Praktikum "Mobile Roboter" mit Simulation und Telerobotik-Zugang (TELEBOTS)

Universität Hamburg
Fachbereich Informatik, Arbeitsbereich TAMS

Dr. Houxiang Zhang, Dipl.-Inform. Tim Baier

Zwischenbericht auf Englisch

Juni 2006

Abstract
A practical course “Mobile Robots” with simulation functions and Tele-robot-access functions for the Bachelor/Master studies at the computer science department (officially: Department of Informatics) at the University of Hamburg is presented in this report. The system design and realization are introduced in detail. The state of development is reported, and some difficulties and solutions during the process of development are discussed. Additionally, the remaining work is outlined.

1. Introduction
This practical course deals with a kind of edutainment robotics system whose object is to offer a chance to different levels of students to acquire knowledge about robotics. At the same time, this learning process will involve for the students, as they can design and build their own mobile robot according to their ideas and program and test it personally.
Many similar edutainment systems have been developed recently. For example, the LEGO Mindstorms is the most famous and popular in the world. Due to the comparatively favorable and durable hardware, the LEGO Mindstorms robot is used particularly for practical robotics courses. But the inefficient original RCX Controller [1] (only with three sensor entrances) and the special graphic programming environment of the Mindstorms system are the bottlenecks. This product is only suitable for teaching the student how to program, it is not efficient and complex enough for the students to understand the real principles of mobile robots. Another product is provided by Fischer technique [2]. The mechanical parts are similar to LEGO bricks but it is more expensive.
The Sony robot dog AIBO is a good prototype for the edutainment purpose. It is well designed and the programming environment is friendly. But it is also only a suitable product for teaching programming, not for understanding mobile robots. The the students can neither change anything about the hardware, nor does it feature an open architecture for testing different aspects of the mobile robotics system [3].

Three famous current practical robotics courses, using the original LEGO Mindstorms systems or Sony AIBO robot dogs, are shown below.

- Eric L Wang offered the robotic task and contests for students as an interesting beginning of the practical course in College House Enterprises, 2002 [4].
- The project “Roberta” based on Mindstorms system is promoted by the CBM BF/ Fraunhofer to pursue an alternative didactical beginning for encouraging girls to learn robotic technology [5].
- An example of a simulator using a LEGO system was developed at the Universität Paderborn. Only the original RCX Controller and its sensors are supported in this limited physics model. There is no possibility for this software to be used in our project.

Apart from several lectures we developed at TAMS, this new practical course "mobile robot" with simulation and Tele-robot-access will be used at the computer science department of our university. It is also suitable for professional schools or for further education. An ideal tutoring system for the practical course on mobile robotics is shown in Fig. 1. The students can build different mobile robotic prototypes based on their imagination. Using the software environment, they can program the robots and carry out simulations offline. Then they can download the soft code to the controller and test on-site. During this process, they have to deal with the sensor information and realize the various movement functions by controlling the motors. They will learn the sensorial technology and motor-control methods. At them same time, the students can overview the process using a web-camera and can interrupt in

case of malfunctions. The most important aspect is that the student will get a thorough knowledge of the principles of mobile robotics.

Furthermore, as a practical course at the TAMS group of the Department of Informatics, the following characteristics will be emphasized:

1. Development of high-quality contents in a new software infrastructure
2. Advance to the topics and typical problems of the mechatronics (programming, simulation, sensor evaluation, building of modular robots).
3. Conversion of the didactical method learning by doing.
4. Flexible access over a Web-based interface
5. Employment of the remote teachings, including remote access to the devices
6. Support by a Tutoring system with multimodal, proactive assistance.

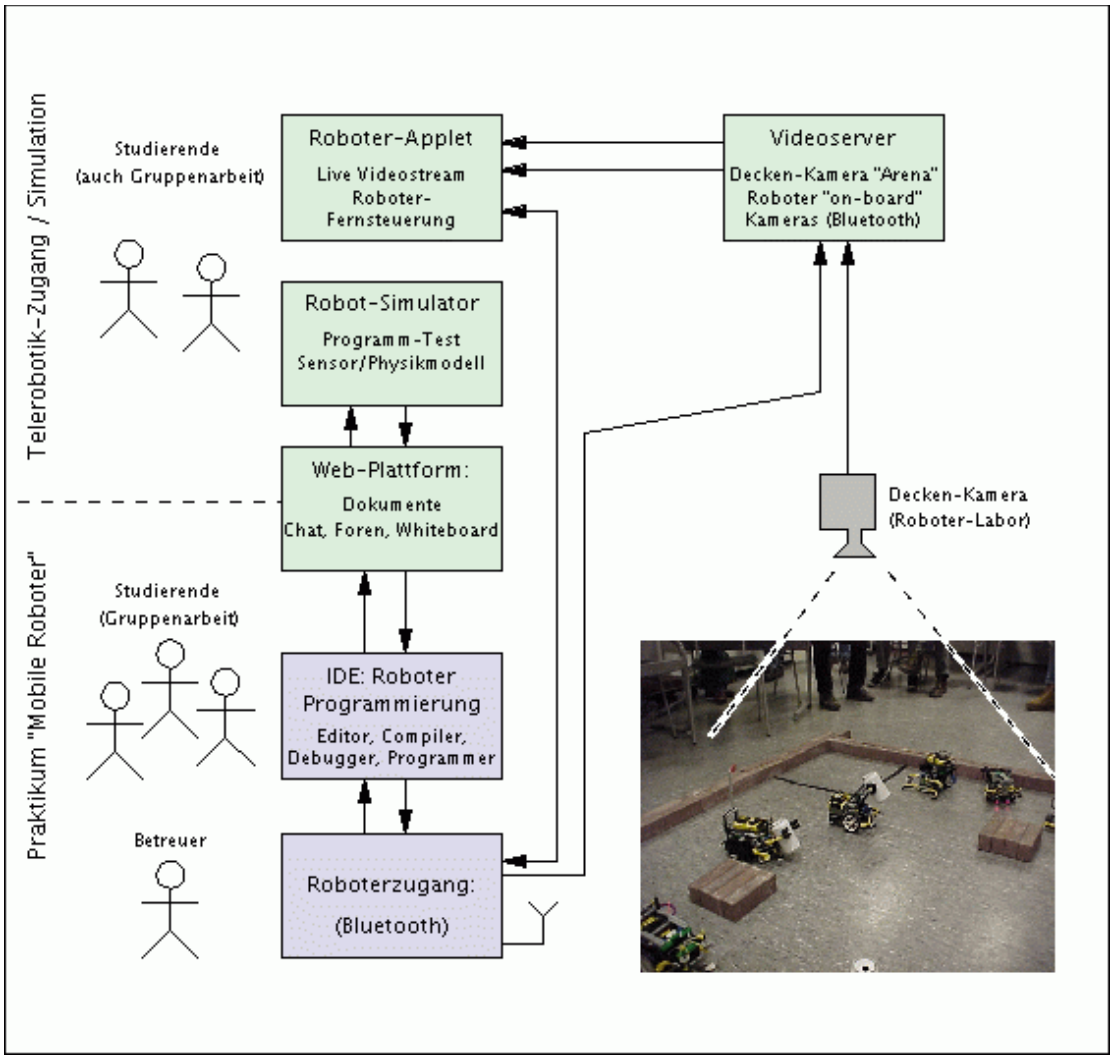

Fig. 1 Block diagram of an ideal tutoring system for the practical course

2. System Description

In our implementation, the "Mobile Robot" system is designed as shown in Fig. 2. The whole system consists of three parts: the mechanical parts, the hardware and the software. The mechanical parts are the physical background of the project. The hardware including the driving motors and sensors is the bridge between the mechanical parts and the software. All aspects will be introduced one by one.

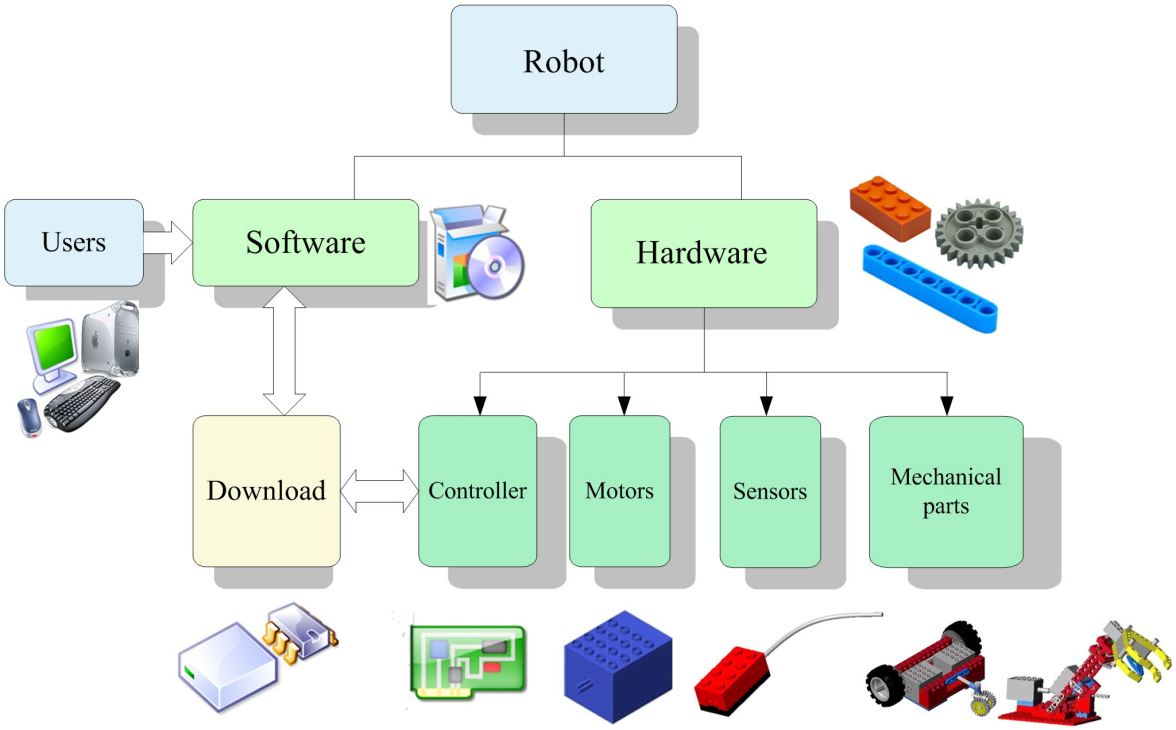

Fig. 2 Block diagram of our new tutoring system for the practical course

2.1. Scenario design

The first key issue in this project is the scenario in which the Telebot system works. Three aspects are important: dimension of the environment or scenario, the obstacle situation and how many mobile robots will work there.

2.2. Mechanical part

The best choice for this project is Lego bricks which can meet all requirements of functionality, extensibility and easy handling. As we know, the problem of LEGO bricks is that this company offers many products of which we only need a few. We have to choose the suitable bricks for our purposes in order to lower the cost.

2.3. Control hardware

The hardware includes the controller, senor inputs and DC motor outputs.

The original RCX Controller of the LEGO system is not efficient enough for our project since only three sensor entrances and two DC motor outputs are included. Meanwhile, the special graphic programming environment of the Mindstorms system is not convenient for university students who prefer to program in C++ or Java. As a conclusion, we will design a new control system which includes two layers: a lower level controller which will be in charge of collecting the sensor information and actuate all DC motors; and a higher level controller which will take care of global control functions such as wireless communication

with GUI, image processing, path planning.
Another problem as we mentioned before, are the motor output and sensor input of LEGO Mindstorms. They are not only limited and simple but also expensive so that we have to find some other cheap alternative. We should add other kinds of bricks for sensors and DC motors to the system ourselves.

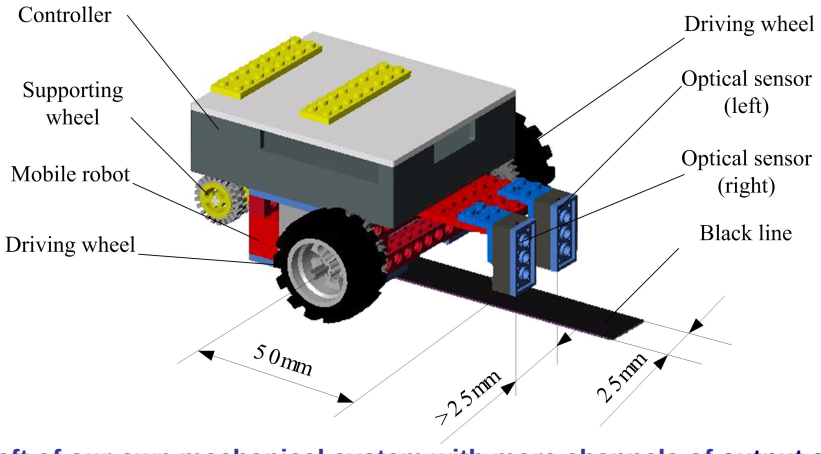

Fig. 3 Draft of our own mechanical system with more channels of output and input

2.4. Software requirements

The concept of the course leads to the following requirements for the software. First of all the software must provide a robust and stable interface to the robot hardware. The robots should be available 24/7[1] and for this reason the system has to be very stable. Secondly the system should be easy to use by the students. The system has to provide a remote programming interface. It should make no difference for the students whether they work directly in the laboratory with the robot or whether they program it from their homes. Furthermore, nobody, except the designer of the robot, should be required to program the sensor or the PID-motorcontroller for a robot. This requires the hardware to be hidden from the user by means of a hardware abstraction layer (HAL). Moreover the system should not be bound to a specific operating system but should support all of them.
All of the above-mentioned requirements motivate the design of a distributed system. One part of the software is dedicated to the user. The user is able to program the robot with a library providing the functions of the robot on a level above the hardware. This means the interface of the robot must provide specific actions like “move forward, move backward, turn left, turn right, get sensor value” etc.
The other part includes a robot server which sends the commands of the user’s program to the robot. This server has to be capable of running the robot even if the user’s program contains errors and causes failures.

A short recapitulation of the requirements follows:
- Robustness
- Easy to use
- Usable for students of 3. Semester (with JAVA experience)
- Remote programming functionality
- Remote programming and "presence" programming should be the same
- Should provide HAL

[1] It will not be possible that the robots are available 24/7 due to power consumption of the hardware, but the system should be designed like this is the case.

- Should run on at least on Windows and UNIX (Linux)

2.5. Task design

The purpose of this project is to teach students how to build a real robotic system themselves and to give them the confidence for future robotic research. The mechanical parts should be as simple as possible and should meet the requirements of objects in this project. Using the Lego bricks, some suitable sensors and simple DC motors, the students can realize several task functions step by step, from the easy ones to the much more complicated ones. Definitely, all of the task should be realized in the scenario we built.

7. Following a line
   The first task is to follow a line on the floor in the scenario. It is a black line, 20 mm wide and made of adhesive tape. The mobile robot should follow this black line and move forward from one end to the other end.
8. Looking for an object
   The second task is to hunt for a special object in the scenario. There is a burning candle on the floor whose position is uncertain. The mobile robot should find it automatically, move forward and stop in front of it.
9. Following a moving object
   Task three consists of following a moving object, for example a human hand, inside the scenario. A mobile robot is placed in the scenario, remaining immobile. It will only move forward if it detects something in front of it which it will follow till the object disappears.
10. Mapping the scenario & path planning
    The next task is to map the scenario and carry out path planning there. The scenario contains obstacles and looks like a maze. Firstly, the mobile robot should map the work space and draw a sketch map. Then the robot will be returned to the starting point. It should find the shortest path to get to the end point.
11. Cooperation between two mobile robots
    The last task is about cooperation. This task is the combination of all the functions above. The scenario is the same as with task 4. That means it is a maze with obstacles, but the positions of the obstacles are different from those in task 4. A group consists of two mobile robots, robot A and robot B which own different working abilities due to the different sensors and output functions.
12. Competition
    Furthermore, special competition can be developed, such as a 2 to 2 battle or Robocup.

Other possibilities are still open.

3. Implementation

3.1. The scenario

Under all these considerations, we designed the scenario as shown below. This scenario is flexible, easy to build and compatible with Robocup in the future.

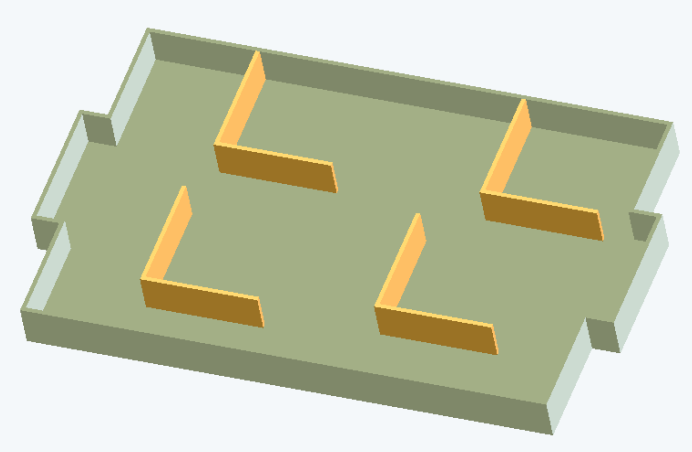

Fig. 4 Scenario design for the practical course

Dimension: 122cm X 183cm; Material: Wood;
Characteristic:

1. It contains 4 “L” Walls, whose positions are unfixed.
2. The floor of the platform is *White*, and the normal walls and “L” walls are in *Black*.
3. The obstacles can be rebuilt using “L” walls due to the standard shape. For example, a “T” obstacle can be built by two “L” s.

3.2. Mechanical system

Based on the cooperation with the group of ARMS at Beihang University (BUAA), we have designed our own DC-Motor Bricks and Sensor Bricks. All of our new sensor bricks and DC motor bricks have been double-checked and fit original LEGO bricks well. Fig. 5 shows the mechanical system which can be used to achieve different kinds of mobile robots.

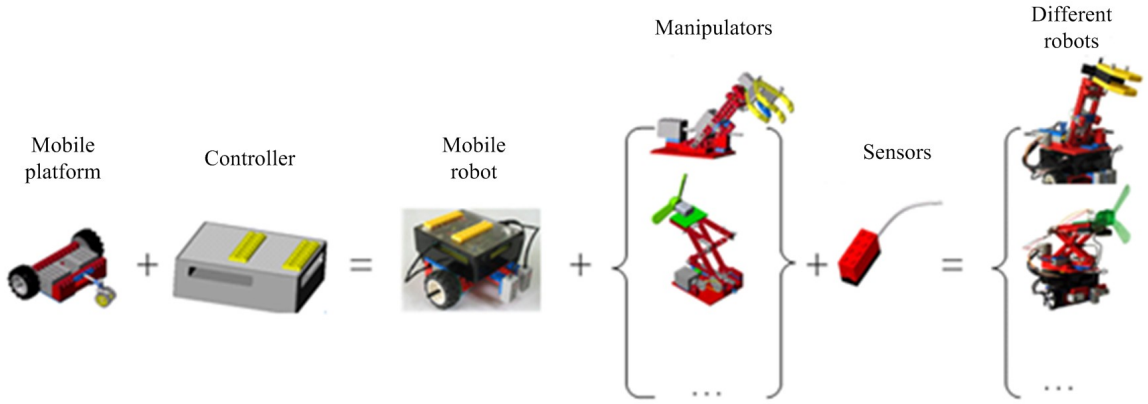

Fig. 5 The Mechanical system

3.3. Control hardware system

The specifications of the hardware are given below.
1. Enough I/O resources: 9 channels of sensor input which include touching sensors, infrared sensors, ultrasonic sensors and so on; the output channels are shown in the following table.
2. Network communication/ wireless communication
3. Power input 8-24v
4. Software downloads using RS232 or ISP
5. USB CCD camera
6. Extensibility for further research.

The students can program the soft code on their own laptop at home; then send it to the client terminal to simulate. They can control the mobile robot with online or offline modes in our lab. Here online means that all the control commands are sent to Card A from the client terminal and sensor information will be sent back during feedback using a wireless communication interface. Offline means the mobile robot will work autonomously; the students can interrupt the work of the robot once there is any malfunction. The images from the USB camera will be sent back to the GUI all the time.
The tutor will monitor all of the tests of the students and give them the necessary assistance.

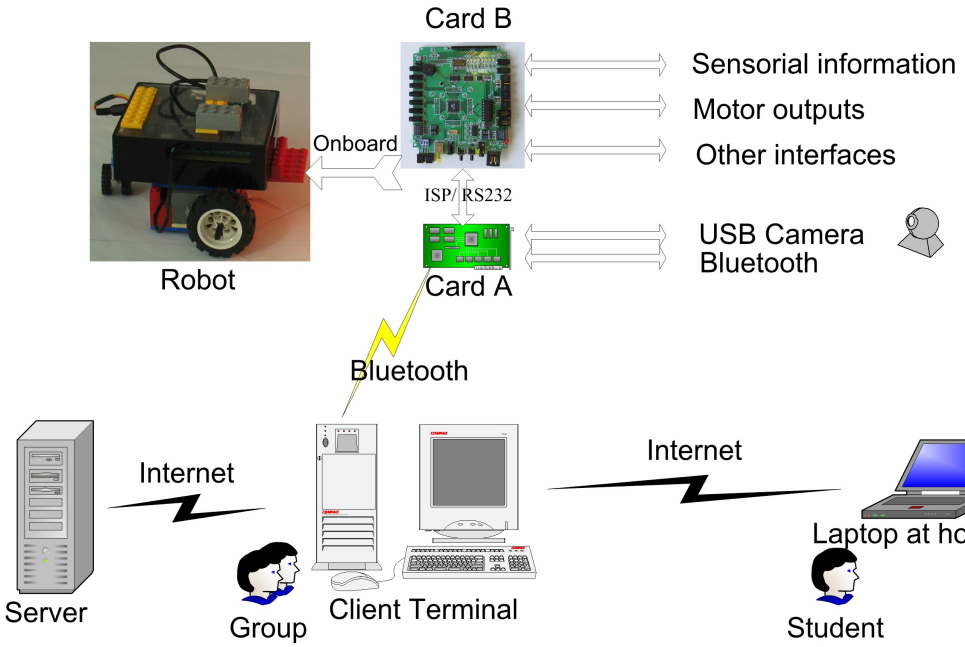

Fig. 6 Block diagram of hardware hierarchy

	Output				
Type	LED	Buzzer	Relay	DC motor (PWM mode)	DC motor (On-off mode)
Number	8	1	2	2 channels	2 channels

3.4. Software

The concept of the software is shown in Fig. 7. On the left is the user interface. It should provide a GUI for the overhead camera and the robot reservation system. Furthermore it should provide status information on each of the robots. The user is able to program the robot in the way he is used to, e.g. just with a standard editor or with an IDE.

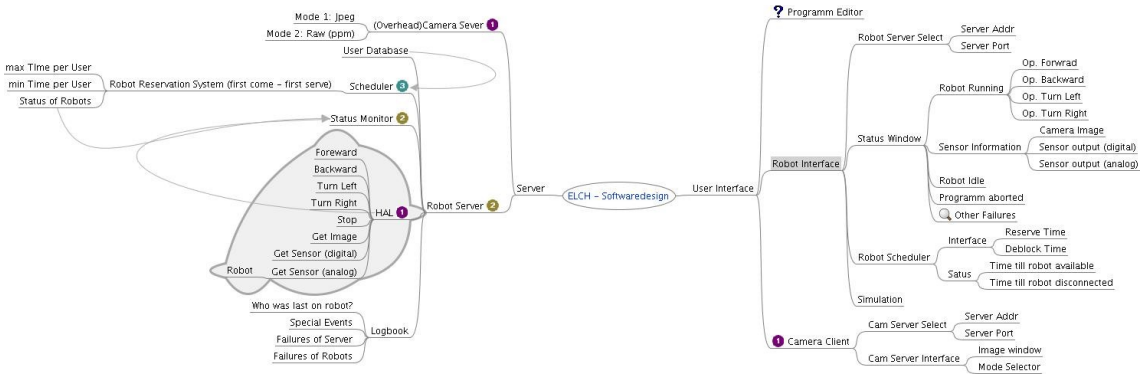

Fig. 7 Software layout

The o other side shows the robot server system. This system contains at least two servers. One is for the overhead camera and the other is used to get access to the robots. The overhead camera server just provides the images form the overhead camera. The robot server is a little more complex.

The main parts of the server should be: An interface which provides access to the robots. The idea is, that the students do not program the robot directly but use an interface or robot proxy. The commands from the students' program will be sent via wireless interface from the proxy to the robot and will be executed on the robot. This has the great advantage that the students can only use functions provided by the proxy. If the system designer makes sure that no functions on the robot can crash it, the students will not be able to crash the robot. Each robot should have its own proxy server, so that in case of the unlikely event that the program of the user crashes and blocks the proxy, the other robots will not be infected.

A scheduling system that organizes a timetable for each robot has to be set up, if there are more users than available robots. This includes automatic reservation and release as well as user-motivated release of a robot.

A user database which handles user access and privileges must be implemented. This gives the possibility to set up different user groups and give these groups special resources, an extra amount of computation time or a higher priority for the scheduling system.

A status server will be required to keep track of the statues of each robot. E.g. is a robot in use, is it idle, how long will it be in use if not released by "owner", battery power etc. A logbook which keeps track of the robots should be included, especially for failure and so on.

Robustness and HAL will be achieved by using a robot proxy.

3.5. Progress

To date, the framework for the whole system has been developed. The system functional design, the mechanical system and the control hardware system have been completed. The software hierarchy is on the way. Therefore the system can be run although it should be improved further. Fig. 8 shows the mechanical example with our own four DC-motor bricks and several sensor-bricks.

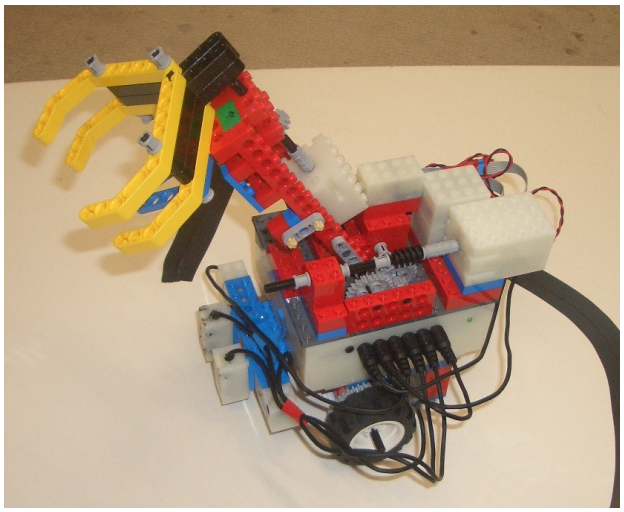

Fig. 8 An example robot using our own new bricks

Fig. 9 and 10 show the two layers of the hardware system.

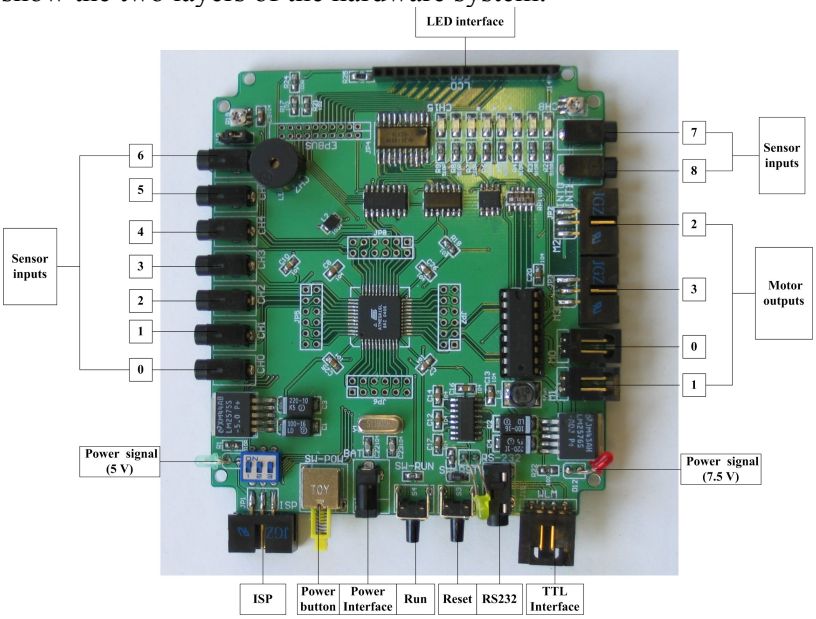

Fig. 9 The lower level controller designed by us

The specifications of the lower level controller follow.
1. ATmega16 is used as microprocessor.
2. The sensor channels from 0 to 6 can be used as digital or analog inputs; 7 and 8 can only be used in a digital way.
3. Power input should be 8.4V—24V
4. Two communication interfaces on the board: RS232 and TTL
5. ISP for downloading the drivers
6. Motor outputs 0 and 1 can be controlled by PWM signals; 2 and 3 are only under the on-off mode.
In our design, the higher level controller will manage the USB camera, the wireless interface between the onboard hardware and the client terminal, and the information exchangebetween the two layers of controllers on the robot.

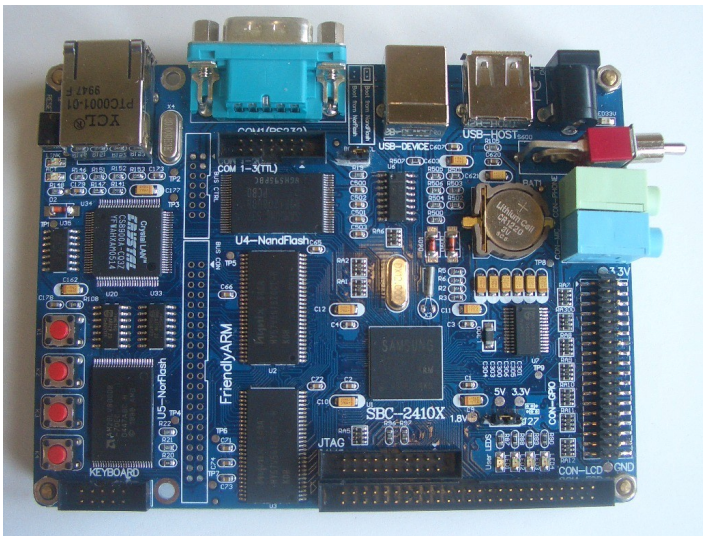

Fig. 10 The higher level controller with USB, RS232 and network interfaces for global functions

Fig11. shows the first test version of the software interface for programming the controller .

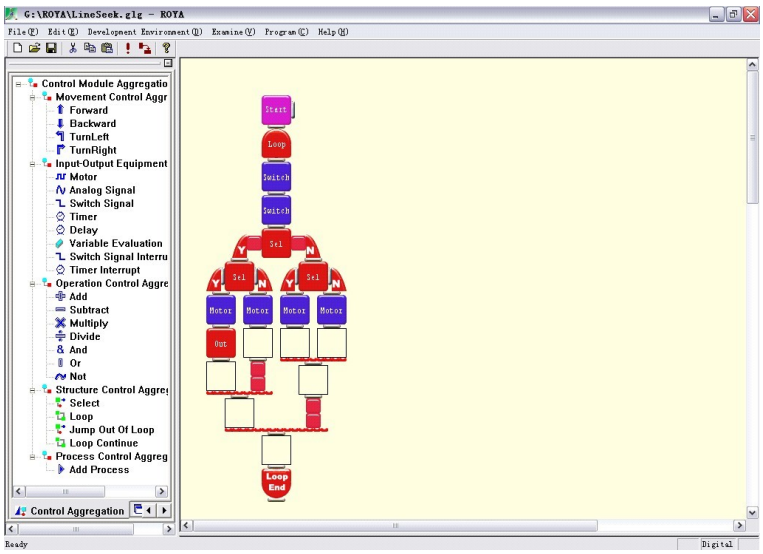

Fig. 11 The first test version of the GUI

The drivers for sensor information collecting and DC motor actuating are under development.

3.6. Difficulties and solutions

1. The usage-stability of our own new motor and sensor bricks

The new bricks we designed still need to be improved since we are not experts on LEGO bricks. Currently the new bricks are 95% stable for use; especially sensor bricks sometimes do not work perfectly since we only use cheap sensors in this project due to the financial limitation. It is a normal problem of sensor technology. We are still trying to make sure that all of the sensors work stably all the time. It is gratifying that the DC motor bricks are stable enough by now due to our efforts.

2. Image transferring and wireless communication

According to the requirements of this project, the Tele-access capability should be met. The Bluetooth or RS232 is used as a wireless communication device considering the velocity and the data transfers. The CCD input is not used for image processing; the onboard microprocessor only sends the images to another GUI by a wireless communication interface at first step. Even if our hardware system can realize all these functions, at the moment, we still wonder if the communication velocity is fast enough, which depends on the tasks. Furthermore, as a tutoring system, it is very hard to realize the image analysis on time. All of the images will be sent to the client terminal to analyze.

The higher level controller has a network interface. So we can use it for transfer data if necessary.

2. Software realization

As mentioned before, in our system the students will program on their own computer and the robot server will send every simple command to the hardware on the mobile robots and get feedback including sensorial information and other system status information in real time. This way, the onboard drivers should be powerful enough to realize all of the functionalities as described in task design. The drivers should be modules, easily extendible and transparent. Totally, the drivers include all the below functions:

Initialization, Reset, Interrupt, Move forward, Move backward, Turn left, Turn right, Movement Stop, Fan start, Fan stop, Sensor feedback, Status feedback.

As an easy-to-use software for developing distributed systems, the GenRob (R) and Roblet (R) platform seem to meet all requirements (for details see www.genrob.com[6] and roblet.org respectively). The robot server (robot proxy) can be implemented as a RobletServer on which the students can run their programs. The programs can be written in JAVA which should be known and easy to learn by each student. If a student program crashes, the RobletServer will remove it from schedule and stay alive. This enables a high grade of robustness and stability. The GenRob (R) and Roblet (R) system [7] are implemented in JAVA. Therefore it will be possible to run the system on every hardware for which a JRE 1.4 or higher is available.

If the students are able to program the robot directly and the robot hardware is powerful enough to run a JAVA program directly, it is easy to move the proxy directly onto the robot and provide more functions to the user.

The software on the client terminal includes three aspects: a robot library which contains the definition of the robot programming interface, a robot simulation and a GUI for the overhead camera.

The students are programming the robot in JAVA and compile their programs with a standard JAVA compiler. They will be able to choose whethertheir program should be executed directly on the robot server and control a real robot or if a simulation of their robot should be started.

The simulation part includes five aspects: scenario building, robot building, task description, simulation and summary. The students can build the experimental scenario and mobile robot at home before they test their ideas using the real system. It is very important for the students to attend this part of the lecture due to their little experience on robots at the moment. They can test whatever they want in a visual environment. Step by step, they will clear their ideas and know how to realize some special functionality.

The GUI is the last part consists of commands input area, sensorial information area, status information area, CCD camera and general information. The students can start a task automatically or send some commands directly to the hardware onboard. STOP button is used to cease all problems during the task realization. All of the outputs of the sensors are

shown in Sensor information area in a real time. Meanwhile, Text information about the system status will be shown to the students. The images from CCD cameras are used for monitoring at the first step. In future, we can analyze them for furthermore information.
Currently the user database server including a user-administration GUI and user verification interface have been implemented and tested. The administrator is possible to add and delete and modify users as well as managing groups.
The scheduling system has been implemented in parts and is still under development. Currently a user can reserve a slot for a robot and delete it. Furthermore the system administrator is capable of modifying time slots and deleting them, as well as reserving time slots of arbitrary length for maintenance.
For testing a simple robot proxy has been implemented. This robot server is able to control an AIBO by sending motion commands to it via a wireless network interface.

4. Future work

Based on the currently achievements, we will focus on the difficulties, especially on implement of the software part later.
Firstly, now we will try some more stable sensors for our project in order to give a stable system for student to test and meet all of the requirements as mentioned before.
Secondly, the communication interface part will be paid attention according to the different tasks. At the same time, we will keep the possibility to analyze the image data for the future.
The software will be extended. The scheduling capabilities must be improved by means of automatic rescheduling if a robot turns out e.g. The robot server has to be extended that not only one robot can be accessed at one time but multiple robots.
The simulation part in the software is very important for tutoring system. It will be a focus in this project from now.
All above, the following work will be served to the system integration.

5. Conclusion

With the support of ELCH, a practical course “Mobile Robots” with simulation functions and Tele-robot-access functions which is for the Bachelor/Master studies is being currently developed. This report describes the system in detail and discusses difficulties and solutions. The system can run now and is being improved.

Reference

1. www.legomindstorms.com
2. www.fischertechnik.de
3. www.joker robotics.com
4. www.aibo europe.com, openr.aibo.com
5.tams-www.informatik.uni-hamburg.de/applets/hades/html/hades.html
6. www.alex.ais.fraunhofer.de/zeno/web?action=content&rootid=15465
7. www.genrob.com
8. roblet.org