some relevant parametric information. The interface permits also to manually control the robots through a *space mouse* device or directly from the keyboard, change some robot internal variables and specify certain game parameters.

## 3 Functional Architecture

From a functional standpoint, the whole robot society is composed of functionally heterogeneous robots. The functional architecture is *scalable* regarding the number of robots (or *agents*) involved. This means that, when a new robot joins the society, no changes have to be made to the overall system. The functional architecture establishes three levels: an organizational level dealing with the issues common to the whole society, a relational level where groups of agents cooperate/negotiate in order to establish a mutual agreement(commitment) concerning the execution of a particular action or the achievement of some objective, and an individual level encompassing all the available behaviors of each robot. A behavior is a set of purposive primitive tasks sequentially and/or concurrently executed. These primitive tasks consist of *sense-think-act loops*, a generalization of a closed loop control system which may include motor control, ball tracking, ball following, etc. For this year participation in RoboCup, some modifications were made on the relational and individual levels, especially in what concerns cooperation among robots [2].

Individual behaviors can be temporarily modified to allow cooperative relations between teammates. The negotiation implemented concerns two Forward players who actively try to get the ball. If two or more Forward players see the ball, a communication protocol is used by all players involved in order to determine which player is closest the ball. So, all players broadcast the estimation of its distance to the ball, and after that negotiate which player will follow the ball and which should return to a pre-defined location near one of the goals.

## 4 Agent-based Programming Language

The idea beyond the development of a programming language specially adequate for implementing multi-agent systems follows the work previously done by some members of our team — an agent-based programming language called RUBA [4]. The goal is to have a way for defining agents' architecture, creating agents, establishing communication links among agents, specifying cooperation mechanisms (based on a particular teamwork model), creating and deleting temporary sub-groups, and removing agents.

[1] Since this agent implements a primitive task running on a robot, it is called here a micro-agent.

The initial version of the computational model for the language consists of two classes of objects: *agents* and *blackboards*. A blackboard is the basic communication medium among agents, either to communicate among themselves, or between them and the external world. In what concerns RUBA, the current language specifications propose several improvements: extension of the blackboard for a distributed system, efficient blackboard indexing using a hierarchical name-space, and event-driven programming.

Conceptually, a blackboard is a centralized repository of data. The idea of a distributed blackboard is to distribute the information (data) among the agents. Practically, a blackboard is a mapping of symbols (hierarchically organized in nested name-spaces, *e.g.* `robot0.sensors.collision.2`) to variables. This scheme is supposed to uniformly implement several and different processes, such as message passing, shared memory, distributed data and local variables. A blackboard is implemented with an hash table of names to variables. Each variable has a set of attributes, such as scope, location, policy, type and lock. Also, there are a set of primitives to access the variable: `read`, `write`, `hook` and `lambda` [3].

## 5 Conclusions and Future Work

Currently, our robots are capable of simple but essential behaviors, composed of primitive tasks, such as following a ball, kicking a ball, scoring goals and defending the goal, using vision-based sensors and the other available sensors. Our current and future work is centered on concluding the development of the self-localization system based on a vision camera and a mirror, updating and tuning of the low-level software, design and implementation of an agent-based programming language suitable for multi-agent systems, study and development of a teamwork model and its integration with our functional architecture.

## References


1. P. Aparicio, R. Ventura, P. Lima, and C. Pinto-Ferreira. *ISocRob - Team Description*, chapter In Minoru Asada and Hiroaki Kitano, editors, RoboCup-98: Robot Soccer World Cup II. Springer-Verlag, Berlin, 1999.
2. P. Lima, R. Ventura, P. Aparício, and L. Custódio. *A Functional Architecture for a Team of Fully Autonomous Cooperative Robots*, chapter in this book. Springer-Verlag, Berlin, 1999.
3. R. Ventura, P. Aparício, and P. Lima. Agent-based programming language for multi-agent teams. Technical Report RT-701-99, RT-401-99, Instituto de Sistemas e Robótica, IST-Torre Norte, March 1999.
4. Rodrigo M. M. Ventura and Carlos A. Pinto-Ferreira. Problem solving without search. In Robert Trappl, editor, *Cybernetics and Systems '98*, pages 743–748. Austrian Society for Cybernetic Studies, 1998. Proceedings of EMCSR-98, Vienna, Austria.
5. Alex S. Fukunaga Y. Uny Cao, Andrew B. Kahng, and Frank Meng. Cooperative Mobile Robotics: Antecedents and Directions. In *http://www.cs.ucla.edu:8001/-Dienst/UI/2.0/Describe/ncstrl.ucla_cs%2f950049?abstract=Cooperation*, December 1995.