CoolRUNners Team Description Paper for RoboCup Small-Size League 2006

Steffen Prüter[1], Hagen Burchardt[1], Frank Sill[1], Frank Golatowski[1], and Ralf Salomon[1]

[1]University of Rostock, Faculty of Computer Science and Electrical Engineering, Institute of Applied Microelectronics and Computer Engineering, 18051 Rostock, Germany
{steffen.prueter, hagen.burchardt, frank.sill, frank.golatowski, ralf.salomon}@uni-rostock.de

Abstract. This paper describes the CoolRUNner team that participates in the RoboCup small-size league. It focuses on the description of the various components of control architecture. Controlling is done on two levels: the global control, with the distributed strategy module and the broadcast communication via DECT and the local control and positioning of the robot.

1 Introduction

The CoolRUNner team was founded in 2001 to develop robots that can participate in RoboCup small-size league. The development team mainly consists of high-school students and PhD students as well as research assistants and other research staff. The goal of the CoolRunners team is to participate in competitions and to provide a platform for both theoretical studies and practical work. The first robot models were presented in 2002. However, it took further two years and two robot generations until the CoolRUNners the German Open 2004 for their first time. At the German Open 2005, the team was the first runner-up out of four German teams. In 2005 a couple of high-school students won the special price of the German Engineers Association VDI [1] for their 3D simulator for the robot's play. Currently, the team is working on improving and advancing the robots to successfully take part in the European Open and the RoboCup World Championship 2006.

2 The CoolRUNners Hardware

Fig. 1 shows the current hardware platform. All of the components have been developed and manufactured by the team itself. The robot consists of three stories. The bottom level hosts the motors, wheels, wheel-encoders, accumulators, and the kicking device. The robot uses an omnidirectional drive with three wheels, which is commonly used in the small-size league. A gear belt couples each wheel with its motor. With its Faulhaber motors with 7000 rpm and 16 mNm, the robot reaches a

maximum speed of 2 m/s. 12 Ni-Mh accumulators with a capacity of 2000 mAh at 18 volts provide enough energy for up to 1 hour of operation time. The kicker mainly consists of a coil with 200 windings and a 10 cm long core. The motor driver, the 5 volts power supply, and the 180-volts charge pump for the kicking coil are located in the middle story. The 5-volts power supply provides up to 1 A for the microcontroller and the DECT communication module. The charge pump consumes 0.2 A at 18 Volts to load the capacitor with 0.2 A. This allows a kicking rate of 1.3 kicks. The power of the kicking device is controlled by the length of the activation pulses and enables the microcontroller to fine-tune the kick for passing. The motor drivers are able to supply 2.5 A for each motor. The microcontroller and the communication module are mounted on the top level. The microcontroller is a 16bit HC12 board at a clock rate of 16 MHz, 128 KB Flash and 8 KB RAM. The communication is realized with a Digital Enhanced Cordless Telecommunications (DECT) module with an enhanced protocol for broadcast.

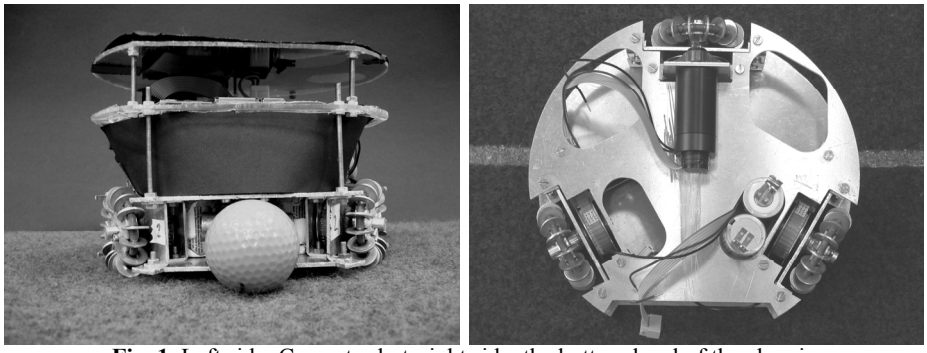

Fig. 1. Left side: Current robot, right side: the bottom level of the chassis.

The field is observed by two DFK21F04 FireWire cameras, which send their images to a standard PC. The DECT module is connected on the RS232 port of the PC.

3 Global Behavior Control

The global behavior control, as illustrated in Fig. 2, starts with the two cameras which are mounted above the field and provide a global view of the field. The FireWire cameras capture the field in 640x480 pixels in the YUV411 format at 30 frames per second. The YUV411 format provides the luminance for every pixel but the color information for only each forth.

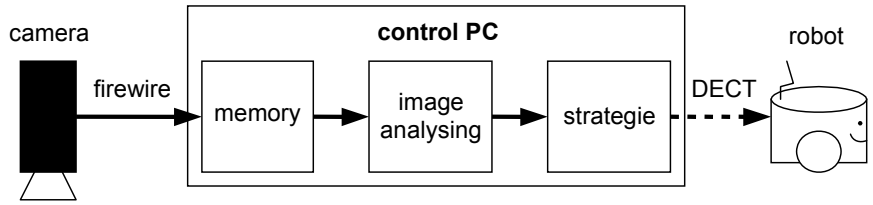

Fig. 2. Global control loop for robot's behavior.

The image data is analyzed by a capturing software which runs on the PC. It recognizes the positions off the ball and all robots as well as their alignments on the field. A subsequent global strategy module, which is also running on the PC, uses this information to calculate the roles for every robot by utilizing a potential based algorithm. The subsequent information processing can be done in two ways: The first option is that the PC calculates the command resulting from its role and sends it to the robot. The second is that the robots receive their roles and do all the calculation by themselves. To allow for the second one, it is necessary that all robots have the positions of all objects on the field, and that the field and the objects have the same global coordinates.

3.1 Communication and Coordinates

The DECT modules have a communication range of up to 100 meters. DECT uses its own frequency band between 1.88 GHz and 1.9 GHz. It does therefore not conflict with any other wireless communication technology that is used in RoboCup. The bit error rate in that a short range of up to 10 meters is low enough to omit any error detection or error correction. The disadvantage of DECT over the commonly used un-encoded radio communication is that its protocol is package based. One packet can transmit at most 37 bytes of use-content. Thus, transmitted data size has is limit to one packet because sending of a second data packet cost much time and increasing the reaction time of the robot.

The resolution of the field is 1024 and 512 pixels in x and y-axis, respectively, and 256 degrees for the orientation. Furthermore, 7 Bits reserved for the robot's commands. All this data together require 315 bits (~40 bytes). Therefore it is not possible to send all data in one packet. In order to optimize the data throughput, three different packet types are used: Two packets for the global coordinates and one packet for local adjustments.

The communication protocol has been optimized in that they control PC sends only local changes as long as the position has changed less than 70 mm. Only when the changes are larger than that, the PC sends all new global positions.

3.2 Distributed Strategy

As has been introduced above the robot's commands can be calculated on the PC or directly on the robot. For the latter, is a role based strategy an essential component.

The global strategy module can than handle global or higher cognitive behaviors as well as special situation, such as throw in or kick off; the robot cares for all this simpler but more accurate behaviors.

The global strategy module selects the best-fitting role for each robot through a combination of different overlapping potential fields. In every control cycle, the potential fields and thus the roles are recalculated for all robots. Fig. 3 illustrates the potential fields for the role `middle defender`. The team play form right to left and each robot has a number with the potential on its position. In the figure, all opponent robots are labeled with a "G". This first potential field is very large and extends to the whole field, with the highest potential being white at the standard position of the `middle defender` role. The second potential field is small, and its center is at the position of the nearest opponent robot of the standard `middle defender` position. Both fields together result in the potential that each robot has at its own position. A higher potential shows that the robot better fits to that role.

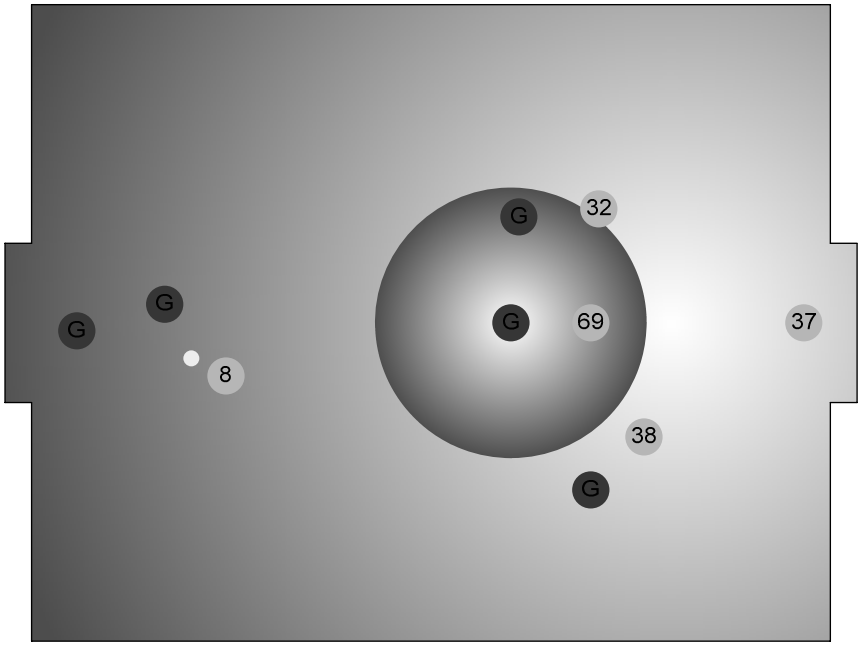

Fig. 3. Potential fields for the role `middle defender`. For more details please see text.

Subsequently, the global strategy chooses a role for each robot depending on the potentials and the current game situation such as defensive, offensive, throw in, or kick off. The roles which need no exact positioning or fast reaction with other objects calculate the resulting robot commands in the global strategy. Other roles that have direct contact with the ball or need to drive at an exact position are calculated by the

robot, because they have more precise information about their own positions and behaviors. The general rule is that rather reactive roles are processed on the robots, whereas more deliberative roles are processed on the global PC.

4 Delay compensation

The major problem when controlling robots remotely is that the delay in the processing sequence. As shown in Fig. 4, the processing sequence starts at taking the camera image and ends at the robots executing their received (action) commands.

As a consequence, the robot's position in the camera image does not correspond to its current position. The resulting actions are either inaccurate or may even lead to improper behavior in the extreme case [2],[3]. For example, the robot may try to kick the ball even though it is not in range anymore.

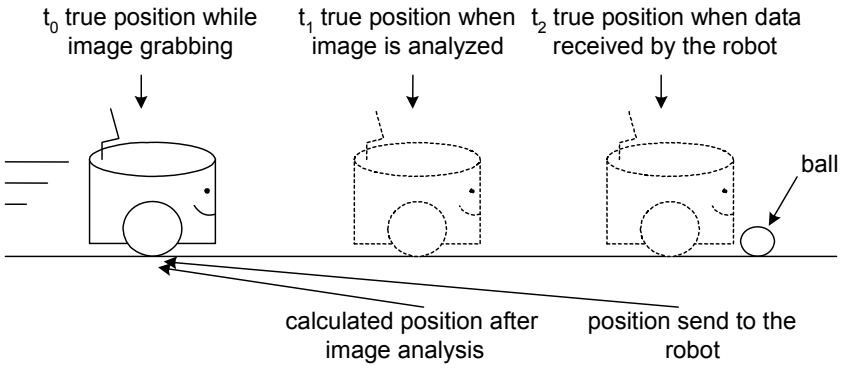

Fig. 4. Time delay in relation to processing states

Fig. 4 illustrates the various processing stages and corresponding robot positions. At time t_0, the camera takes an image of the robot at its current position. During the time, needed for the image analysis, the robot is moving, and at the end of the image analysis (t_1), the robot has already advanced to the middle position. At time t_2, the derived action commands arrive at the robot, which has further advanced to a new position. As Fig. 4 demonstrates, the robot has received a position that is farther away from the ball than its actual position.

The common way of eliminating the delay is to predict the robot's future position in the global control loop [4]. The global controlling has access to the coordinates that are provided by the camera. Therefore, a well-defined prediction equation or a neural network may be used. An option, being more in line with the autonomous-mobile-agent approach, is to compensate the delay on the robot. Here, the distance the robot has driven during the processing time of the camera image is added to the received coordinates. As the processing time delay is longer than the interval between two received data frames, the driven distances between the frames are stored in a history. When a position frame is received, a fixed number of position offsets from the history

are added to the received position. The distances are derived from the motor tick sensors. When the image recognition software is not able to find a robot, the robot's position in the data frame is set to zero. In this case, the robot then uses its internally tracked coordinates rather than updating its position with the received ones. Our experiments have shown that this way of eliminating the delay leads to a sufficient accuracy. For further improvements, a neuronal network [5],[6] that compensates the wheel slip can be implemented on the robot.

5 Roles on the Robot

Sending coordinates to the robot has a second advantage. The robots are aware of all of them: their positions, the ball's position, and the positions of the opponents. The position-awareness enables the robots to perform autonomous behaviors. The server sends only the role and some parameters. Every robot decides how it fulfills the role's task within the given parameters. In the current state of implementation, the robot only has a basic set of role implementations: a kicking role and a ball passing role.

The most basic action is to drive to a position with a certain heading. This command serves as the basis of all roles. Here, the server only sends the destination coordinates and the target heading of the robot. The robot calculates the distance to the target position, the direction it has to drive and the heading angle difference between its start and its destination position. Based on these parameters the robot calculates the speeds for the three wheels.

Whenever a ball needs to be shot, the ball-kicking role is used. Fig. 5a shows the actions a robot is performing defined by the ball-kicking role. Here, the server sends the center of the area that the ball should hit and the area's radius to the robot. The robot's task is to autonomously navigate into a shoot position behind the ball and to shoot the ball when the predicted firing line intersects the target area. Because the robots position is updated by tracking the wheel encoders, the robot can test for the correct shoot position at a much higher frequency than the global server could do.

The ball-passing role is a special case of the ball-kicking role. Fig. 5b shows the actions a robot is performing defined by the ball-passing role. When the ball target coordinates match the coordinates of another robot, the robot assumes that it should pass the ball to this one. The distance to the target robot is calculated and the shooting strength is lowered appropriately in order to avoid bouncing of the ball. In the future, the robot will also predict the position of the target robot, which enables to passing the ball to moving robots.

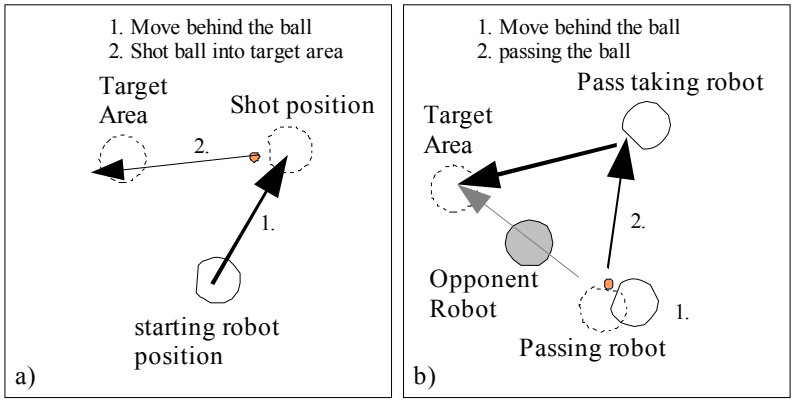

Fig. 5. Implemted roles on the robot. a) goal shot, b) passing

6 Conclusion and Outlook

The CoolRUNners team focuses on a potential-based role implementation. The team's special approach is to implement those parts directly on the robots that are traditionally calculated on the global strategy server. The communication between the global server and the robots utilizes the DECT technology, which does not interfere with other wireless communication technologies that are used in the RoboCup leagues. Its main advantages are the high communication range and the low error rate. Current developments are the enhancements of the image analyzing software, the ball detection on the robot, path planning on the robot, and the advancement of the movement control on the robot.

Acknowledgements
The authors gratefully thank Prof. Dirk Timmermann, Prof. Hartmut Pfüller, Guido Moritz, and Thorsten Schulz for their continuous support and funding.

References

[1] 'VDE setzt auf forschenden Nachwuchs' in VDE dialog, 4/2005, page 12.
[2] J.C.Alexander and J.H. Maddocks, On the kinematics of wheeled mobile robots, Autonomous Robot Vehicles, Springer Verlag, pp.5-24, 1990.
[3] Balakrishna, R., and Ghosal, A., "Two dimensional wheeled vehicle kinematics," IEEE Transaction on Robotics and Automation, vol.11, no.l, pp.126-130, 1995
[4] A. Gloye, M. Simon, A. Egorova, F. Wiesel, O. Tenchio, M. Schreiber, S. Behnke, and R. Rojas: Predicting away robot control latency, FU-Berlin, June 2003.
[5] D. Rumelhart, J. Mccelland: Parallel Distributed Processing, MIT Press, 1986
[6] D. Rumelhart: The basic ideas in neural net-works, Communications of the ACM, 1994