CREATIVE MATH.
13 (2004), 121 - 126

The System Architecture of the Vienna Cubes Project at the Technikum Wien

H. Wahl, P. Voith, A. Hofmann, R. Pucher, A. Mense and A. Nimmervoll

Abstract. *"How difficult is it to teach robots playing soccer autonomously?"* This question was the beginning of the "Vienna Cubes" – a team consisting of lecturers and students of the University of Applied Sciences Technikum Wien that wanted to make small robots playing soccer. The goal of the project was to develop a system of hardware components (i.e. the robots) and a software framework for handling these robots. The RoboCup organization and the RoboCup world championship perform the platform where the Vienna Cubes can match with other teams all over the world. The Cubes decided to participate in the so-called F180 Small-Size League. The paper gives an overview of the whole system architecture emphasized to the implemented software framework and the ideas behind. How can the robots observe the soccer rules? How can they follow different playing strategies? How do they communicate with each other? How can they differ between the own team and the competitors? Even how do they know their current position and the position of all other robots? How must the software be tuned to fulfill the timing constraints? These questions have to be taken into account to ensure a stable system playing soccer autonomously.

1. Introduction

Soccer – a game just for human beings? This question occurred when we heard about the RoboCup group for the first time. This organization took the initiative to arrange a competition in soccer for autonomous robots. We got into this idea and decided to participate with our own team. The Vienna Cubes Team was born.

In the year 2003 – the first year of our team – we could qualify for the world championship in Padua, Italy. One year later – in 2004 – we also qualified for the world championship in Lisbon, Portugal. Lots of improvements concerning the hardware and the software of the robots were done. The Vienna Cubes got up to the quarter final. A great result and we are motivated to go on for the future.

2. The Vienna Cubes Team

The Vienna Cubes Team mainly consists of students of the University of Applied Sciences *Technikum Wien*. A few lectures provide their knowledge and help with organizational needs.

Students from the course programs Electronics and Computer Science are involved. They learn the theoretical stuff by attending the lessons. With consolidated knowledge they are able to build the hardware and implement the software of the robots.

Received: 05.07.2004. In revised form: 12.12.2004.
2000 *Mathematics Subject Classification.* 68T40, 68T45.
Key words and phrases. *Autonomous robots, robotics, artificial intelligence, image recognition, communication, Java framework.*

3. System Overview

Figure 1 shows the overview of the whole system. The *Field* in the center of the picture shows where the action takes place. Here the robot teams play against each other. The size of the field and the maximum dimensions of the robots are defined by the rules for the F180 Small Size League. Each team consists of four field players and one goalkeeper. Each robot is marked with color dots comparable with soccer shirts in human beings' soccer games (see [2]).

Figure 1: The System Overview

Above the field two cameras monitor the ongoing game. The cameras are connected to a computer that runs an *Image Recognition (IR) software*. This software finds the positions of all players (own and opponents) by analyzing the picture stream and searching for the colored marks.
The calculated coordinates of the robots are sent to a second system performing an *Artificial Intelligence (AI) software*. Primarily it analyzes the current situation of the game. Using several algorithms the program determines the best reaction and orders the robots to execute the next playing steps by transmitting commands to the own players. For example it instructs a robot to go to a new position with a specific speed. Furthermore like in a real soccer game there are a lot of rules that have to be taken into consideration. The commands are sent via Bluetooth. Because of the fact, that actually this is a very complex procedure a separate PC is used.
The arrows in Figure 1 show the main dataflow from the cameras through Image Recognition to the Artificial Intelligence system which generates commands which are sent to the robots. The robots itself just act on instruction and do not have any own logic or intelligence.
At last the system have to mention the *Referee box* and the *Graphical User Interface (GUI)*. The serial interface for the Referee box is obligatory because there has to be a referee that keeps an eye on the game and watches out for any infringements. The referee stops the game, i.e. the Referee Box instructs the software of the teams to stop the game and take some special action like penalty, free kick, etc.
The Graphical User Interface displays a lot of system parameters and outputs and gives the possibility to monitor and analyze the systems calculations and decisions. Moves and even whole games can be recorded, reviewed and analyzed using this component. Since the robots have to play autonomously there is no way to control them manually. The main task of the GUI is to get a little bit more insight into the decision processes of the Artificial Intelligence, to find errors in the algorithms and to improve them.

4. The Software Framework

The whole software framework is written in Java. At the start of the project there had been a lot of worries about the performance of Java for implementing a real time system. But several tests had passed with positive results and excluded

all possibility of doubt.
The main parts of the framework are the Image Recognition and the Artificial Intelligence which will be explained in more details.

5. Image Recognition

For the Image Recognition process two cameras are used. Actually, the software is scalable to even use more than two cameras. Each camera provides pictures streams with a resolution of 720 x 288 dots and 25 frames per seconds. As a result the software has to deal with two parallel video streams which have to be synchronized. The single pictures are digitalized by frame grabber cards and then scanned looking for objects like the ball, the own robots or the opponent robots. Specific rules are given by the organization, e.g. the color of the ball is orange, the borders of the fields are marked white and the field itself is green.
Each robot is marked with color dots identifying the robot and enabling calculation of the robots orientation on the field. Knowing the orientation is important because the ball is kicked with a kicking device located on the front of the robot.
The developed software system is highly configurable. For example all used colors can be redefined. The frame rate can be reduced for debugging sessions and as mentioned above different numbers of cameras and even different kinds of cameras can be configured.
The main purpose of the IR is to calculate and format data for the Artificial Intelligence. The achieved colors of the incoming pictures are translated to defined colors for several objects (a blurred orange of the recognized ball is transferred to a straight orange).
Because the camera is installed about four meters above the field distortions occur. Also the pictures provided to the IR are actually only 2D mappings of the 3D real world. To calculate the coordinates of the robots and get correct data for the AI a lot of corrections and approximations have to be done. After the calculation process the positions of all paying robots, their directions and the position off the ball are transmitted to the Artificial Intelligence.

6. Artificial Intelligence

The AI is the heart of the software. It weighs up the current situation and makes decisions concerning the next steps. It provides a behavior based system to control the robots.
The software is multithreading and allows the system to be flexible. The usage or non-usage of those threads is controlled by an external property file. The software could change its behavior without being recompiled.

Software architecture in game
Figure 2 outlines the actual configuration of the system. The image recognition software and the referee send data and commands to the artificial intelligence software via UDP and Serial Interfaces. The Data Logger collects all data and saves it

into a database (DB). After the game the whole play can be replayed and enabling easily finding and fixing softwarebugs.

Figure 2: Configuration of the System

Software architecture out of game

Figure 3: Simulation on one single computer.

The system is able to simulate a whole game. In that case, the behavior of both teams is controlled by the same AI software. The graphical user interface gives the user the opportunity to set up parameters like the speed of the game or the number of Robots (see Figure 3).

7. AI Modules

The software mainly consists of 6 parts. Because of system dependent time delays the image data cannot be processed as it comes from the source. Unfortunately the vision system and the mechanics cause a delay time of about 120 milliseconds. For example if a robot drives at a speed of 2.5 m/s the image recognition will detect it 30 cm away from the real position. This fact makes prediction necessary. The Prediction module is responsible for adjusting the data.
After that a strategy handler selects the best fitting playing strategy. Each strategy uses skills that can be assigned and reassigned to robots easily. The reassigning mechanism is called position switching (see Team Description Paper [1]). A potential field search method supports the software to find open spaces. The trajectory module controls the robot physically (see Figure 4).

Figure 4: The AI Modules

The software uses a potential field to find gaps and open spaces in the opponent's defense. This field consists of three layers: field hotspots layer (opponents goal highest, our own goal lowest), opponent layer (marks the opponent robots) and pass possibility layer (line of sight). To each layer a value is assigned that indicates the quality. A cost function as a combination of these values calculates one single value. The weights of the three layers for combining are changed by application.
The trajectory generation could be changed by parameters such as acceleration, maximum speed or gain. The trajectory planning is also used for obstacle avoidance. The algorithm gives the system the possibility to calculate the time a robot needs to reach its planed position.

8. Outlook

Discussions with the other teams at the world championship in Portugal brought many new ideas how we could enhance our system. We think of communication between the robots themselves combined with feedback from the robots directly to the software. It could result in less calculation power needed for finding the current

position.

By exact observations of the opponents and learning how the opponents' algorithms work we could outwit them by their own strategies.

Anyway, we work hard on improving our system and we will see what the next year world championship will bring.

References

[1] Vienna Cubes Web Site URL: http://cubes.technikum-wien.at, June 2004
[2] RoboCup Web Site: http://www.robocup.org, June 2004

University of Applied Sciences
Technikum Wien
Höchstädtplatz 5
A-1200 Vienna Austria
E-mail address: harald.wahl@technikum-wien.at