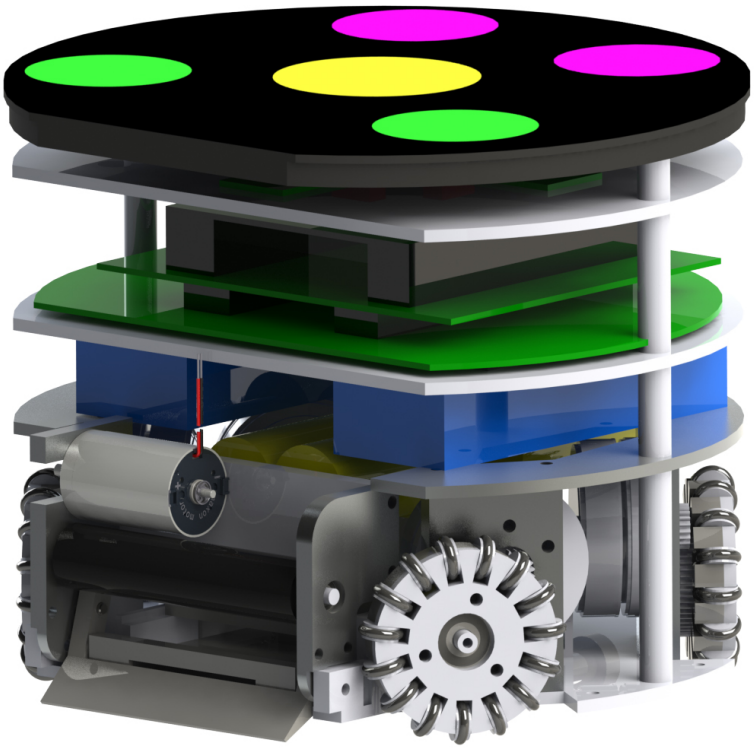
ER-Force
Team Description Paper for RoboCup 2014

Harald Bayerlein, Alexander Danzer, Michael Eischer, Adrian Hauck, Markus Hoffmann, Philipp Kallwies and Markus Lieret

Robotics Erlangen e.V.
Pattern Recognition Lab, Department of Computer Science
University of Erlangen-Nuremberg
Martensstr. 3, 91058 Erlangen, Germany
info@robotics-erlangen.de
http://www.robotics-erlangen.de/

Abstract. This paper presents an overview description of ER-Force, the RoboCup Small Size League team from Erlangen located at Friedrich-Alexander-University of Erlangen-Nuremberg, Germany. New developed parts in the hardware and electronic design are described. The last paragraph is about the new concept behind our strategy. Furthermore, upcoming changes and improvements are outlined.



1 Introduction

This year's TDP discusses diverse mechanical and electronical problems as our robots received a complete overhaul. The mechanical part is addressed to less experienced teams in order to facilitate the development of adequate hardware and to provide help entering Small-Size-League. The sections on electronics and strategy show solutions to some problems that occurred during last RoboCup and rather support established teams.

2 Mechanics

During the last two years we developed a new mechanical design for our robots that will be used for the first time in this year's RoboCup. The following paragraphs will outline the key-features and the function principle of our new hardware.

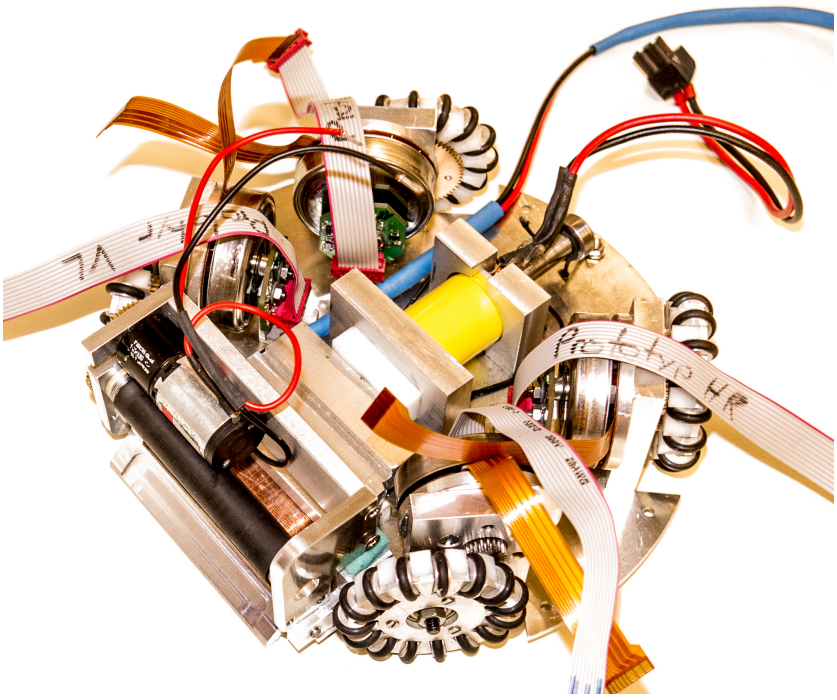


Fig. 1. Mechanical design overview

2.1 Kicking systems

Our new kicking system consists of a flat kicker powered by a cylindrical solenoid in the center of our robot and a chip kicker pushed by the plunger of a solenoid mounted in the bottom plate of the robot.

Flat kicker The flat kicker is the typical construction with a divided plunger rod. One part is made of magnetic steel and the other one is made of a non magnetic alloy. The plunger is attached to the kicking plate which is mounted in a linear slide made of polyoxymethylene (POM). This material is used as it is a very lightweight but stable material and shows very low friction coefficients in combination with aluminum alloys.

The kicking plate and the non magnetic part of the plunger rod are made of 7075 aluminum alloy to gain a lightweight but robust construction, allowing to obtain a maximum ball speed of approximately 9 m/s.

Chip kicker Due to the fact that our last chip kicker design revealed some serious problems, such as a lack of power or failure of some components, a new kicking mechanism is implemented in the new robot generation.

It consists of a flat solenoid glued to the bottom plate of the robot and a small bent chip kicking plate that is pivoted at the mount of the dribbling bar above the solenoid. It chips the ball when pushed by the plunger. Due to the complex geometry this solenoid is sintered of polyamide, later on resin-impregnated to increase stability and wound with six layers of AWG 24 enameled wire.

The chip kicker is also made of 7075 aluminum alloy to prevent deformation caused by high forces that occur during the kick. The plunger is a rectangular plate made of S235 and a thickness of 3.5 mm. When accelerated through the solenoid, it transfers the impulse to the chip kicker and then stays centered in the solenoid. Therefore no additional end stop to limit the movement is necessary.

By now, it is possible to chip the ball at a maximum distance of 5 m meanwhile reaching a maximum height of 1.5 m at the zenith of its trajectory.

2.2 Dibbling Device

The dribbling bar is an aluminum rod mounted in two friction bearings and covered with a neoprene rubber hose.

Properties of different materials, such as polyurethane or silicon, were evaluated as coating for our dribbling bar. We finally decided to use a neoprene hose with a durometer A hardness of 70 and 16 mm in diameter as it offers the best friction between dribbler and ball. When the hardness is significantly lower than Shore 70A, the dribbling bar will deform when in contact with the ball. This leads to poor ball reception and handling as it repetitively jumps away from the dribbler and then rolls back due to its rotation.

Materials with durometer A value greater or equal then 90 like certain nitrile rubbers may be not flexible enough. The ball will rebound too much when impinging up at the dribbling bar and the reception deteriorates.

As dribbling motor a customized Maxon DCX22 motor with an idle speed of 12400 rpm and nominal torque of 14.6 mNm is used. The torque is transmitted to the rubber-covered dribbling bar using gears with a transmission ratio of 35:55.

Since there is an additional transmission between dribbling bar and ball a theoretical rotating velocity of 6000 rpm and 30 mNm torque can be reached. Because of irreversibility in the frictional connection the actual ball kinematics will differ but turned out to show a sufficiently high performance to allow for a very good ball handling.

Motor and dribbling bar are attached to a pivoted mounting that rotates about seven degrees backwards upon ball impact and is damped by two pieces of polyurethane foam to dissipate the kinetic energy of the ball when a pass is received and to insure a maximum of ball handling performance.

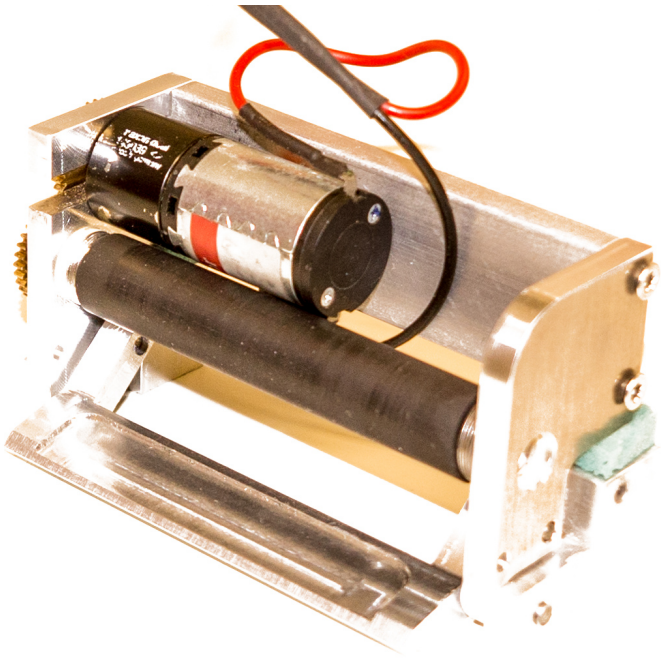


Fig. 2. Chip kicker and Dribbling device

2.3 Wheels and chassis

The wheels are typical omniwheels consisting of a main part made of POM with a diameter of 52 mm and within this component 15 subwheels are symmetrically mounted on individual axes. Each subwheel has a diameter of 9.2 mm and is covered with a Shore 70A nitrile rubber o-ring for best friction and drivability.

The two rear motor units are screwed to the bottom plate at an angle of 45 degrees, the two front units are mounted at an angle of 35 degree to enable consistent forward and sideward movement.

Maxon EC-45 flat motors are still used, transmitting their torque to the wheels by a 3:1 gear transmission.

3 Electronics

In our new 2014 electronics design, we introduced numerous major changes and new features. All printed circuit boards were redesigned as we developed a completely modular concept which we will describe in the following section. Furthermore, our new motor control featuring sine commutation and the changes to our transceiver for a more stable return channel will be discussed.

3.1 Modular Design

Although we already placed every elementary function of the robot on a separate board since 2012, we decided to advance this concept with the opportunity a complete redesign offers. We made positive experiences with the old modular design regarding maintenance and reliability. However fixing an unapparent problem still could take a long time, e.g. we placed the output stage of our old motor controllers on additional boards but left the corresponding microcontrollers on the mainboard. When there was a problem with the mentioned microcontroller, we had to search for the problem manually in a time-consuming way.

As a result of these experiences we decided to use an all-modular approach for our new generation as depicted in fig. 3. The central distribution board is horizontally connected to motor controllers and a boost controller which can be interchanged as we use the same connectors at every docking slot. These connectors provide power (+3V3, +5V and V_{BAT}), CAN and JTAG to the docked boards. The main controller is attached vertically to the distribution board and handles the communication with the RF-board on the top of the robot via SPI using LVDS transceivers. Additionally, there is the possibility to add a BeagleBoneBlack microcontroller board onto the main controller which also communicates via SPI with the main board. During design and test stage, this addon allows us to do live-debugging attaching a Wifi dongle to the BeagleBone's USB port.

The radio functionality of the RF-board was not changed, because we are very content with its performance. However we also applied our modular concept to it: The RF-Docking board is connected to the main controller and distributes its signals to the four RF-Modules, each consisting of the nRF24L01 transceiver chip and a PCB-Antenna which is described in our last year's TDP.

3.2 Sinusoidal Commutation

We decided to change motor commutation for the brushless DC drive motors from block to sinusoidal. One advantage of the sinusoidal commutation is a

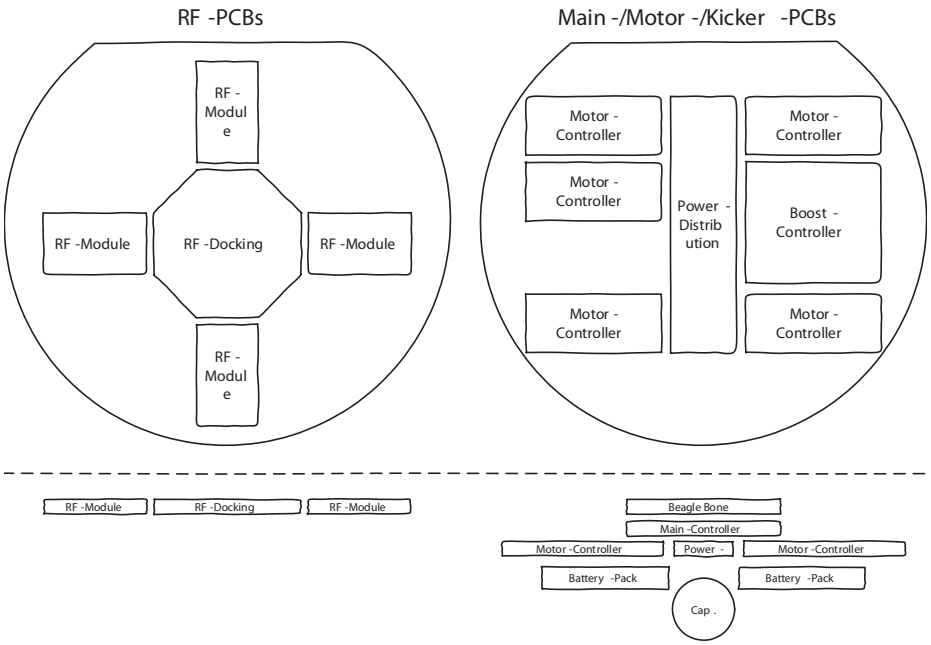


Fig. 3. Modular PCB Design

slightly higher torque (about 4%). The main advantage however is that motor speed and torque are independent from each other. The basic structure of the driving circuit is outlined in fig. 4.

For commutation three sinusoidal currents are necessary, one for each phase (A, B, C). The phase-shift for phase B is $+120°$ ($\frac{2}{3}\pi$) and for phase C it is $-120°$ compared to the current of phase A. The resulting currents are:

$$
\begin{aligned}
I_{\mathrm{A}} &= \hat{I} \cdot \sin\left(2\pi f \cdot t\right) \\
I_{\mathrm{B}} &= \hat{I} \cdot \sin\left(2\pi f \cdot t + \frac{2}{3}\pi\right) \\
I_{\mathrm{C}} &= \hat{I} \cdot \sin\left(2\pi f \cdot t - \frac{2}{3}\pi\right)
\end{aligned}
$$

This leads to a rotating magnetic field. The rotor of the motor follows this field. The motor speed n only depends on the frequency f of the current applied to the phases and the number of pol-pairs p of the motor:

$$
n = \frac{f}{p}
$$

Changing the phase-shift for phase B to $-120°$ and phase C to $+120°$ leads to a reversed motor direction.

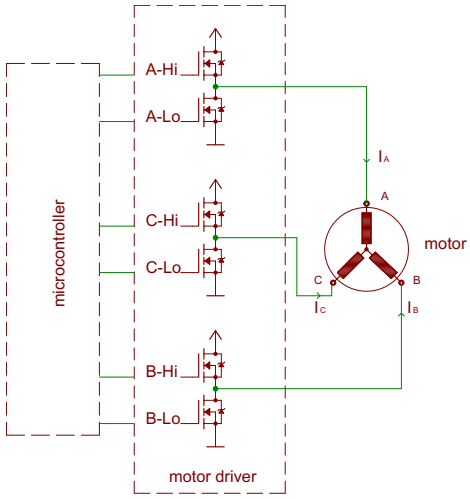


Fig. 4. Motor drive circuit

The maximum torque produced by the motor depends on the amplitude of the currents $\hat{I}$. As the frequency and the amplitude of the sinusoidal currents are independent from each other, motor speed and torque can be influenced independently.

The required sinusoidal signals for the motor driver are produced by the microcontroller via PWM. To get the positive part of the sinus, the high side transistor of the corresponding half bridge is switching according to the PWM signal, while the low side transistor is switched off. For the negative part the low side transistor is switched on and off while the high side remains off.

By measuring the angle between the rotating magnetic field and the rotor position the torque applied by the load can be determined. The rotor position is given by the hall sensor signals. Each hall sensor corresponds to the sinusoidal signal of one phase. When no load is applied to the motor this angle is approximately zero. When the load increases the angle also increases. By increasing the motor current the maximum available torque of the motor is increased and the angle is reduced, as long as the load is not changed. When the current is held on a constant level and the load decreases, the angle also decreases and the motor current can be adjusted accordingly by decreasing the amplitude of the sinusoidal signal.

3.3 Radio communication

In our last year's TDP the wireless part of our robot was described. This year we will focus on the PC-side. For our main transceiver connected to the control-PC we use special boards with NRFs and signal amplifiers. In the current design we have one transmitter and one receiver in one case. As first tests have shown the transmission-path computer-to-robot had no problems. We separately tested

the signal-path from the robot to the computer. Again no transmission troubles. Unfortunately the simultaneous operation of up- and downlink didn't work at all. Even a frequency difference between both links did not fix this problem. Responsible for this behavior is the amplifier on the receiving board. The amplifier on this module amplifies the whole range of frequencies supported by the NRF-chip. So the strong transmitted signal distorts the robots signal. The first fix for this problem was to define separate timeslots for transmitting and receiving, as shown in fig. 5.

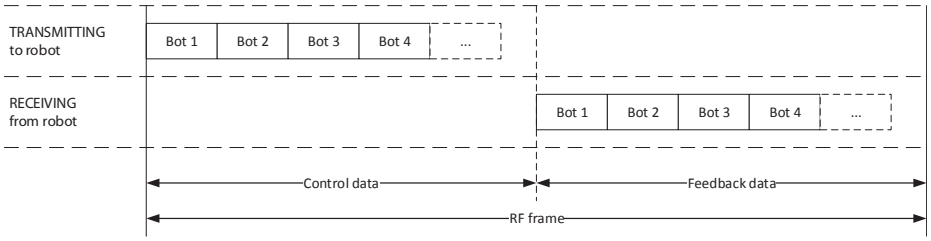


Fig. 5. RF frame

Afterwards we hit the idea to change our antennas from no name 10cm dipoles to specific ones. Now we use normal wireless LAN equipment. It's the cheapest solution und 100% compatible with the frequency range of the NRF-Module. For transmitting to the bot we use a dipole antenna with a vertical radiation angle of approx. 20°. In the horizontal plane the radiation angle is 360°, of course. This characteristic results in a signal gain of 7 dBi. For receiving data from the bot we have a directional antenna. With this devices the receiving angle in the horizontal plane is 80°, in the vertical one it is 70°. With these changes the simultaneous data flow of up- and downlink works fine. Of course the position of the transceiver to the bots is now essential for the performance. For the vertical position the transmitting antenna is the limiting factor. The ideal height is the field level. The horizontal position is limited by the receiving antenna and should be at the center of one sideline.

4 Strategy

Shortly before RoboCup 2013, we decided to switch from a Skill-Tactics-Play (STP) approach to a multi-agent system. The reconstruction is now finished and we would like to present its main ideas.

4.1 Motivation

Although a lot of teams, including last years finalists [1] [2], use STP in their strategy design, it turned out not to be suited for our needs. Some properties of STP, which can be found in the original description [3], impose problems.

A *play* in STP assigns a role to every robot of the team. With six robots on the field, the high number of possibilities leads to many plays. Unfortunately, this implies that several plays cover similar situations. Extracting shared behavior and deduplication of code is hard because plays are mainly state machines.

Another technical difficulty is that STP is not designed to keep state upon play switching as each new play assigns new tactics. Consider a defender which is not involved in an offensive move, for example when marking an opponent. Whenever a new play comes to execution, the robot will receive a new tactic, although it was not actively participating in the previous move. Compensating for this problem leads to needlessly complex plays.

The necessity to terminate plays at a specific moment in time poses a big challenge. Taking passing as an example, it is quite easy to tell if a pass was successful in retrospect. But it is hard to detect reliably if something is going wrong as there is no time to wait for certainty. Aborting too early may result in wasted opportunities. Reluctance may cause the strategy to pursue an unavailing goal. This may be inevitable for a single robot, but the problem with STP in such a situation is that there is a single objective for the whole team. In case of waiting for a failing pass, not only the pass receiving robot is following a plan to no purpose but the whole team.

As the small size league aims towards more robots on the field, it seems to be more practical to define goals for each robot individually instead of for the whole team. Furthermore, this approach enables us to handle various numbers of robots on the field more flexibly.

4.2 Overview

Our new design implements a multi-agent system. Access to the world model and all strategy logic happens on a single computer, so communication and information sharing is quite straightforward. Nevertheless, each robot acts on his own as an agent and interacts with his teammates through a well-defined messaging interface. There are three types of agents: Goalkeeper, Defender and Attacker. The type of an agent defines a number of behaviors which in turn choose a specific task to accomplish. The *trainer* is a special team member. He decides the ratio between defensive and offensive players. Another important job of him is the synchronization of collaborative tasks.

4.3 Architecture

A robot operates on three layers of abstraction called *agent*, *behavior* and *task*. In each layer there is access to the message system with each robot representing a participant.

Each *agent* type has an own set of *behaviors* to choose from. The selection depends on the current game situation. A behavior aims at a high-level target like performing a free kick or providing default behavior for a defender. The next level of abstraction is the *task* which handles the actions of a robot. That is, a task tries to accomplish a concrete goal while having access to *skills* for physical

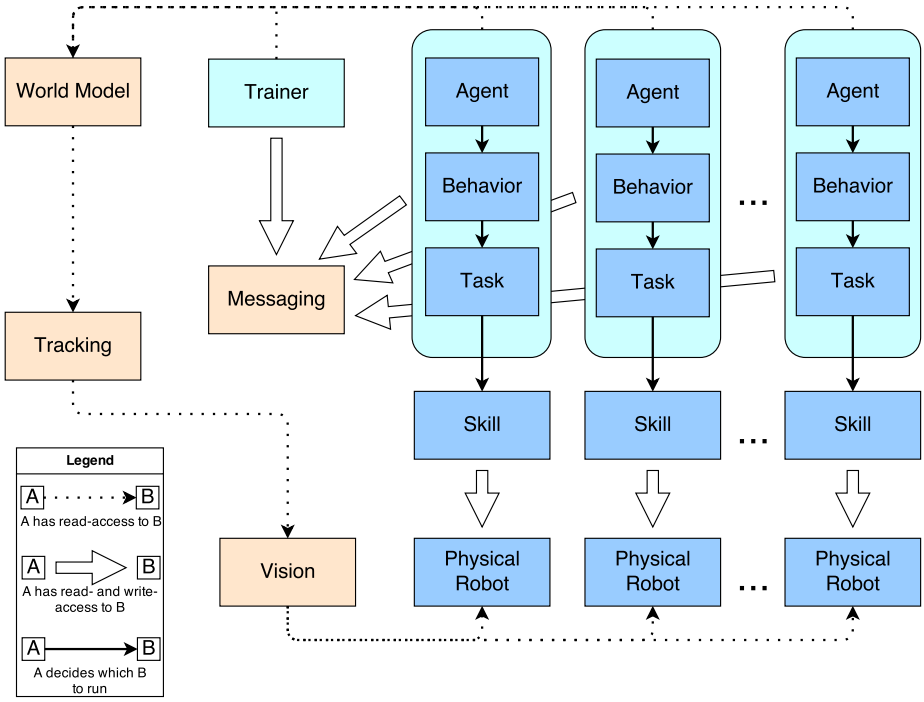


Fig. 6. Overview of the ER-Force strategy architecture

control. A skill is a high-level command to the robot like moving to a position or performing a shot with a certain speed.

The messaging module provides a common channel for agents and trainer. Unlike in STP, there are no hierarchic decisions for all robots. The trainer acts as a mediator for actions which involve several agents. Game patterns emerge from the interplay of all robots.

We instantiate an agent for every friendly robot on the field. This is a very flexible way of handling a variable number of players on the field. There is no explicit check on the number of robots in the game. An agent looks only for a suitable teammate if necessary.

Long-term planning, especially when involving more than two robots, is more challenging with this approach. But the trainer component is well-suited for this task. Sophisticated agent communication aims for making hard-coded action sequences dispensable. We concentrate on a highly dynamic style of play which is characteristic of the small size league. That is, all agents reconsider their current planning at every time step but also respect ongoing sequences of actions.

4.4 Passing

As described in the previous paragraph the agents only share information via message passing. Thus to allow for coordinated maneuvers like passing an interaction protocol is required.

The first step for passing is to determine which robot should be the pass target. That robot should be able to make use of having the ball and in most cases not be a defender or the goalie. Passing to these robots should be avoided as losing the ball could lead to a goal. In order to avoid this problem every possible pass target has to send a message to the robot currently owning the ball. The shooting robot then decides which of these robots to pass to.

The pass is signaled to the receiver by sending another message which causes that robot to prepare for receiving the ball. Due to the possibility that the ball owner changes its decision the receiving robot only switches to catching the ball once it has been shot. This ensures that only one agent considers itself the pass target.

The described messages are sent repeatedly as long as the action takes place. That is as long as a robot takes part in an action it keeps sending the related messages. This simplifies handling changes in the list of possible pass targets.

Communication is also required to have only one robot trying to catch the ball while it is rolling over the field. Each robot who wants to get the ball informs the trainer how well suited it is for that task. The trainer then decides which agent is allowed to pursue the ball. By handling this decision in a central place it is very simple to add the required hysteresis to avoid flickering of the robot selection.

4.5 Implementation

The strategy is written in the Lua programming language [4]. We use the LuaJIT just-in-time compiler, which allows for low latency even on standard computer hardware. The dynamic interpretation enables a fast work flow. Saving a modified code file triggers the framework to reload the strategy. This is especially handy for quick adaptions in tournament situations.

5 Logplayer

Since RoboCup 2012 the data flow in our software framework uses *Google's protocol buffers* for exchanging data between the different steps of the pipeline. This also includes every information passed on to the GUI. To allow for later inspection of the strategy behavior we store this information in a log file.

Using the logplayer (fig. 7) we can quickly look at every moment of the game. Along with a configurable playback speed we are able to analyze a situation in slow motion or even frame by frame. During last years RoboCup this turned out to be extremely useful for understanding the strategy behavior as it is almost impossible to see all relevant details during the very fast SSL games.

6 Conclusion

In the previous sections an overview of the 2014 hardware system and strategy was provided. It aimed not only at helping less experienced teams but also at

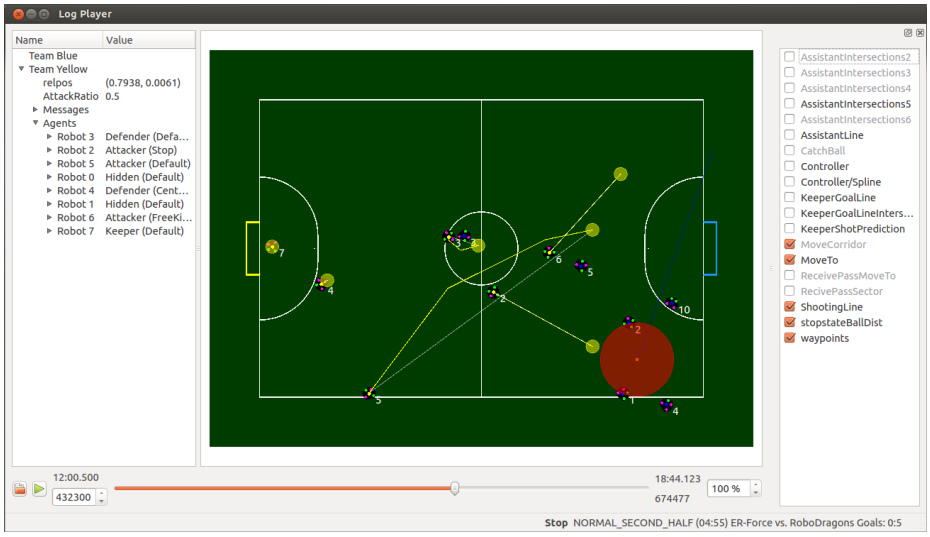


Fig. 7. Screenshot of the logplayer

introducing new design variations to long-established teams. ER-Force would appreciate a discussion with other teams about the improvements presented in order to find feasible system concepts for potential SSL entry candidates.

Implementing a fully modular design in both mechanics and electronics represents an anticipated milestone for ER-Force. Assuming these systems work as well as they promise this allows to focus on unattended improvement opportunities such as path finding and tracking. For instance, using additional information about detected robot patterns from SSL-Vision might strongly reduce the omnipresent problem of disappearing robots when covered by the goal bar or shadows. Therefore, ER-Force aims to contribute to an overall improvement of the Small-Size-League.

References

1. Wu, Y., Yin, P., Zhao, Y., Shen, Y., Tong, H., Xiong, R.: ZJUNlict TDP for RoboCup 2013 (2013)
2. Biswas, J., Mendoza, J.P., Zhu, D., Etling, P.A., Klee, S., Choi, B., Licitra, M., Veloso, M.: CMDragons TDP for RoboCup 2013 (2013)
3. Browning, B., Bruce, J., Bowling, M., Veloso, M.: STP: Skills, tactics and plays for multi-robot control in adversarial environments. Journal of Systems and Control Engineering **219**(1) (2005) 33–52
4. Ierusalimschy, R.: Programming in Lua, Second Edition. Lua.Org (2006)