

# Project SMURFS – A Society of Multiple Robots

David Leal Martínez, Jürgen Leitner

SpaceMaster Robotics Team

<http://smrt.name/>

david.leal@gmail.com, juxi.leitner@gmail.com

## Abstract

A reconfigurable robot society built from scratch with LEGO Mindstorms was presented at the IJCAI 2009. This paper describes the robots as well as the algorithms developed over the last few months. A leader selection as well as a formation for area coverage algorithm were tested and presented at the conference. For localization a birds-eye-view camera system based on reacTIVision fiducial markers is used.

## Introduction and Background

This paper describes the project SMURFS, which was our contribution to *Robotics Workshop* and the *Eighteenth Annual Robot Exhibition* at the *International Joint Conference (IJCAI) 2009* in Pasadena, CA.

The project uses a reconfigurable robot society built from scratch with LEGO Mindstorms over the last 7 months. Separately a cooperation algorithm, in its simplest form, controlling the robots formations, was implemented and tested with a simulator, also developed over the last few months. A machine learning approach and a vector-based approach were evaluated in the simulator and the vector-based approach was then also tested with the reconfigurable robots.

## Reconfigurable Robots

Reconfigurable multi-robot societies is a young area of robotics that promises versatility, robustness and low cost as it relies on a society of multiple robots that will be able to perform an immeasurable amount of different tasks, tasks that not even were thought of at design time.

In order to create a new reconfigurable multi robot society from scratch that could be developed fast and with low costs, it was opted to create a prototype from LEGO Mindstorms NXT equipment, complemented with some extra electronics developed to expand the capabilities of the existing LEGO Mindstorms NXT system.

Reconfigurable robot systems will have a great impact, especially in the field of space exploration, where the

lower mass needs and the flexibility to use the system in many different ways (even not yet foreseen tasks) will be appreciated.

## Cooperative Behavior

Cooperation is not just discussed in robotics but also in political and other human sciences. Robot cooperation takes ideas from biology and tries to extend the behavior-based control to get simple cooperation (Arai et al., 2002).

The use of cooperative robotics reaches technological constraints, even more than regular robotics, because of the need to cope with multiple, autonomous entities. At the same time it is highly inter-disciplinary and draws influences from many other fields of research, e.g. artificial intelligence, as well as biology.

Examples of cooperating in nature (e.g. bees and ants) show possibilities for simple robots to work together to solve a very complex task. In contrast to the low level control of the robot (e.g. motion), cooperation can be seen as high level control, involving task and motion planning, task sharing, and formations, which can be seen as the simplest form of cooperation between autonomous robots.

## The Robots

The system is comprised of four homogeneous units that form a chain/tree society that moves in the horizontal plane, this is, the units will drive around in the floor and reconfigure themselves by driving around each other and linking to create different shapes in two dimensions.

## The Mechanics

The units in the system are mostly made out of LEGO Mindstorms NXT technology, including the main controller unit, sensors, actuators and a set of LEGO pieces that are used to create the mechanical structures. An expansion board was also developed to add some external electronics and expand the capabilities on the LEGO NXT system by adding a servo motor and 7 LED's to be detected by the LEGO Mindstorms light sensor. For more details about the units please refer to (Leal Martínez, 2009). A birds-eye-view of one unit can be seen in Figure 1.

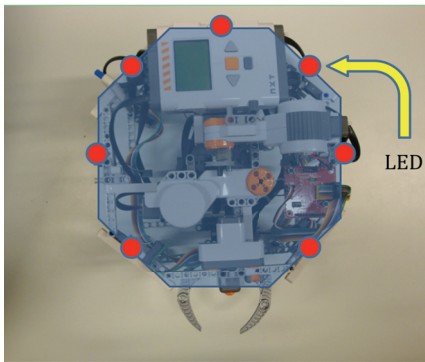


Figure 1: Robot Rotation

## The Motion System

In this prototype a novel motion technique was developed using a single actuator for traction and two actuators to both steer the driving direction and change the orientation of the outer structure independently. This enables two or more units to be attached to each other and to steer their individual motion systems without putting any kind of stress in the link, this way avoiding any undesired change in the orientation of the joint structure that could occur while steering the individual motion systems. Units are also able to totally disable their motion system to let other units do the driving and save batteries without interfering or resisting the overall movement of the joint structure.

## The Expansion Board

As the NXT brick can only connect to up to 3 actuators through its output ports and up to 4 sensors in the input ports, to expand these constraints, an Expansion board was developed using the ATmega164P micro controller, which has an I2C bus to be able to connect to the NXT, as well as 32 programmable I/O lines, real time clock, six PWM lines, two serial UARTS, analogue comparators, and many other features that make it suitable to expand the current capabilities of the NXT.

## The Software

Regarding reconfiguration a leader assignment and a reconfiguration algorithms were created the effort to start the foundations of a robot society. For the cooperation the simple case of a formation control was chosen and two algorithms were tested a organizational-learning oriented classifier system (OCS) and a simple vector-based approach.

## Reconfiguration Control

In this robotic context a society will be defined as a group of robots working in a designated area that will discover the existence of more units of the same kind and try to communicate with them in order to work together.

**Leader Assignment.** As this society needs to have a temporal leader to coordinate all the units while working towards the goal. An algorithm to designate a leader among the units present in the working area was developed and it's explained below.

As a free or master unit discovers other units close to it, it will pick the first one in its discovery list, connect to it and check its working status. It could be that the unit already is working for another master unit, leading some units towards a task (is a master) or operating alone. Depending on this unit status the unit asking will then either become master or slave of the newly discovered unit, or none depending on the case shown in Table 1.

**Assembly.** To get the units unto the desired configuration, the master unit will move all the needed units to attach to the main structure until it is completed. To achieve this it will follow the following steps:

1. **Check** if there are units assembled and if the structure is reusable, this meaning that some units are connected in a position that will be useful to reach the desired configuration, if so those units will be retained and the rest will be ordered to detach and move away from the main structure.
2. **Prepare** the main structure to be able to rotate around its axis. This will be unique at every iteration of the assembly run as every time a unit is attached/detached the master unit needs to take into consideration all the tires of the attached units in order to choose the best motion technique to make the structure rotate, and send the commands for the slaves to lift and rotate the tires accordingly.
3. **Point** any unit not in the main structure to be pointing in the main structure's direction. To achieve this, the master unit will make the main structure, that could be comprised of only the master unit, will turn all of its LEDs on and start rotating around its axis, while having all the available free units rotate looking for the source of light, and when any particular unit has reached the desired value in light intensity coming from the light sensor, all the rest will be ordered to stop moving and the main structure and the candidate unit will refine their position until the candidate is pointing straight to the main unit in the best possible angle.
4. **Align** the candidate unit to the desired hook up place. Now that the free unit is pointing towards the main structure in the best possible way, the main structure will now turn off all the LEDs but the one in the place where it wants the candidate unit attached, and will rotate until the candidate unit reports to have the same value of light intensity coming from the light sensor as the one achieved in the point step.
5. **Approach** the master unit will order the candidate unit to approach the main unit until it reaches hook up distance. The candidate unit will approach the main structure driving in a straight line while having the light sensor have the same or higher values than the ones acquired before until both the light and ultrasonic sensor confirm the unit is in the desired position. In case the light sensor stops detecting

**Table 1: Leader Assignment Cases and Their Outcome**

Asking Unit Status (AU)	Listening Unit Status (LU)	Result
Working alone (with known task)	Working alone	AU becomes master of LU
Working alone	Working alone (with known task)	LU becomes master of AU
Working alone (without task)	Working alone (without task)	Both stay as Free units and continue to search for more units that could have a task.
Leading units <sup>1</sup>	Working alone	AU becomes master of LU
Working alone	Leading units.	LU becomes master of AU.
Working Alone.	Following another unit.	AU gets the name and network identifier of the master of LU and tries to locate it to ask to join.
Leading units.	Following another unit.	AU gets the name and network identifier of the master of LU to try to locate it and ask for status.
Leading Units.	Leading Units.	If working for the same goal the one more advanced in the reconfiguration process <sup>2</sup> will become the master unit and will acquire the slaves <sup>3</sup> of the newly acquired unit. If they are both in the same stage the one having higher number of slave units will become the master unit and acquire the slave units of the other master.

the LED, the unit will stop and report to the main unit, and in this case the main unit will start turning the structure in an oscillatory motion until the candidate detects the LED again so it can resume its approach.

6. **Hook Up** to the main structure by closing the linking tool until it firmly grasps the connection point.

### Formation Control

The two algorithms for formation control were implemented in C++ and tested in a separately developed simulator called *SMRTCTRL*, described in (Leitner, 2009).

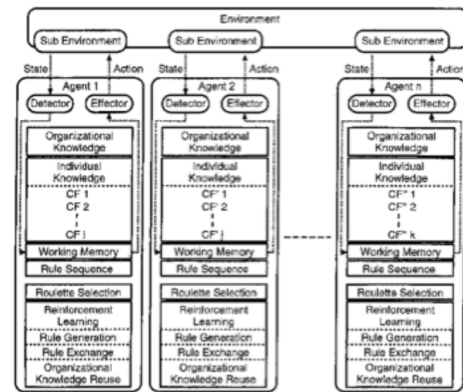
**Simple Vector-based Control.** This method uses attractive and repulsive forces, represented by vectors to control the cooperation of the robots. The vectors are calculated using the distance to the midpoint of the society, a force away from too close neighbors and depending on the coverage an attractive or repulsive force to all neighbors. The separately calculated vectors are added up and generate a movement for each robot. The coverage optimization is an emergent property of this algorithm.

**Organizational-learning oriented classifier system.** OCS takes an organizational learning approach and adapts it to machine learning in multi-agent systems. The method is described as “learning that includes four kinds of reinter-

preted loop learning” (Takadama et al., 1999) and tries to generate a hybrid solution, aiming to overcome restrictions of the Michigan and Pittsburgh approaches (Goldberg, 1989). The system, based on the learning classifier system (LCS) (Holland, 1976), uses reinforcements and evolutionary learning and is sketched in Figure 2.

The ML approach is used to calculate the actions to be taken to optimally place the robots for area coverage. The system consists of autonomous agents, with local LCS-based implementations. Each agent is able to recognize the environment and its local state and is able to change the environment due to a chosen action.

Each agent has its local memory, which is used to create, store and update rules, also called *classifiers* (CFs). These rules are used to select the most suitable action for the cur-



**Figure 2: OCS Architecture**

<sup>1</sup> Leading units by default have a task defined.

<sup>2</sup> By more advance in the process it means that it requires less units to attach to the structure to reach the desired form or shape.

<sup>3</sup> Slave units will not be asking other unit's status.

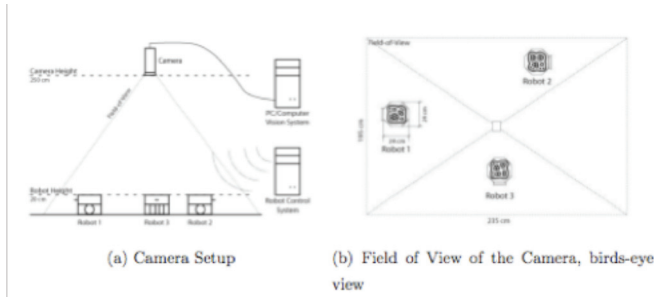


Figure 3: Gargamel Localization System

rent state. The agents apply reinforcement learning (RL) and genetic algorithm (GA) techniques for rule management. For this they detect the environment state and decide on an action based on the state. Each agent updates its rule-set and at certain crossover times exchanges some rules with another agent. A more detailed description of the two algorithms can be found in (Leitner, 2009).

### Localization

A webcam (QuickCam Pro 5000) was used in connection with a visual tracking system, nicknamed *Gargamel*, to allow localization of the robots. A simple sketch showing the experiment arrangement can be seen in Figure 3. The existing *reactIVision* system with fiducial markers was used (Kaltenbrunner and Bencina, 2007) in connection with the *SMRTCTRL* simulator.

The robots could not be controlled very precisely, even though some effort was put into the action programs, for example, the *MOVE\_FORWARD* program, it could not be ensured that the robot would move exactly one cell per action. Videos of the system controlling the robots, as well as a short presentation of the robots can be found online<sup>4</sup>.

### Results

The algorithms for formation control were tested in the simulator, evaluated and compared. Figure 4 shows a comparison of the vector-based approach and the OCS learner. In both cases terrain interactions are disabled and a standard initial configuration is used, i.e. elliptic target areas are used. The case of using circular target areas in the vector approach is added for reference.

The robots were controlled via Bluetooth and using a vector-based placement algorithm. But since the library used had no multi-threading support they were controlled sequentially. Figure 5 compares the motion of the robots as seen by the simulator (shown by the boxes) and in real-life (circles). The data presented is the average of 3 test runs performed with the final revision. The position of the (virtual) target is marked with a T. The graph shows the robots performing 12 actions (per test run), of which 3 are rotations, the rest movements. To visualize the orientation and

	Vector (circular)	Approach (elliptic)	O (average)	C (best)	S (worst)
Coverage	100 %	12 %	26.58 %	46 %	0 %
Steps	29	83	64.03	17	289
Distance [px]	230	880	1235.3	582	1843

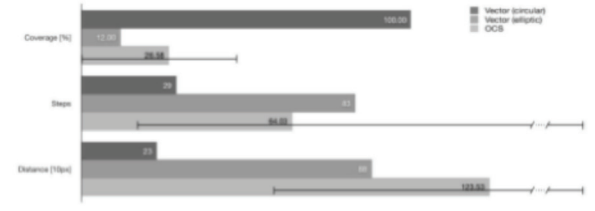


Figure 4: Comparison of the Formation Algorithms

rotation small lines are added to one side of the square or circle. Discretization errors at the center can be seen with all robots. The robots' *MOVE\_FORWARD* motion were tested and yielded 20.1 – 20.5cm, with the first movement usually a bit shorter (19.2 – 19.6cm), probably due to the initial orientation of the castor wheels.

In the first test runs a discrepancy between sent action and the outcome of the motion was seen. For example, a *MOVE\_FORWARD* action would lead, due to drift, also to a movement to a side (seen in the motion of robot 1 in Figure 5); this was then reduced as much as possible.

### Conclusions

Creating a robot society is a very complex and time-consuming task, especially when building it from scratch. This work describes the creation of a prototype society made of LEGO Mindstorms, and although some problems arose and the system could not reconfigure completely autonomously, some good results were achieved, such as:

- Prototyping with LEGO Mindstorms is feasible, basic algorithms tested the viability and stability of the system and point out areas where design improvements are needed

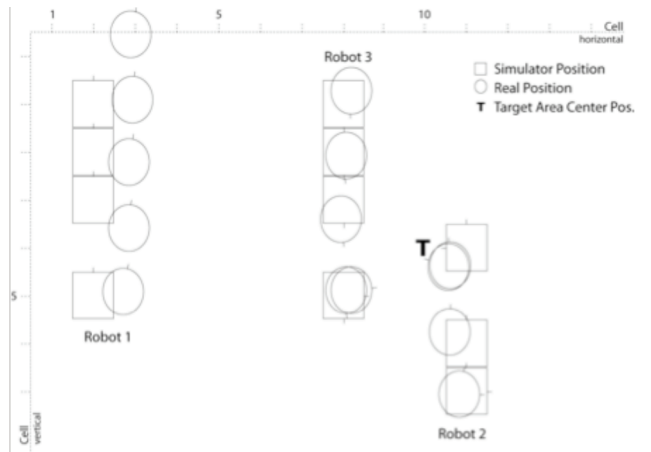


Figure 5: Robots Movement: Simulator vs. Real World

<sup>4</sup> <http://smrt.name/ijcai/>

- Lego Mindstorms NXTs can be upgraded with the expansion board and sensors and motors can easily be added
- A single traction motion technique described above was tested and good results, especially for systems that need to have the structure independent from the traction, were achieved
- The *leader assignment algorithm* was successfully implemented in *LabVIEW*
- Initial testing was done on the approach for mobile reconfiguration.
- The robots worked together with the algorithms after just a few minor changes
- The OCS approach works for area coverage, for simple cases though its complexity and computational overhead allows for better performance with a simple vector-based approach. In environments with elevations and obstacles, as well as changing conditions, the OCS approach performs better.

### Acknowledgements

The authors would like to thank the organizers of the ‘18<sup>th</sup> Annual Robot Exhibition’ for their help and the AAI and LEGO for their generous financial support.

### References

- Arai, T., Pagello, E., and Parker, L. 2002. *Guest editorial: Advances in multirobot systems*. IEEE Transactions on Robotics and Automation, 18:655–661.
- Goldberg, D.E. 1989. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman.
- Holland, J.H. 1976. *Adaptation*. In: Progress in Theoretical Biology.
- Kaltenbrunner, M., Bencina, R. 2007. *reactivision - a computer-vision framework for table-based tangible interaction*.
- Leal Martínez, D. 2009. Reconfigurable multi-robot society based on LEGO Mindstorms. Master’s thesis, Helsinki University of Technology. Espoo, Finland.
- Leitner, J. 2009 *Multi-robot formations for area coverage in space applications*. Master’s thesis, Helsinki University of Technology, Espoo, Finland.
- Takadama, K., Terano, T., Shimohara, K., Hori, K., Nakasuka, S. 1999. *Making organizational learning operational: Implications from learning classifier systems*. Computational & Mathematical Organization Theory 5(3) p229–252.