A NEW MINIROBOTICS SYSTEM FOR TEACHING AND RESEARCHING AGENT-BASED PROGRAMMING

Rodrigo da Silva Guerra[1], Joschka Boedecker[1], Norbert Michael Mayer[1,2], Hiroshi Ishiguro[1,2], Minoru Asada[1,2]
[1]Dept. of Adaptive Machine Systems,
[2]HANDAI Frontier Research Center,
Graduate School of Engineering, Osaka University, Osaka, Japan
{rodrigo.guerra,joschka.boedecker,norbert,asada}@ams.eng.osaka-u.ac.jp

ABSTRACT

This work describes a novel minirobotics system which incorporates mature technology proven efficient over the years in multi-agent robotic soccer competitions into a new versatile research and educational platform. We describe the technical aspects supporting this multi-agent robotic framework, and as an example application, we describe a method for introducing agent based programming concepts as a hands-on experience targeted at inexperienced students learning C as their first programming language.

KEY WORDS

agent-based technology,educational hardware,research on advanced technology in education,virtual reality,multimedia and hypermedia applications

1 Introduction

Since March of 2006 both CITIZEN Co. and Osaka University committed themselves to the endeavor of developing together a new miniature, and yet affordable, robotics platform. The project is intended for fostering education, research and development using robotics competitions as a driving force. The system is based on a miniature multi-robot system which mixes reality and simulation through an original Augmented Reality (AR) environment. The project has a two-folded focus: research and education. The outcome is a candidate to be a new RoboCup Soccer Simulation sub-league under the name of Physical Visualization (PV for short). The focus was on versatility and affordability, taking advantage of well established industry technologies to allow the development of an inexpensive platform. In order to do that we used the know-how of the cutting-edge and low cost watch technology as a basis for building an affordable miniature multi-robot system mixing reality and simulation. This allows the employment of a large number of robots in a rather reduced space with a very low budget and amazing portability. Both the robots and the system are to be constantly upgraded and improved, being developed together with CITIZEN. Three dominant characteristics of the project are: (a) affordability, (b) standardization and (c) open architecture.

As an experimental application for the above system we performed a short introductory level programming course for undergraduate engineering majors. In this course we used simulated robotic soccer as a framework to teach agent-based programming. The approach we chose corresponds to what Lund [1] terms *guided constructionism*, i.e., a combination of traditional constructionist approaches [2, 3] and explicit guidance in forms of lectures and coaching by more experienced students. Because the system is still under development the course was built around the 2D soccer simulator [4] which has been widely accepted as a standard research and educational platform for multi-agent applications. Nevertheless the same principles apply and we prepared a simplified C interface which should hold for both simulation and real robots. Numerous works have already been published using the 2D simulator, including both scientific level research papers (e.g. [7]) and agent programming courses (see e.g. [8, 9]). This helped making this system a robust platform for testing ideas in multi-agent disciplines. This also proves the power of having a standard for practicing education and research works.

We believe the limited programming skills of students should not prevent them from experimenting with the basic concepts of agent-based programming. On the contrary, we think allowing these students to experiment with their first codes already in such a dynamic and motivating environment helps boosting their learning of programming concepts as they become necessary for producing more elaborate autonomous soccer teams. Amplified interface to program the agents was created making it possible to cope with a very narrow time frame of only 5 sessions (each of 3 hours duration) with lectures and programming work plus an additional separate session for a class tournament.

2 Technical Aspects

The main system are illustrated in figure 2-a. Robots obey commands sent by a central server through an IR beam, while their actual position and orientation is feedback to the server by a camera located on the top. Meanwhile a number of visual features are projected onto the field by using a flat display. This system merges technical characteristics and concepts from two of the most mature RoboCup leagues, Simulation and Small-Size [10], and adds a new key-feature: augmented reality.

All the robots are centrally controlled from one CPU but their decision making algorithms run on separated networked clients, making the robots behave autonomously virtually isolated from each other. Position feedback is based on colored markers placed on top of the robots which are detected through a vision system. Robot control is based on strings of commands sent by frequency modulated infrared signals.

One characterizing feature of the system is the unmodelled embodiment dependent representation of the robots. Contrary to the misleading impression the term ”physical visualization” might imply, the robots *are not* mere physical visualizations of some sort of internally simulated mechanism of any kind. On the contrary, the system blindly sends client commands to the robots which may (or may not) respond by performing arbitrary movements. In other words, changes on the physical body of the robot would not require changes on the server internal representation of the robot's mechanisms for there is no such a thing.

2.1 The position feedback

The position of the robots (and eventually other objects, such as ball) is detected from the processing of high-resolution camera images. The computer vision system currently implemented can be divided into three main subsystems: (a) undistortion, (b) blob detection, and (c) identification & orientation. Each one is described in the following paragraphs.

Undistortion: The vast majority of consumer cameras are known to have no significant lens distortion, therefore it is common practice to assume a linear pinhole model. Despite the fact of the PV robots being real three-dimensional objects occupying volume in space, the domain of possible locations for their bodies over the plane of the flat screen is known to be confined into a two-dimensional space. Because of that the calibration problem can be reduced, without loss of generality, to a plane-to-plane linear transformation from the plane of the captured image to the plane of field itself. This transformation is a single linear 3×3 matrix operator which defines a homography in the two-dimensional projective space (see figure 1). In the presence of significant lens distortion the simple addition of a prior step for radial lens undistortion, such as in Tsai's method [11], is likely to be sufficient. Refer [12] for a more extensive review on the projective geometry approach to computer vision.

Blob detection: After undistorted, the image is segmented into blobs of certain colors of interest. These colors are defined by a mask in the three-dimensional $Y \times U \times V$ space. Adjacent pixels, in a 8-neighborhood, belonging to the same color mask configure a single blob. The area (total amount of pixels) and center of mass (average (x, y) coordinates) of the blobs are extracted. Blobs whose mass values are not within a tolerance range from the expected are discarded. This procedure is used for finding the center of the colored marking patterns on the top of each robot – the red shape seen on figure 2-b.

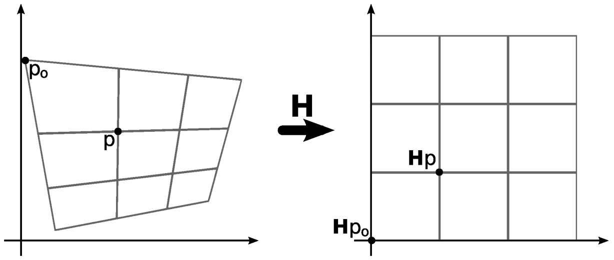


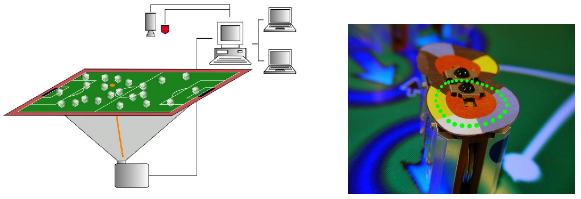
Figure 1: Plane-to-plane projective undistortion based on homography transformation, where $\mathbf{H}$ is a 3×3 matrix operator and p and p_o are 3-dimensional vectors representing points in the two-dimensional projective space

Identification and orientation: The process here described is inspired on [13]. Once a potential blob is found, a radial pattern of colors is sampled within a pre-defined radius of its center. In figure 2-b these sampling locations are artificially illustrated by a closed path of little green dots. This pattern is cross correlated with a database of stored patterns, each of which uniquely defining a robot's identity. Let's denote $x(i)$ to be the color in the pattern x at the angle i. The cross-correlation r_{xy} is calculated accordingly to the equation 1 for each pattern y the database, and for each $\Delta\alpha$ in the interval $[0^o, 360^o)$. If, for a pattern x, the minimum value of $r_{xy}(\Delta\alpha)$, for any y and $\Delta\alpha \in [0^o, 360^o)$, exceeds a minimum threshold, then the corresponding y gives the identity of a robot, and $\Delta\alpha$ gives its orientation.

$$r_{xy}(\Delta\alpha) = \frac{\sum_{i=0^o}^{360^o} \left[(x(i) - \bar{x}) \cdot (y(i - \Delta\alpha) - \bar{y})\right]}{\sqrt{\sum_{i=0^o}^{360^o} (x(i) - \bar{x})^2} \cdot \sqrt{\sum_{i=0^o}^{360^o} (y(i - \Delta\alpha) - \bar{y})^2}} \tag{1}$$

2.2 Augmented reality

The idea about the augmented reality setup is an extension of a previously published similar concept where robot ants would leave visually coloured trails of ”pheromones” by the use of a multimedia projector on the ceiling of a dark room in a swarm intelligence study [14]. Huge improvements in versatility, flexibility, and standardization can be introduced by applying that concept into a more customizable system. The figure 2-a shows an illustrative drawing and figure 2-b shows an actual picture of our system in action. Given the reduced size and weight of the PV subleague robots the application of a conventional flat display as the field becomes feasible – depending on the application, displays as small as 20-inches are more than enough. This adds much versatility to the system without adding much costs and without complicating the required setup.



(a) (b)

Figure 2: On the left an illustration of the overall system including the feedback control loop (infrared transmitter, camera, server) and the augmented reality screen. On the right an actual close-up picture of two robots playing using such setup. (See last paragraph of sub-section 2.1 for explanation about the green dots)

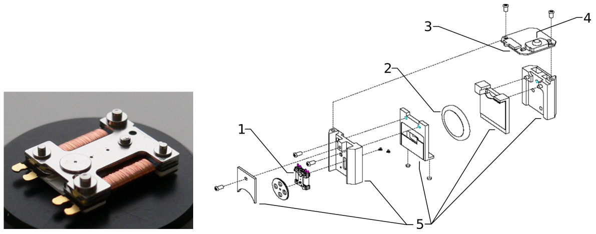
The mixture of reality and simulation enables projections of environmental features surrounding the real robots. By doing so, not only the environment becomes more visually appealing, but also allows for an enormous variety of new applications which would be otherwise impractical extending the possibilities to the limit of one's imagination.

2.3 The miniature robot

Until now, a few developments have been made on very small sized robots, being ALICE one of the most prominent names (see [15] for a survey). In terms of hardware the (current) robot here described is not much different from those many other mini-robots that have been developed so far. We emphasize that it is the unique features brought together by our framework allied to the low cost, robustness and simplicity of the architecture that make this system so attractive.

The first versions of the miniature robot here used were originally developed by CITIZEN as merchandise devices for demonstrating their new solar powered watch technologies [16]. Since March of 2006 three new prototype versions were already developed for matching the requirements of this project. The most current version of the robot has dimensions of $18 \times 18 \times 22mm$, no sensors, an infrared receiver and is driven by two differential wheels. This first robot was purposely designed to have rather simplistic hardware configuration as a starting point, a seed, to be followed by numerous upgrades in the long term. The main robot components are (numbers in accordance to figure 3-b):

1. Motor – Customized from wristwatch motor unit. See figure 3-a.
2. Battery – Miniature one-cell rechargeable $3.7V$ lithium ion polymer battery with capacity of $65mAh$.
3. Control board – Currently based on the Microchip $8bit$ PIC18 family of microcontrollers, each robot comes equipped with a PIC18LF1220 which features 4kb of re-programmable flash memory.



(a) (b)

Figure 3: On the left a close-up picture of the step motor, on the right an exploded view of revealing the robot parts.

4. IR sensor – An IR sensor is used in order to listen for commands from the PC. The sensor operates at the $40kHz$ bandwidth modulation (same of most home-appliance remote controls).
5. Body – The resistant durable body of the robot is micro-machined in aluminum using CITIZEN's high precision CNC machines.

2.4 Robot's firmware and control protocol

The current control protocol was programmed in C and compiled using the proprietary MPLAB C18 compiler. Because of the physical nature of the infrared light beam, commands have to be sent by the server to one robot at a time, in an ordered fashion. This implies that bigger number of robots result in longer control lags. Therefore the protocol format was designed so that the command could be sent in a very short time. The current command protocol has a length of $12bits$: ID ($5bits$), left command ($3bits$), right command ($3bits$), and bit parity check ($1bit$). Less frequently used instructions are multiplexed from a sequence of two or more commands.

3 The Simplified C-Interface

We prepared a set of wrapping functions allowing beginner students to implement their code in a very compact and simplified way requiring only the use of standard C code. The idea is similar to that of RoboSoc [17], but with a much stronger focus on simplicity – at the price of loosing generality. These wrapping functions were constructed around the Trilearn base code published in 2002 [18]. The whole environment was wrapped into four control functions and a few sense/act functions. The listing 1 shows an example of a complete agent which is capable of passing the ball to a teammate (variable names are consistent with figure 4). Together with our strict vector approach (which is detailed in section 3.1) this simplified interface enabled the students with very little programming experience to program a variety of soccer playing behaviors in a very clear and intelligible way.

Listing 1: Example of a simple ”passing” agent

```
#include <cinterface.h>
int main(int argc, char *argv[])
{
        struct strct_vector c;
        struct strct_vector e;
        pv_init("MyTeam");
        while (pv_update())
        {
                c = pv_getball();
                e = pv_getteammate(1);
                if (pv_cankick())
                {
                        pv_kick(e);
                } else {
                        pv_steerto(c);
                }
                pv_flush();
        }
        pv_close();
}
```

3.1 The Vector Arithmetics Approach

Most people learn basic operations with vectors already at school. Vector arithmetics are visually intuitive, especially in the two dimensional space, where calculations can be approximated by sketching simple strokes on a piece of paper. More than that, the analogy of the soccer field as a two-dimensional space gives a very straight forward interpretation for directions and magnitudes of two-dimensional vectors which can represent forces, accelerations, velocities of players and ball. In fact, this approach is so straight forward that it is very common to see implementations involving two-dimensional vector arithmetics of some kind across the various RoboCup Soccer leagues.

In our approach we developed a complete framework where all essential elements necessary for making a team of soccer playing agents could be implemented by the exclusive use of two-dimensional vector arithmetics and nothing else. Moreover, we took care of describing all necessary components relative to the self in an agent-centered approach excluding completely explicit global coordinates of any sort. This agent-centered perspective allows a deeper understanding on the agent perspective embodied with sensors and immerse in the environment.

In the figure 4 we illustrate how one can take advantage of the two-dimensional vector representation for a very visual strategy planning. Suppose, for instance, your agent were to mark an opponent by placing itself between the opponent and the ball. The vector from the ball to the opponent, according to the figure 4, is given by the expression $\mathbf{c} - \mathbf{d}$. One could simply scale down this vector, let's say, half-way ($\frac{\mathbf{c}-\mathbf{d}}{2}$), and sum to the vector representing the direction to the ball, yielding the expression $\frac{\mathbf{c}+\mathbf{c}-\mathbf{d}}{2}$. This last expression would be the direction the agent should go. Similarly, if an agent were to kick the ball ($\mathbf{c}$) into the center of the goal ($\frac{\mathbf{a}+\mathbf{b}}{2}$), it would need to go towards $\mathbf{c}+(\mathbf{c}-\frac{\mathbf{a}+\mathbf{b}}{2})(r_r+r_b)$, where r_r and r_b are the radius of the robot and the ball respectively (compare with the previous expression).

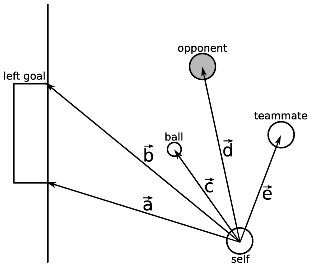


Figure 4: An example of game situation where all necessary elements can be visualized in terms of two-dimensional vectors

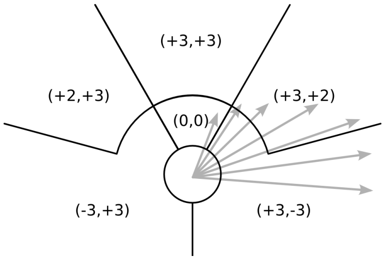


Figure 5: State machine implementing a simple steering algorithm for a two-wheeled differential driven robot

A web-based applet was developed in Java as a tool for helping the students draw their vector arithmetics on the screen of the computer and test their ideas on different game situations.

The velocity of each wheel could be set into three different constant values, both backward and forward, or zero (stopped). In such case, for example, $(+3, +3)$ would make the robot go forward, $(+2, -2)$ would make the robot spin clockwise and $(+3, +1)$ would make the robot go in an arc-trajectory to the direction forward/right. In order to face such problem we implemented a simple state-machine. See figure 5. This approach was chosen for being simple and effective enough so that the student with their limited experience in programming (and robotics) could easily understand and eventually improve their agent's navigability by customizing their own steering methods. This way they not only could understand and use very realistic robots in their experiments but also had contact with a state-machine for decision making in their code.

4 Example Application: C Crash Course

The course was performed at the Osaka University University during the month of March of 2007 to an audience of 62 second-year engineering students (amoung which only 4 were females). It was composed by five separate lec-

Day	Content summary
1st	Introduction, review of concepts of vector arithmetics, basic agent programming concepts
2nd	Introduce the most elementary code, explain the use of a simple Makefile, start working on very atomic behaviors (e.g. kick to goal and run to the ball)
3rd	Start generalizing these atomic behaviors with the introduction of simple functions (e.g. find closest teammate)
4th	Make the agent even more versatile introducing dynamic role assignment and general changes of behavior according to things such as own-id, side of play, etc.
5th	Divide the class in groups of four students and start developing their own teams for the final tournament.
6th	Tournament

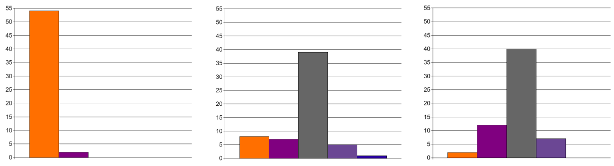
Table 1: Summary of the course program

tures of three hours each, given twice a week plus an extra sixth lecture where the tournament was realized. The overall program of the class is summarized in the table 1.

Except for the first class, which was almost completely theoretical, all the other classes were build around the student exercise and practice. In the end of each class a homework was a assigned, which would involve and stretch concepts learned in class, but also bring up issues which would only be formally introduced in the next class. This was done in order for them to experience the needs before trying the solution, so that when the solution was presented its value could be more promptly understood. For instance, students would compile using the command prompt in the first class, but be introduced to Makefiles in the second class, and they would implement different skills (such as passing) inside their main loop in the second class but learn how to create more generic parameterized functions in the third class, and so forth. Despite all the theory being not completely new to them (they came fresh from a theoretical introduction to programming), it takes quite a lot of practice and experimentation until they can really put their knowledge into work in such a dynamical situation.

4.1 Results

At the end of the course, despite the fact that we didn't talk about real robots during any of the practical classes, students showed that they have increased significantly their confidence about how much they believed they could make a real robot play soccer. See figure 6. Furthermore a much bigger number of students declared to have become more interested in both robot soccer and agent programming. Moreover, in the end of the course we found a very strong correlation between their confidence on their own agent programming skills with their confidence on how much they believed they could make a real robot play soccer (compare charts in figure 6-b and 6-c). On our interpretation the above indicates that, although we worked only with a very simplistic two-dimensional simulation environment, the students could still relate this to a broader concept applicable to real robots.



(a) (b) (c)

Figure 6: Results of the pool where students were invited to self-evaluate to what extent they believed they could make a real robot play soccer (a) when asked before and (b) when asked after the course, and similarly for making (c) an artificial agent play soccer after the course. The five columns in each chart represent the total number of students that chose each of the corresponding five options, which were, from left to right: (1) can do nothing at all, (2) can do almost nothing, (3) can do a little, (4) can do well, (5) can do very well. (the neutral option was explicitly omitted)

Figure 7: People get very attracted to the small robots

Furthermore, the authors noticed the positive effects of the tournament on the motivation of the students, which were often verbally expressed on their final reports, and also rather evident when reviewing the videos recorded during the tournament.

5 Discussion

This paper introduced the main technical and conceptual characteristics of a new miniature robotic platform. The system is still in a prototypical phase, but once it matures and more and more applications become available, it is expected to be very useful in research as well as education scenarios due to its flexibility provided by the AR environment, and the low production costs of the small robots.

During the class we collected some strong evidence of the popularity of robotics among the students. Figure 7 gives an idea of the kind of entertaining atmosphere the mini robots produce on people, attracting crowds in the public demonstrations of the system.For future work, besides developing the technical capabilities of the system further, we plan to investigate how the (eventual) use of real robots in class would influence their performance if compared to the data collected during this course in which we used simulation only. Furthermore, we want to make our evaluation methods conform to suggested [19] standards thus enabling easy comparison of results.

6 Acknowledgements

The authors would like to thank to Chisato Yoshida for her help on the empirical portions of the work, to Matthias Bohnen for developing the nice Java applet for helping on vector arithmetics, to the teaching assistants for their great help both during the classes and later on analysing the collected data, and to the students themselves for their collaboration and understanding. The authors also want to thank JSPS and JST and the Japanese Ministry for Sports and Education for their financial support.

References

[1] Lund, H.H.: Robot soccer in education. Advanced Robotics **13**(8) (1999) 737–752

[2] Papert, S.: Mindstorms: Children, Computers, and Powerful Ideas. Basic Books, New York (1980)

[3] Papert, S.: Constructionism: A new opportunity for elementary science education. A proposal to the National Science Foundation. Massachusetts Institute of Technology, Media Laboratory, Epistemology and Learning Group. (1986)

[4] Noda, I., Matsubara, H., Hiraki, K., Frank, I.: Soccer server: A tool for research on multiagent systems. Applied Artificial Intelligence **12** (1998) 233–250

[5] Kok, J.R., Vlassis, N.A.: Using the max-plus algorithm for multiagent decision making in coordination graphs. In: RoboCup 2005: Robot Soccer World Cup iX. Lecture Notes in Artificial Intelligence, Springer (2006) 1–12

[6] Stone, P.: Layered Learning in Multiagent Systems: A Winning Approach to Robotic Soccer. MIT Press (2000)

[7] Stolzenburg, F., Obst, O., Murray, J.: Qualitative velocity and ball interception. In: Proceedings of the 25th Annual German Conference on AI: Advances in Artificial Intelligence. Lecture Notes In Computer Science (2002) 283 – 298

[8] Corradeschi, S., Malec, J.: How to make a challenging ai course enjoyable using the robocup soccer simulation system. In: RoboCup-98: Robot Soccer World Cup II. Lecture Notes In Artificial Intelligence, Springer (1998)

[9] Heintz, F., Kummeneje, J., Scerri, P.: Simulated robocup in university undergraduate education. In: RoboCup 2000: Robot Soccer World Cup IV. Lecture Notes In Artificial Intelligence, Springer (2001) 309 – 314

[10] Igarashi, H., Kosue, S., Kurose, Y., Tanaka, K.: A robot system for robocup small size league:js/sii project. In: Proceeding of RoboCup Workshop (5th Rim International Conference on Artificial Intelligence). (1998) 29–38

[11] Tsai, R.: An efficient and accurate camera calibration technique for 3d machine vision. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition. Lecture Notes In Artificial Intelligence (1986) 364–374

[12] Faugeras, O.: Three-dimensional computer vision: a geometric viewpoint. MIT Press, Cambridge, MA, USA (1993)

[13] Shimizu, S., Nagahashi, T., Fujiyoshi, H.: Robust and accurate detection of object orientation and id without color segmentation. In: Proceedings of RoboCup-2005: Robot Soccer World Cup IX. Lecture Notes In Artificial Intelligence, Springer (2005) 408–419

[14] Toshiya Kazama, K.S., Watanabe, T.: Traffic-like movement on a trail of interacting robots with virtual pheromone. In: Proceedings of the 3rd International Symposium on Autonomous Minirobots for Research and Edutainment (AMiRE 2005). (2005) 383–388

[15] Caprari, G.: Autonomous Micro-Robots: Applications and Limitations. PhD thesis, Federal Institute of Technology Lausanne (2003)

[16] Yoshikawa, K.: Eco-be!: A robot which materializes a watch's life. Nature Interface (8) (2002) 56–57

[17] Heintz, F.: Robosoc, a system for developing robocup agents for educational use. Master's thesis, Dept. of Computer and Information Science, Linkopings Univ. (2000)

[18] de Boer, R., Kok, J.R.: The incremental development of a synthetic multi-agent system: the uva trilearn 2001 robotic soccer simulation team. Master's thesis, University of Amsterdam (2002)

[19] Sklar, E., Parsons, S., Stone, P.: Using robocup in university-level computer science education. ACM Journal on Educational Resources in Computing **4**(2) (2004)