

53. IWK

Internationales Wissenschaftliches Kolloquium
International Scientific Colloquium



Faculty of
Mechanical Engineering



PROSPECTS IN MECHANICAL ENGINEERING

8 - 12 September 2008

www.tu-ilmenau.de

th
TECHNISCHE UNIVERSITÄT
ILMENAU

Home / Index:

<http://www.db-thueringen.de/servlets/DocumentServlet?id=17534>

Published by Impressum

Publisher
Herausgeber Der Rektor der Technischen Universität Ilmenau
Univ.-Prof. Dr. rer. nat. habil. Dr. h. c. Prof. h. c. Peter Scharff

Editor
Redaktion Referat Marketing und Studentische Angelegenheiten
Andrea Schneider

Fakultät für Maschinenbau
Univ.-Prof. Dr.-Ing. habil. Peter Kurz,
Univ.-Prof. Dr.-Ing. habil. Rainer Grünwald,
Univ.-Prof. Dr.-Ing. habil. Prof. h. c. Dr. h. c. mult. Gerd Jäger,
Dr.-Ing Beate Schlütter,
Dipl.-Ing. Silke Stauche

Editorial Deadline
Redaktionsschluss 17. August 2008

Publishing House
Verlag Verlag ISLE, Betriebsstätte des ISLE e.V.
Werner-von-Siemens-Str. 16, 98693 Ilmenau

CD-ROM-Version:

Implementation
Realisierung Technische Universität Ilmenau
Christian Weigel, Helge Drumm

Production
Herstellung CDA Datenträger Albrechts GmbH, 98529 Suhl/Albrechts

ISBN: 978-3-938843-40-6 (CD-ROM-Version)

Online-Version:

Implementation
Realisierung Universitätsbibliothek Ilmenau
[ilmedia](#)
Postfach 10 05 65
98684 Ilmenau

© Technische Universität Ilmenau (Thür.) 2008

The content of the CD-ROM and online-documents are copyright protected by law.
Der Inhalt der CD-ROM und die Online-Dokumente sind urheberrechtlich geschützt.

Home / Index:

<http://www.db-thueringen.de/servlets/DocumentServlet?id=17534>

Jürgen Gausemeier, Rafael Radkowski
Bernd Kleinjohann, Philipp Adelt, Willi Richert, Henning Zabel

Augmented Reality to Support the Testing of Autonomous Systems by the Example of Soccer Robots – Concepts and Experimental Results

Introduction

By the term “Autonomous System” all technical systems are denoted, which have the commonness to follow up their purpose autonomously. These can be mechatronic systems, embedded systems or agent-based technical systems. One property of autonomous systems is the inherent intelligence of their control-software, which complicates the testing of those systems. At the C-LAB (Cooperative Computing & Communication Laboratory), a joint research and development laboratory operated by the Siemens AG and the University of Paderborn, the so called Paderkicker soccer team has been developed (Figure 1) [1]. The Paderkicker team consists of five robots that already participated successfully in the German Open competition in 2004, 2005 and 2007, the Dutch Open 2006 and the RoboCup 2006 World Championships. A single Paderkicker robot is an autonomous real-time system; it includes embedded real-time architectures, real-time vision, learning and adaptation from limited sensor data, skill learning and methods to propagate learned skills and behaviors in the robot team.

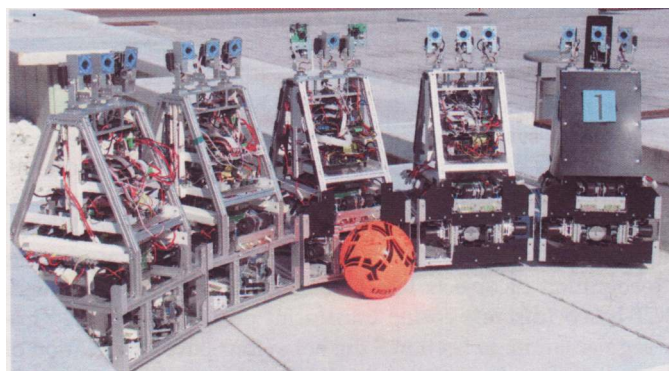


Figure 1: The Paderkicker soccer team

For orientation on the field, the robots use a vision system. It contains three independent pan-tilt cameras that are used for active viewing. Each camera may focus and track independently a different object of interest like a ball, corner post, or goal. A vision module controls the cameras. It processes the images separately, perceived by the

three cameras and tries to detect certain objects like the ball, goals or lines. The detected objects are fused by a particle filter (Bayes Filter), which also evaluates odometric data from the driving system of the robot. At least, the particle filter describes the estimated position of different objects, like a goal, lines, etc. in relation to a single robot. Thereby each particle describes an estimated position of one object. The particle filter is the basic for the model and action system of the robot to plan and prepare the next moves.

To get accurate data is a big challenge, particularly as sensors normally deliver noisy data. Due to the noisy data, the object recognition detects unimportant things as a line or a ball. The latter occurs in an inexplicable and irrational behavior. Because a single Paderkicker is an embodied embedded system, which reacts autonomously during a game, it is difficult to understand and find the origin of this unexpected behavior. The inner state of a robot and its inner decisions are hard to observe for the developers. To understand the behavior seen on the field, a visualization of the detected particles and the related objects is necessary.

The technology Augmented Reality (AR) is a human-machine interface, which superimposes the perception of the reality with computer-generated information [2]. This information can be for example a 3D-model, text or an annotation. They are visualized with relation to a real world object. In order to be able to use AR, special viewing devices are needed. A classic viewing device is the so-called head mounted display (HMD): A goggle with two displays in front of the eyes instead of the eyeglass lenses. The user sees the reality in the displays and the computer-generated information, which superimposes the reality. In our approach, AR is used to visualize the data of the particle filter during operation time of robot. AR visualizes detected particles as points on the field and in the environment of the field. The advantage is that the vision data is visualized with respect to the related real world object. This way, AR facilitates to understand the world, the robot perceives.

This paper describes the AR system and the visualization we use to show the developers, which objects a robot has recognized during a game. First we present an introduction to the Paderkicker robots. We describe the software architecture and the vision system. Next, the AR system is described. This includes the used hardware, the software architecture and the communication between the Paderkicker and the AR system. Following, the applied visualizations are described as well as some experimental results. We finish the paper with a summary and an outlook.

Related Work

AR has been used in many different projects to support the handling of technical systems during operation. Two main aims could be identified in these projects:

- Visualization of internal data of a technical system during operation.
- Support the interactive manipulation of technical systems.

In the first case, AR is used to superimpose data about the system directly at the position the data is related to. This enables a faster perception of the information, because no transfer of the data to a related object is necessary, like it is, if the data is visualized on a screen without any relation to an object. One Example can be found in [3]; the authors present an AR-application for maintenance of electric circuits. The user sees information about the position of certain electrical elements and the handling, which needs to be performed during maintenance. Another work is presented in [4]. The authors developed an AR-application, which visualizes different features of a metal working tool. The application visualizes the result of simulations, showing the behavior of the working tool, directly on the machine. Eckes developed an AR-system for the ramp-up process of production lines [5]. The AR-system visualizes different data, e.g. the state of a switch, using state diagrams. This helps the engineers to get an overview about the system behavior and helps to detect errors. Reinhäckl and Schilling developed an AR-application for an erode process [6]. One of their application scenarios describes the comparison of the position of the real erode electrode with the reference position of a simulated one. The difference is shown with an AR visualization of the electrode. Zabel introduce a approach for the programming of a 5-axis milling machine [7]. The operator programs the machine using the regular operating panel. Instead of milling real material, the milling process is simulated and the results are visualized online at the milling machine. This helps to prevent errors during the programming of the machine.

In the second case, AR is used for the interactive manipulation of a technical system. Here, visualization is used to control or program a technical system. Zäh et al. use AR to program an industrial robot [8]. The operator sketches a path, which is visualized using AR. The robot is able to follow this path. In [9] an AR-system for the remote operation of different technical experiments is presented. The visualizations represent virtual operation and display instruments, which are communicated to a real experimental setup. The user can use the virtual instruments to control the real experiment. Göktogan and Sukkarieh use AR to support the control of Unmanned Air Vehicles (UAV) [10]. The AR-application shows the sensor information of a UAV to the operator. This helps to get

a better understanding of the current situation. Furthermore they can use the AR-system to control the UAV.

Paderkicker

The Paderkicker are autonomous operating robots. To enable autonomous operations, a structured software architecture is necessary that facilitates intelligent decisions, based on sensor information. This section describes the logical software architecture of the Paderkicker, followed by a brief overview about the position estimation.

Logical Robot Architecture

The logical software architecture is structured according to the triple tower model of Nilsson [11]. That model distinguishes between perception, model and action tower (Figure 2).

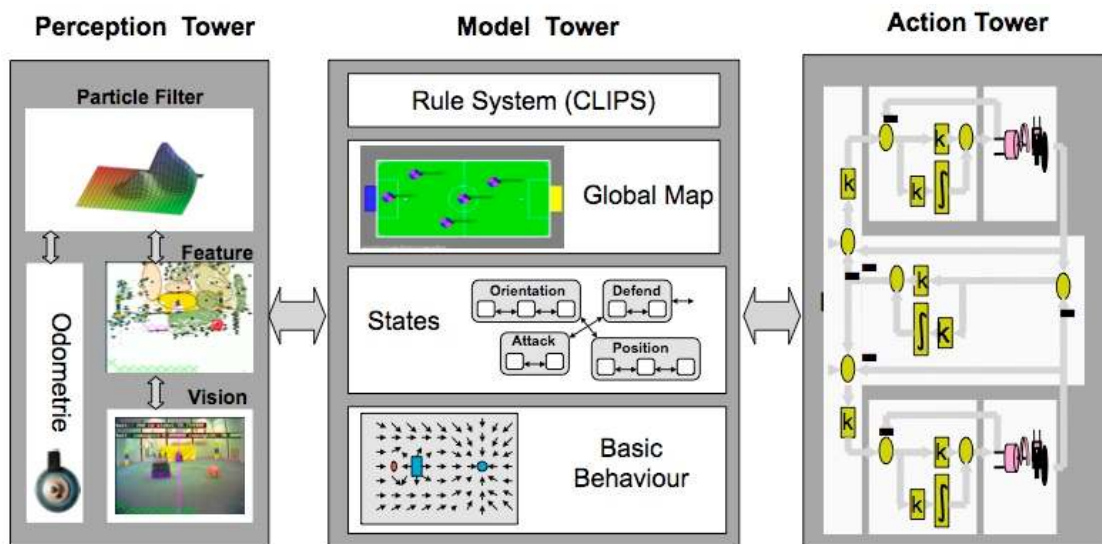


Figure 2: Paderkicker logical software architecture

The perception tower is responsible for processing and fusing the sensor inputs. It provides the input for the model tower. At the lower level, the feature extraction module processes raw image data from the camera and emits a number of features. For instance, the position of various objects like the goal, the lines of the field, the ball etc. is determined with respect to a robot. Also data from the driving system's odometric sensors are evaluated. The robots are equipped with four omni-directional drives (omni-wheels). Every omni-wheel is driven by a single motor. The sequence of rotary movements of the four driving motors is used to calculate the rotation and translation movement (odometry) of the whole robot. In the next higher level the information from

the driving system and the vision system is merged. Therefore, a particle filter is used (Bayes Filter).

The model tower is responsible for planning and selecting actions, to react appropriately in a certain situation. It is divided into various layers of abstractions. The lowest layer is the basic behavior layer. It takes care about the basic moves of like move forward, turn, kick ball and that a robot is switched off in critical situations. On top of the behaviour level, a reactive layer is realized using motor schemes, according to Arkin [12]. This reactive layer is controlled by the next higher level, which determines action sequences for a robot. Here, the playing strategy of a robot is planned and coordinated with teammates. This strategy is coordinated using a wireless network (WLAN); information about goals, opponents or other obstacles is distributed in the Paderkicker team. This information is used to set the role of each robot. Roles are e.g. attacker, defender, etc. At the topmost level, the world model of the robot is managed. It receives the data from the perception tower and form a model of, how the robot “perceive” its environment. The action tower realizes the control of the actor systems. Here the physical signals are generated in that form, needed by the actuators. Via specific hardware components, pulse-width modulated signals are generated and analogue values for voltage or current are measured. Cascaded PID controllers are used to realize current and revolution speed control for a single wheel.

From object recognition to robot behavior

The Paderkicker uses a vision system to detect objects at the field [RKK+06]. In general, the detection and recognition of objects is based on a color segmentation algorithm. The features of segmented color regions are compared with pre-defined features in order to identify objects or other robots. Recognized objects are checked for plausibility and deleted if necessary. Following objects are searched for:

- Field area
- Landmarks
- Ball
- Free Space
- Field lines

The object recognition system feeds the model tower with the necessary information to plan the next steps of the game. The behavior system is structured into the components world model, strategy automation and behavior module. The world model module stores

the information about the surrounding, recognized by the object recognition module. Depending on the teammates world module the strategy module decides the current strategy for every robot. Strategies include e.g. “Defend” or “Attack”. After the strategy module has committed to a role, an according finite state machine of the automation module realizes the strategy. Its states include e.g. ball facing or staying between goal and opponent. Every state in the automation module is mapped to a low-level reactive behavior like move to ball or kick ball in the behavior module. Finally, this module executes the actual actions.

The entire behavior of a robot depends on the object recognition of the vision system. If the vision system detects wrong objects, for instance an image at a wall as an opponent or a colored dressed person as a landmark, the localization quality will drop and a wrong robot or object location may be assumed. As a consequence, the robot shows an unexpected behavior. Without knowledge about the vision system perception, it is difficult for the developers to identify the cause for a problem. Most errors occur, if a robot estimates a wrong position for itself. This self-localization is performed in two steps. First, the robot detects the field with its lines and the field area itself. Detecting lines and the field is difficult, because the robot looks only to a small section of the field, and only a small section of a line is visible at anytime. Additionally, there is no visible relation between a certain line section and the field. The robot tries to recognize every visible line section and represents them using a particle filter model. Each particle is related to a line point. In the second step, the robot tries to map a model of the field to the detected line points. If the model meets the line points, the robot can estimate its own position accurately. Otherwise the robot does not know its own position. The latter happens, if the robot detects line segments that are not actually lines, but e.g. a person outside the field.

Application Setup

To support the developers of the Paderkicker, an AR application has been developed, which visualizes the line points, the internal field model, the particle filter data that represents the robot position, and the resulting current and estimated position of the robot on the field. AR shows the position of an estimated particle with respect to a real world object. Following, the hardware setup and software architecture of the AR system are described.

Hardware Setup

An overview about the used hardware is shown in Figure 3. At the left bottom of the figure, the Paderkicker soccer robots are located. At least, one soccer team consists of four Paderkickers. To coordinate its behavior with other team members, the robots communicate among each other via wireless LAN. To coordinate the whole team, a so called Team Server is used. The team server determines the roles of each robot and enables the referee to control the soccer game. A PC for the AR application is connected to the Team Server via LAN. Therefore, a Pentium IV, 2.6 GHz with 2 GB RAM and a NVIDIA GeForce 8800 GTX graphics card is used. As output device, an optical see-through HMD from Carl Zeiss is used. We use a monocular setup for the left eye. The HMD has a resolution of 800x600 dpi at 60Hz. It has a viewing field of 40° in diagonal and weights 100 g. It is connected to the VGA jack of the PC by an image correction unit. Because the HMD uses cables for the power supply and the VGA input, the user cannot walk over the entire field. Each cable can influence the vision system and is an obstacle on the field. The user can only move along one side of the field, but by now, this is enough to receive a good overview about the situation.

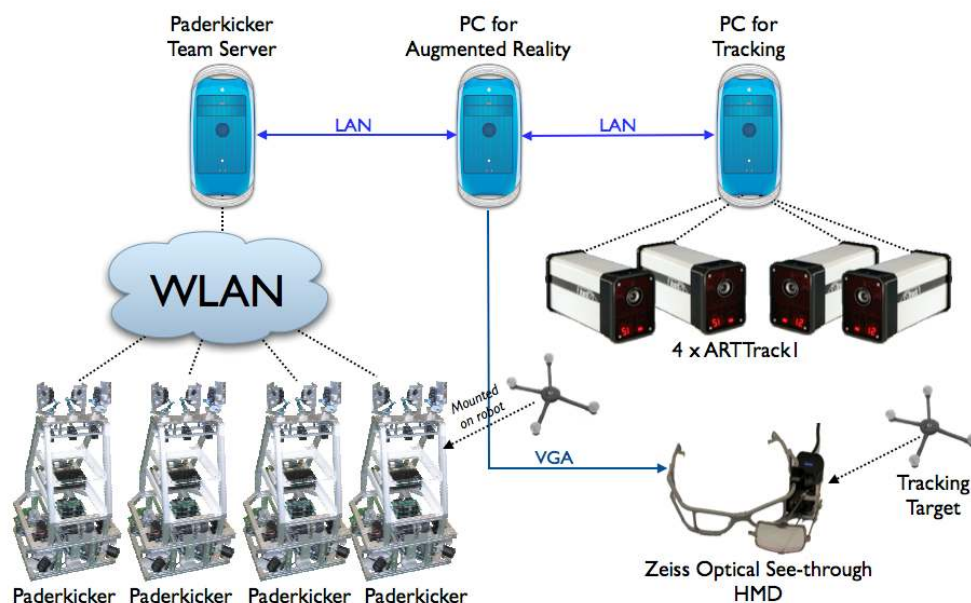


Figure 3: AR system hardware setup

For tracking purpose, an optical tracking system is used. We use an A.R.T. infrared tracking system; the setup consists of four ARTTrack1 tracking cameras. The cameras are mounted at the corners of the field (Figure 4). The dimension of the field is 5.5 x 7.5 meters; two ARTTrack1 cameras cover one half of the field. Unfortunately, there exist some tracking gaps, especially at the corners. The AR application tracks the user and

the Paderkicker robots. Tracking targets are mounted on each robot and facilitate the AR application to distinguish between each robot and to receive their position and orientation.

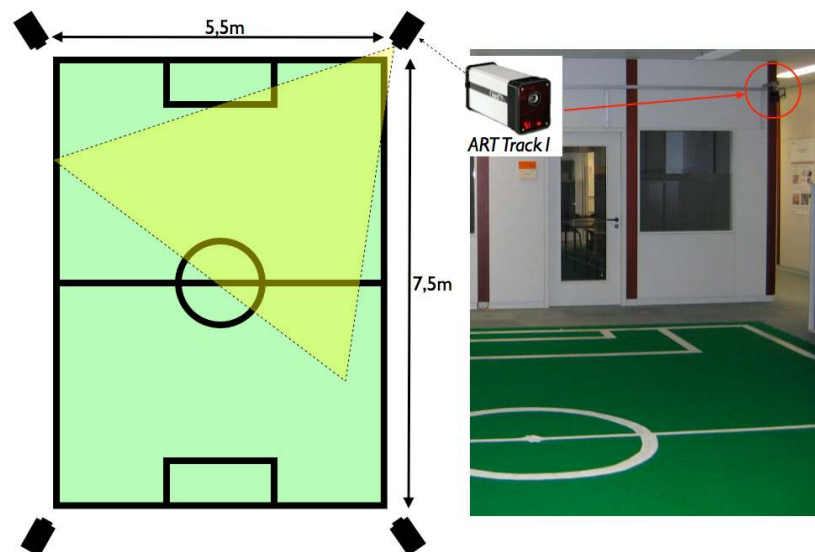


Figure 4: Tracking setup at the Paderkicker soccer field

A fifth tracking target is mounted at the HMD to track the position and viewing direction of the user. The tracking system needs its own PC, the tracking data is submitted to the AR application by LAN (UDP/IP).

Software Architecture

Figure 5 shows a schematic overview about the software architecture. It is divided into three main parts: The Paderkicker core software, the Team Server and the AR application. At bottom, the Paderkicker software components are located. Each Paderkicker is an embedded system; its controller operates under hard real-time conditions. Each Paderkicker submits data about its estimated position, the estimated ball position, its role, and some state data to the Paderkicker Team Server. Therefore, a wireless LAN communication is established, which uses the TCP/IP protocol. Because there is no direct connection to the actuators and the sensors of a robot, the Paderkicker Team Server doesn't operate under hard real-time conditions. Additionally it hasn't any direct influence to the behavior. The Paderkicker Team Server collects all the data, submitted by the robots and transmits it to the AR application. Therefore, two ways are used. The particle filter data are transmitted directly to the AR application using TCP/IP. The position data and the system internal states of each robot are transmitted via a communication server.

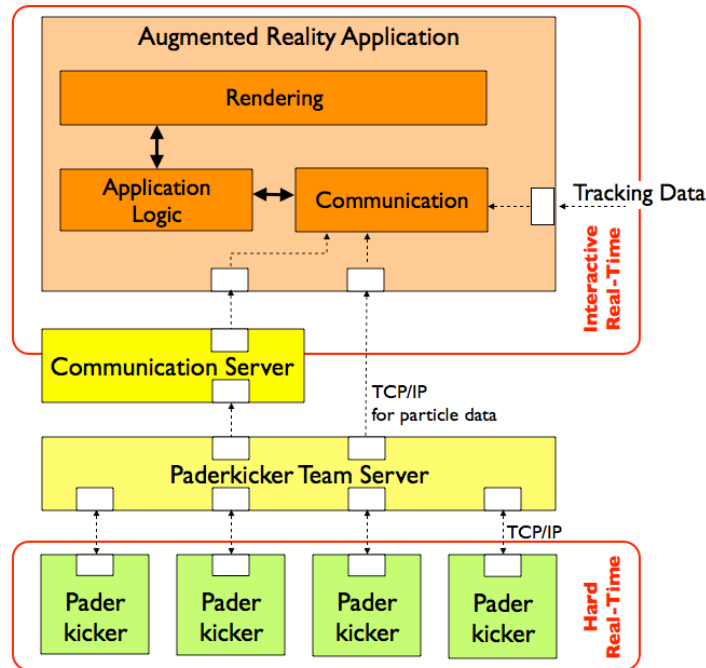


Figure 5: Overview of the software architecture

The AR application itself consists of three main components: A render component, a communication component, and an application logic component. The communication component receives and sends all data, which is needed to exchange between Paderkicker and the AR application. The application logic component controls the visualization of the data. It handles user inputs, switches between different visualizations and reviewed incoming data, whether a certain threshold is crossed. The rendering component is the main component for visualization. For rendering OpenSceneGraph is used [13]. Each visualization is represented as a scene graph and integrated into the main scenegraph of the entire application.

The AR application runs under frame rate conditions. Due to that condition, the application has to react immediately to every user's input. In opposite to the Paderkicker robots and the Paderkicker Team Server, the real-time conditions are not restrictive. The communication server is used as an interface between the components, working at real-time and the AR application.

Visualization of Object Recognition Data

To understand the behavior of a Paderkicker, the developers need to know their perception. Therefore, a set of visualizations is used. In the following, these visualizations are presented.

Field Estimation

One important task for a robot is to detect the field on that it's playing on. Therefore, the robot tries to detect the white lines on and surround the field. Internal, each detected possible line section is represented by a particle. At least, the robot uses between 10 to 100 particles to estimate the position and orientation of the field. To estimate the position and orientation, the robot matches an internal model of the field onto all detected particles. Figure 6 shows how the AR application visualizes the particle and the model of the field. Both figures show the view of a user through a HMD.

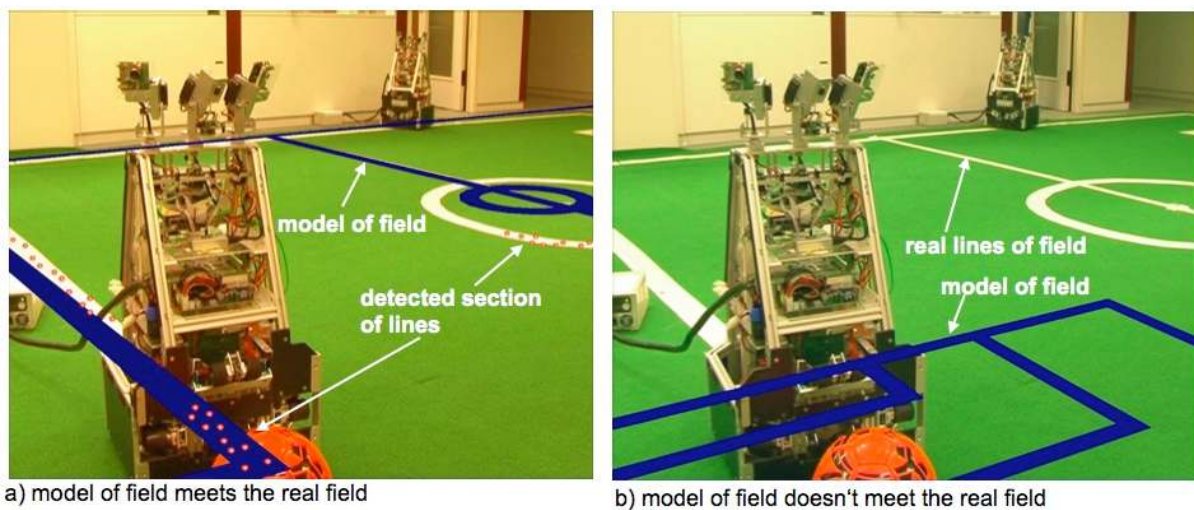


Figure 6: Visualization of the field estimation

In the left figure, a Paderkicker on the field is shown and how it perceives the field. The user sees red points and a blue virtual field. The red points indicate detected line sections. The blue model represents the robot internal model of the field. Both, red points and the blue model are visualized with respect to a robot. In this picture, the blue model of the field and the red points meet the position of the real field. This implies, that the robot detects the field section very accurate and is able to estimate the position and orientation of the field.

In the right picture, the blue model of the field doesn't meet the real field. It is visualized in front of the robot. In that case the robot hasn't been able to detect line sections, because no red points indicate any line. This results in incomprehensible moves of the Paderkicker.

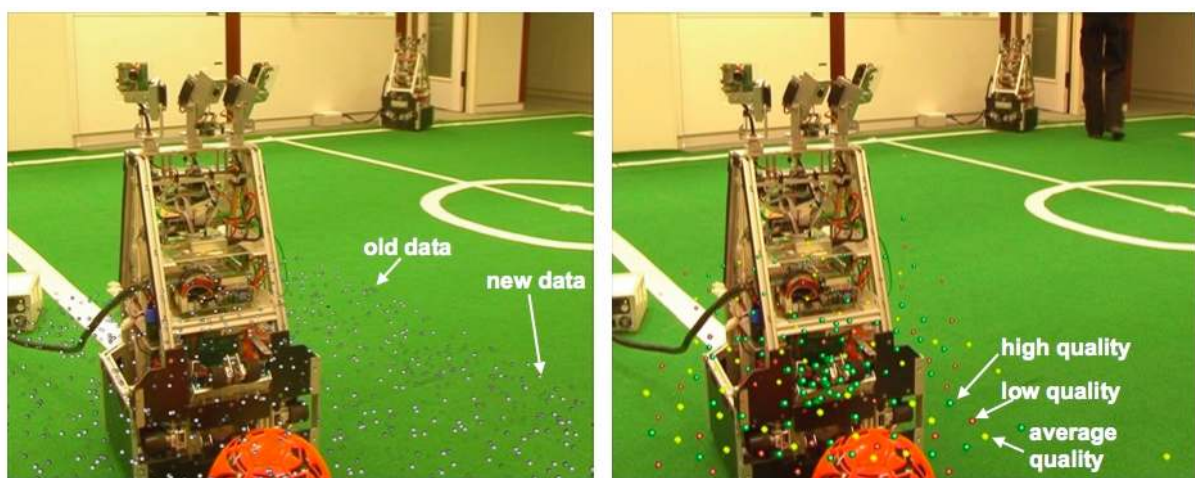
With this visualization, the developers see which line sections the robot has detected, which other objects in the environment it has detect as line sections and how good the model matches the real field.

Robot Position Estimation

After the robot has mapped his internal model of the field onto the detected line points, it estimates its own position and orientation. Therefore, a particle filter (Bayes Filter) is used to take into account the inaccuracy of the vision system and the odometry.

Figure 7 shows, how the user sees the data of the particle filter through the HMD. In Figure 7a), a robot is shown that is surrounded by many points. Each point represents a certain particle of the particle filter. Because, each particle has a different durability, new particles are shown opaque, old particles become bit by bit transparent. If a particle exceeds a time limit of 15 sec, it will be deleted. This helps to understand the changes during runtime.

Figure 7b) presents a second visualization of the particle filter. Each particle has a certain probability that expresses, how good the estimation of that particle was. This probability is indicated by a color. Three colors are used: red, yellow and green. For instance, red indicates a probability lower than 50%, yellow indicates a probability from 50% to 75% and green indicates a probability higher than 75%. The percent values can be modified during runtime and are not fixed. Only three colors are chosen, because color gradients are difficult to perceive through a HMD. The number of particles, used by the particle filter, is not limited. The AR application visualizes only up to 100 particles as colored points. If more than 100 particles are used, only 100 with the highest probability are chosen. If the distribution of the particles looks like in figure 7b), the estimation of the robot position and orientation works well. The robot is able to plan its next steps. Because of the vision and odometric noise, the particles drift away from the robot during its movements. This effect can be observed, too.



a) Transparency indicates age of the data

b) Three colors (red, yellow, green) indicate quality of position estimation

Figure 7: Visualizations of the particle filter data

Comparison of estimated and tracked position and orientation

Due to the data of the particle filter, finally the robot calculates one position and orientation. Therefore, it uses the most probable particles. To get a quantitative evaluation of the estimated position and orientation, it is compared with a tracked position and orientation. Figure 8 shows a visualization of the final estimated and tracked values.

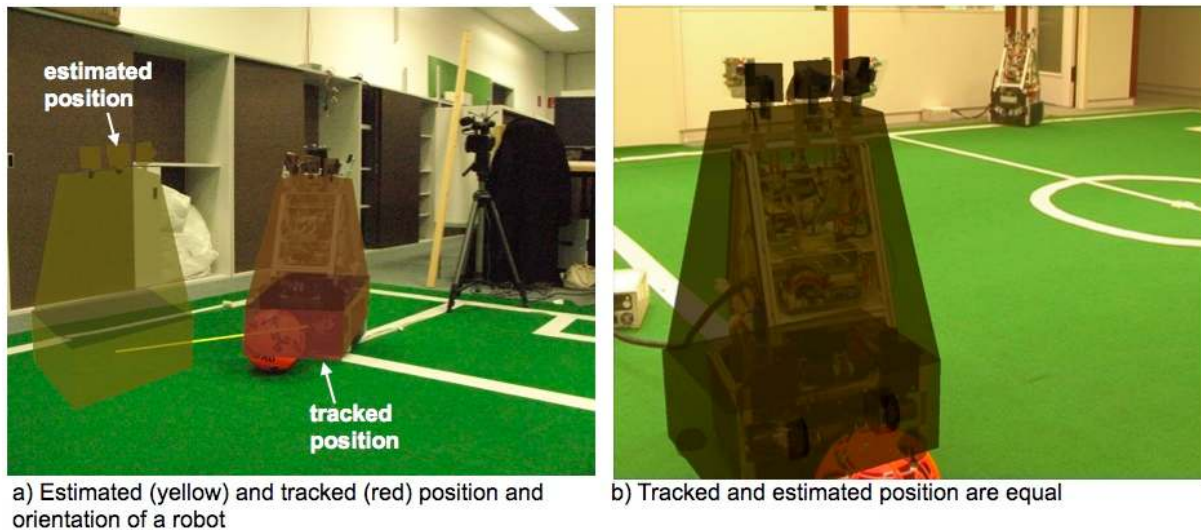


Figure 8: Comparison of the estimated and tracked position and orientation of a Paderkicker

A yellow bounding box visualizes the estimated position, a red bounding box visualizes the tracked position. If the position and orientation estimation works well, the yellow bounding box meets the red bounding box. In that case, the yellow bounding box disappears and only the red is shown (Figure 8b). If the tracked and estimated values don't meet, both boxes are shown (Figure 8a). A yellow line indicates the distance between them.

The presented visualizations aren't shown at the same time. A user is able to choose the visualization and data, he want to use.

Experimental Results

To get qualitative results about the usefulness of the visualizations, the AR application was tested by the Paderkicker developer team. During the test, several test persons evaluate the usability of the AR application and the developed visualizations. In the following, the realization of the test and the results are presented.

Realization of test

To test the AR application, we have prepared a common game situation: A robot should try to find a ball on the field and maneuver it into the goal. During this scenario, a user should look through a HMD and evaluate the visualizations. The different visualizations were presented one after another. Alternately, a monitor could be used. To get an evaluation of the visualization, the users have been asked to explain the meaning of the visualizations. Therefore, four questions have been asked:

1. Do you prefer a HMD or a monitor?
2. Explain the meaning of the visualized particles?
3. Explain the position and orientation of the virtual field?
4. Explain the position and orientation of the Paderkicker?

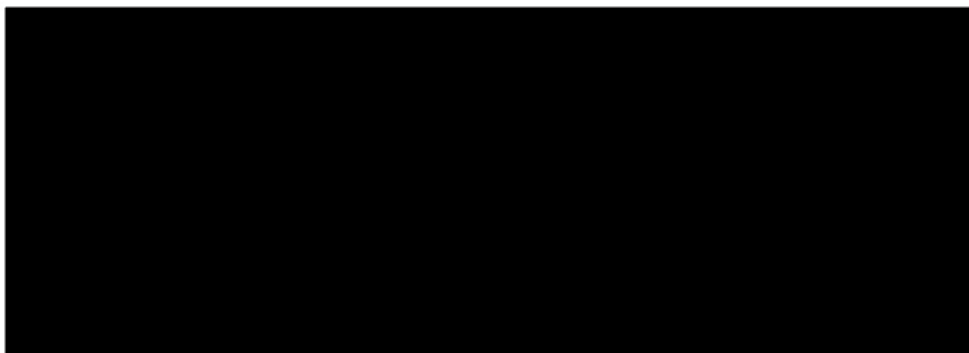
The answers of the users have been written down to a questionnaire. The possible answers for questions 2 to 4 have been: Understand, understand with problems and not understand. By question 1, the users could choose between a HMD and a monitor. They don't fill the questionnaire by them self, an assistant has done this. The users only have answered the questions.

Due to the autonomous behavior of a Paderkicker, it has been difficult to prepare repeatable test conditions. We can only ensure, that every robot starts at the same position, that the ball lies in the center of the field and that the starting states of a robot are the same for every test. Furthermore, the light conditions were not changed.

Results

Table 1 shows the results for the four questions.

Table 1: Results of the experiments



Most of the users prefer a monitor. Reason for this choice has been the limited field of view of a HMD. Due to that, it is not possible to see the entire field at the same time.

With a monitor and a camera, a much bigger field of view can be realized.

The meaning of the visualized particles (2) and the position of the robot have been recognized very well. The only problem the users have was to distinguish between the points that indicate the line section and the points that indicate particle of the particle filter.

The position and orientation of the virtual map was recognized badly. Reason for that is the limited field of view of the HMD. The users don't see the entire field of view, what made it difficult to evaluate the orientation and position. Especially, if the virtual field doesn't meet the real field.

In summary, the AR applications and the developed visualizations are suitable to understand the behavior of the Paderkicker. The test users common position have been, that the AR application helps to understand the perception of a robot.

Outlook

After the first tests of the presented application, we know the potential of AR for the testing and debugging of autonomous robots, in this case, of autonomous soccer robots. The first tests show, that AR helps the developers to understand the behavior of the robots.

But there are still several things that can be improved or added. First, the tracking system has to be extended. It is not possible to observe the entire field with only four tracking cameras. It is necessary to add two or four more cameras. In particular, there is no tracking at the corners of the field.

The presented visualizations can only be shown for one robot at time. But the Paderkickers have to act in a team. Therefore it is necessary to show the vision data from two, three or four robots at the same time. With the current visualizations, it is not possible to show the data of more than one robot.

The Paderkicker will be extended and improved by teams of students. One of these teams should use the AR application not only for tests; they should use it as a productive tool for testing and error localization. The information from them will be used to improve the AR application and the visualization.

References:

- [1] Richert, W.; Kleinjohann, B.; Koch, M.; Bruder, A.; Rose, S.; Adelt, P.: The paderkicker team: Autonomy in realtime environments. In Proceedings of the Working Conference on Distributed and Parallel Embedded Systems. (DIPES 2006), 2006
- [2] Azuma, R. : A survey of augmented reality. In: Teleoperators and Virtual Environments 6, 1997
- [3] Stelzer, R.; Steger, W.; Saske, B.: Effiziente Wartung komplexer technischer Systeme mittels Augmented Reality. In: Gausemeier, J. and Grafe, M.: Augmented and Virtual Reality, 6. Paderborner Workshop. Paderborn, 2007
- [4] Weidlich, B.; Benölken, P.: AR-Technologie zur Entwicklung von Produktionsanlagen. In: Virtual Reality und Augmented Reality zum Planen, Testen und Betreiben technischer Systeme. 9. IFF Wissenschaftstage, Magdeburg, 2006
- [5] Eckes R.: Augmented Reality-basiertes Verfahren zur Unterstützung des Anlaufprozesses von automatisierten Fertigungssystemen. PhD Thesis, University of Paderborn, Paderborn, 2007
- [6] Reinhäckl, T.; Schilling, M.: Augmented Reality Anwendungen im Werkzeug und Formenbau. In: Gausemeier, J. and Grafe, M.: Augmented and Virtual Reality, 6. Paderborner Workshop. Paderborn, 2007
- [7] Zabel, A.: Augmented reality based approach for the handling of complex machine tool. In Process Interactions in CAM-Programming. Technical Presentation in the PMI-Workgroup, CIRP January Meeting, Paris (France), 2007
- [8] Zäh, M.; Vogl, W.: Interactive laser projection for programming of industrial robots. In Proc. of the Fifth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR 2006), Santa Barbara, CA, 2007
- [9] Salzmann, C.; Gilet, D.; Hungurnin, P.: Remote experimentation: Improving user perception using augmented reality. In: NIWeek 2000, Austin (Texas), USA, 2000
- [10] Göcktogan, A.; Sikkarieh, S.: An augmented reality system for multi-IAV missions. In: Proc. of Sim-TecT 2005, Sydney, Australien, 2005
- [11] Nilsson, N. : Artificial Intelligence: A New Synthesis. Morgan Kaufmann, 1998
- [12] Arkin, R. C.: Behaviour-Based Robotics. MIT-Press, 1998
- [13] Webpage of Open-SceneGraph, www.openscenegraph.org, 2008

Authors:

Prof. Dr.-Ing. Jürgen Gausemeier, Dr.-Ing. Rafael Radkowski
Heinz Nixdorf Institute
Dr. rer. nat. Bernd Kleinjohann, Philipp Adelt, Willi Richert, Henning Zabel
C-LAB
Fürstenallee 11
33102 Paderborn
Phone: 05251 – 606228 (Rafael Radkowski)
Fax: 05251 - 606268
E-mail: rafael.radkowski@hni.upb.de