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

Universität Hamburg
Fachbereich Informatik, Arbeitsbereich TAMS

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

Zwischenbericht auf Englisch

January 2007

Abstract

As we know, a robot is the best way to bring students in contact with technology. With this practical course, students can learn how to use the basic components such as a servo motor, different sensors and mechanical parts (LEGO bricks) to build their own robot (hardware parts) for special tasks, then use either a GUI component editor or a high level programming language (Java, C, C++) to write a program.

Because this "education telerobot" should be used by students who come from different study levels, the required software must be more flexible to use than the old the existing systems (such as LEGO Mindstorms or Fischertechnik). For senior students Java or C/C++ can be used to build a "robot" object on the software level; while the junior students who have no experience of programming can use a simple iconic interface editor to create their own program. Moreover, the software structure of our system should be improved to make the robotic system more flexible and extensible.

The programming part of the "education telerobot" consists of two different levels. The high-level program which runs on a PC or the embedded operation system (Linux) and the low level-program which runs on the micro controller (ATMEGA16L).

1. Introduction

Robots exist not only in "high technology" but also in the education field. In modern schools, a "robot" is not a concept from books anymore. With education robots, students can learn a lot about technology by building their own robot and writing a program for special tasks. Actually, education robots are very popular in Europe and the USA. There are several different types of education robots. Two famous companies in this field are LEGO and Fischertechnik. LEGO (originally through a partnership between LEGO and MIT media Lab) have released a type of programmable bricks: LEGO Mindstorms Robotics Invention System (RIS) with servo motors, sensors and robotics Command Explorer (RCX), which make it possible to add functions or behavior to physical creations made with LEGO pieces. The education version of this production is called "Lego Mindstorms for school" and comes with ROBLAB GUI-based programming software, which has a very friendly interface and is easy to use. Fischertechnik is a German company that develops toys for building models. The mechanical parts and the software parts are similar to LEGO bricks but they are more expensive.

This report deals with a kind of edutainment robotic system whose object is to offer a chance to students of different levels to acquire knowledge about robotics. The learning process will involve design and how to build their own mobile robot according to their ideas. Our systems is defined by two important characteristics: Firstly, the teaching materials and the course are brand-new, which means that we have no prior experience in this area; secondly, the teaching materials are tied to the course. Our efforts will concentrate on developing a cheaper educational robotic system in order to meet the requirements of flexibility, functionality, extensibility, easy handling and low cost. To reach these goals, we should not only improve the hardware, but also the software for this system. The intention of this report is to discuss the software structure, therefore it will handle the hardware improvement of the "education telerobot" very roughly.

2. System Description

The new system consisted of three parts: mechanical parts, the hardware and the software (Figure 1).

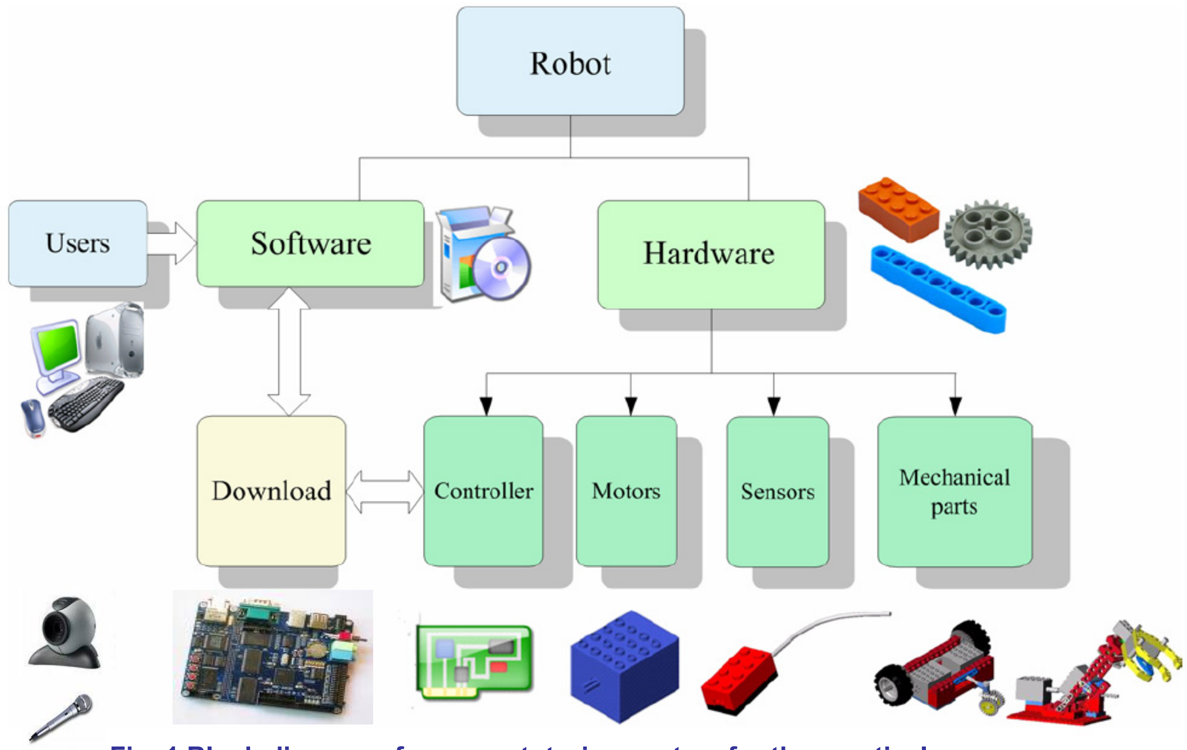


Fig. 1 Block diagram of our new tutoring system for the practical course

2.1. Hardware

- *Micro controller board (ATMEGA16L chips circuit).*
  As mentioned above, the original RCX Controller of the LEGO system is not efficient enough for our project, since only three sensors provide input and only two DC motor provide output. High-level students should work on more complex tasks. So more motors and more sensors are the basic prerequisite. The new controller board of the "education telerobot" can process input from nine sensors and it can drive two PWM DC motors and two DC Relay motors at the same time. It can also run the communication program, which is used to contact the high-level program on another computer.
- *Sensor.*
  The "education telerobot" has four different analogue sensors and digital sensors.
- *Motor.*
  There are two PWM motors and two relay motors.
- *Embedded System Board - Samsung ARM9 chips based SBC-2410x*
  This embedded system board is nothing else than a small PC. It runs a Linux (Kernel 2.40) operation system and provides a universal hardware interface such as USB (host/slave), sound output/input, LED monitor output, TTL/RS232 serial port and Ethernet interface. This board supplies a powerful platform. Firstly, it can be extended by a lot of hardware through its standard interfaces, such as a CCD camera, a wireless network etc. This is also the prerequisite for complex jobs, for example "environment analysis with image processing” "(through a web camera) or "Football playing with robot team" (wireless network communication). This is the second reason for us to give up the LEGO RCX Concentrate

system. For our "education telerobot", we need a flexible and extendible robotic system.

2.2. Mechanical part

The mechanical part consists of normal LEGO bricks, which normally, the students have at least some experience of from their childhood. LEGO bricks provide a popular platform for students to build a good mechanical system with any mechanical structure they want. As we know, the mechanical part of a robot is quite important. The platform made of LEGO bricks is very flexible and extendible.

2.2. Software part

Software consists of a multilevel hierarchy structure. The Software hierarchy is the main subject of this report, and we will discuss it in the following section.

3. Software Hierarchy

3.1. The scenario evaluation of the old robotic system

Let us firstly have a look at the Software Structure of the old system (LEGO Mindstorms and Fischertechnik products). If we buy LEGO Mindstorms, it includes software to edit the task program. It is an iconic GUI software that runs on normal PCs and is really quite easy to use. Students assemble the GUI component to describe a task and the editor program can do the rest. After writing the program, students can download it from the PC to the robot controller. The process is depicted in Figure 2.

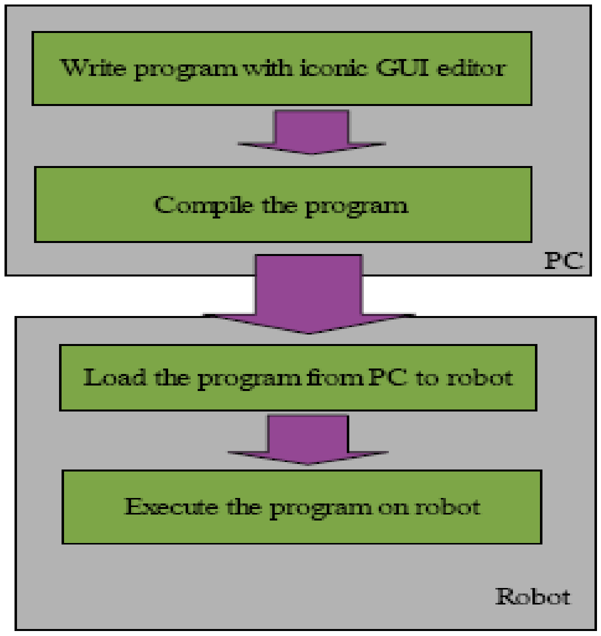

Write program with iconic GUI editor
Compile the program
PC
Load the program from PC to robot
Execute the program on robot
Robot

Fig. 1 Development process using LEGO system

This development process is relatively easy, but the the following disadvantages still need to be improved.

Flexibility.
The iconic GUI interface is not flexible enough. The purpose of this new robotic system is to accommodate more students who come from different study levels. The junior students normally have no experience of writing programs. The iconic GUI interface is quite good in helping them to write their program quickly and easily. On the other hand, the senior students should use their programming experience and knowledge to develop their own more complex program. Therefore the iconic GUI interface may be a limitation for them. In addition, using a high-level program language such as Java or C/C++ is also an important skill of modern robotic technology. Students should learn and exercise high-level program languages on this new education telerobot system.

Security
Students are allowed to use features of the hardware. It should make no difference to the students whether they are working directly in the laboratory with the robot or programming it at home. Nobody, except the designer of the robot is permitted to program the lowest-level drivers, such as the sensor or the motor controller of 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 any specific operation system.

Easy, handling
Students have to load the program again if they want to change the robot's task using LEGO system. It is a really big limitation for the robot to acquire artificial intelligence.
In our system there is a new hardware part, called embedded operation system board (SBC-2410x), which can be extended with different hardware. The system has to invest system resources to control this extended hardware, and more resources to analyse the information. For example, if we add a web camera, we should have an image processing program to analyse it, or if we use WLAN to build a robot team, there should be a program to control cooperation within the team. If the program only runs on the micro controller, it cannot run complex operations as fast as a normal computer. This is also why we cannot run “intelligent” programs quickly on this old robotic system. It cannot change its task or status after analysing complex environment information.

Remote control.
After downloading the program to the robot, it becomes a closed system. It means the robot gets the information through sensors and analysis data inside the micro controller, and then sends the reaction commands to the motors. So we can say, it is not a remote controllable robot since we cannot control it any more after downloading the program. Maybe this is enough for a toy, but not for our telerobot. Normally modern robots can run not only as a closed system but can also be controlled by a host station. With a closed system we cannot get any information from the robot so that there is no "parallel simulation" to describe the robot's environment on the GUI Interface.

Simulation interface.
For education robotic systems, the simulation is an important part. If the system has a good simulation interface, students can test their design (the robot and its program) before they run it on the real robot. With a good simulation, they can find design mistakes, or find out in advance about problems they overlooked. Furthermore with a simulation interface, students can also build their "visual robot" on their PCs at home as well as test ideas in the laboratory.

3.2. General software hierarchy of the new robotic system

We have based the new software hierarchy for our telerobot system on these five criteria:

Improved programming interface
The right part (Fig. 3) of the software is dedicated to the user who is able to use a library providing

basic functions of the robot to program. This means the interface of the robot must provide specific actions like move forward, turn left etc. A C/C++ library implementing these low-level functions has been built. Students can use the C/C++ language to use the functions directly from the library to build their program. Additionally, we have made a Java interface library using Java JNI technology for the students, who are in favour of the Java language. Meanwhile we also have an iconic GUI for the junior students who have no experience of programming with high-level languages.

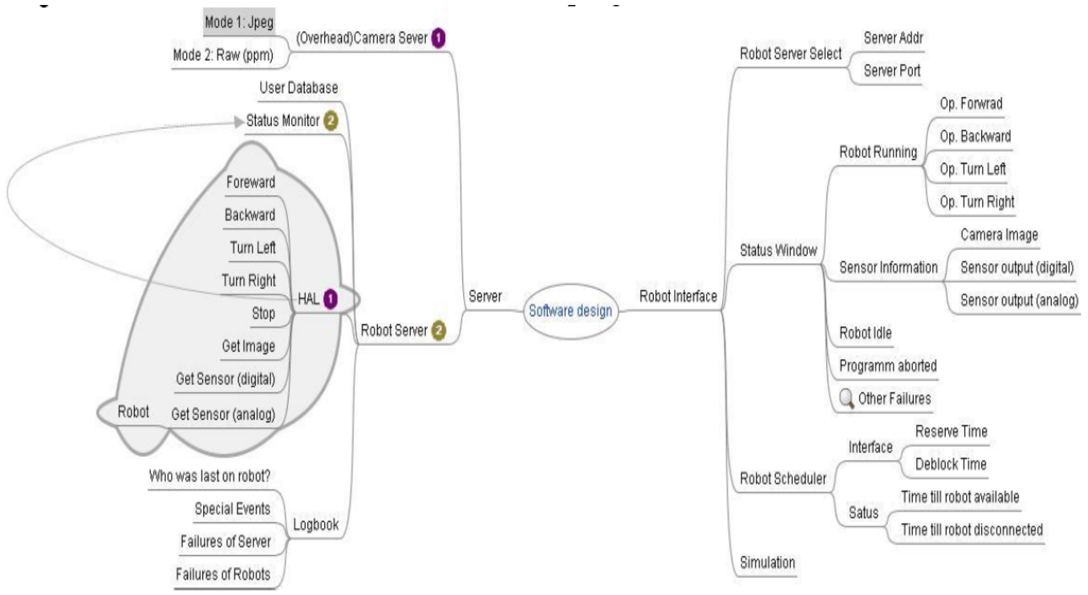


Fig. 2 Software Hierarchy of the new education telerobot system.

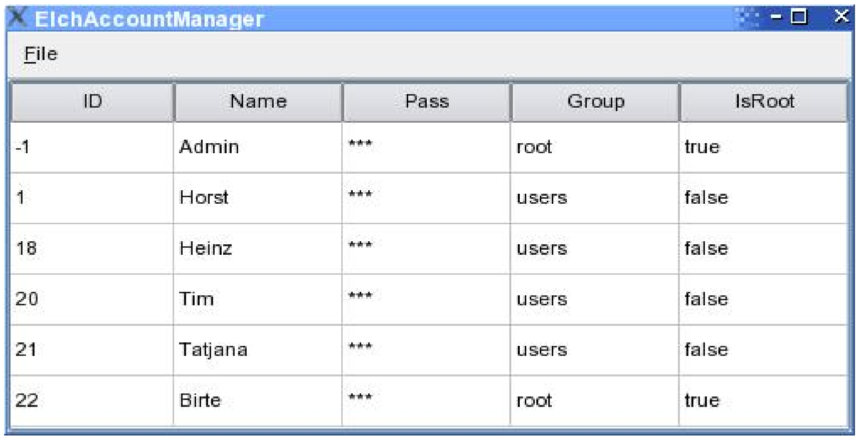
One major advantage of the Java library lies in using the GenRob technology. The GenRob technology provides the possibility to set up a robot system as a distributed system where the network is transparent to the user. It also allows students to program the robot from any computer, regardless of whether they are directly sitting in front of the robots or have established a connection via the intranet at home. In this case the software system consists of two major parts. One is the robot server (Roblet-Server), also named robot proxy, which organises the commands to the robot. The other parts are the programs written by the students. They are programming using a client library which provides an advanced programming interface (API) to program the robot. These programs (Roblets) are automatically sent by the GenRob system to the robot server and executed.

Improved security

On the other side of Figure 3 you can see the robot server system. The main part of the server provides an interface to access the robot. The idea is that we do not let the students program the robot itself but they have a kind of 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 cause a crash, the students will not be able to crash the robot.

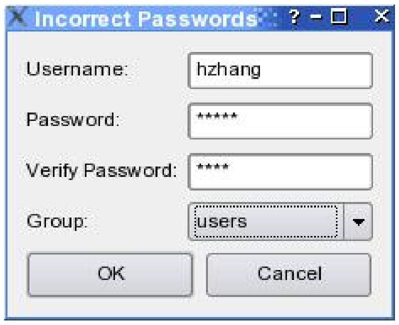
The security is also improved by the usage of one robot proxy for each robot. By this the robots can be shut down, maintained and restarted individually. In case one robot crashes the others are not infected.

Furthermore, by the introduction of a user management system, it can be ensured that not everyone has access to all robots. Access rules can be set up like in a modern operation system. A supervisor management tool has been implemented to simplify the management of the user database. Some screen shots are shown in Figure 4 and 5.



ID	Name	Pass	Group	IsRoot
-1	Admin	***	root	true
1	Horst	***	users	false
18	Heinz	***	users	false
20	Tim	***	users	false
21	Tatjana	***	users	false
22	Birte	***	root	true

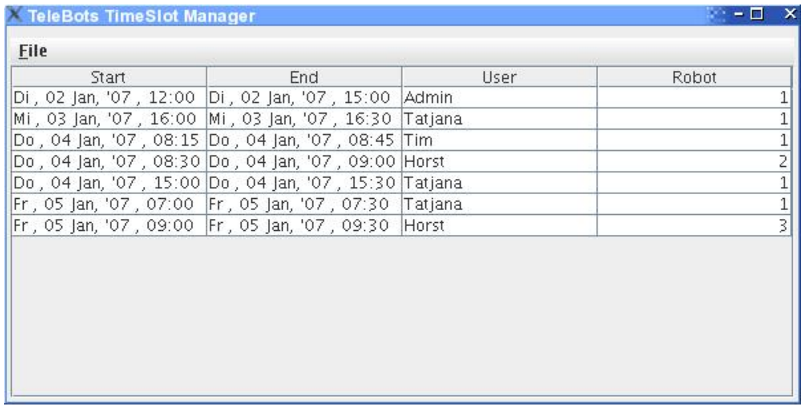
Fig. 3 Account manager showing the user IDs, Name, a masked password, group and supervisor status



Username:	hzhang
Password:	*****
Verify Password:	****
Group:	users

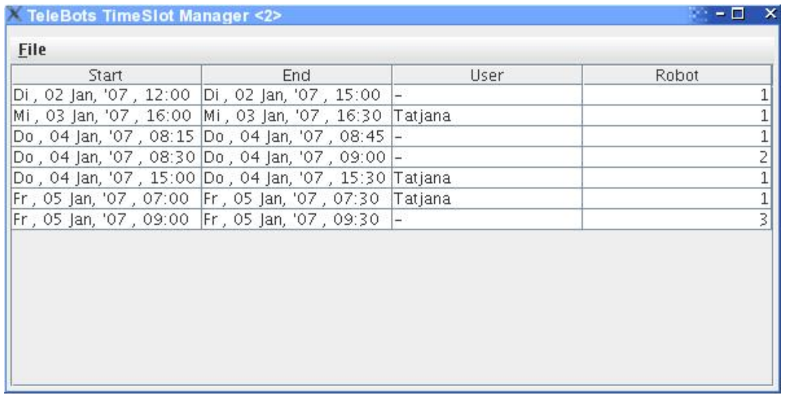
Fig. 4 Add-user dialog

A scheduling system organizes a time table for each robot, in case there are more users than available robots. This includes automatic reservation and release as well as user-motivated release of a robot. The users have to request for a time slot if they want to use a robot, which ensures that only one user at a time is able to use a robot. The supervisor has the ability to block or suspend robots for maintenance or in case of failures. A tool has been implemented to simplify the access to the scheduling database. It is shown in Figure 6 and 7. The scheduling system is also connected to the user management system. This allows the supervisor to install special scheduling rules for each user or group.



Start	End	User	Robot
Di , 02 Jan, '07 , 12:00	Di , 02 Jan, '07 , 15:00	Admin	1
Mi , 03 Jan, '07 , 16:00	Mi , 03 Jan, '07 , 16:30	Tatjana	1
Do , 04 Jan, '07 , 08:15	Do , 04 Jan, '07 , 08:45	Tim	1
Do , 04 Jan, '07 , 08:30	Do , 04 Jan, '07 , 09:00	Horst	2
Do , 04 Jan, '07 , 15:00	Do , 04 Jan, '07 , 15:30	Tatjana	1
Fr , 05 Jan, '07 , 07:00	Fr , 05 Jan, '07 , 07:30	Tatjana	1
Fr , 05 Jan, '07 , 09:00	Fr , 05 Jan, '07 , 09:30	Horst	3

Figure 5: Time Manager with supervisor access, all users are listed


TeleBots TimeSlot Manager <2>

File

Start	End	User	Robot
Di , 02 Jan, '07 , 12:00	Di , 02 Jan, '07 , 15:00	-	1
Mi , 03 Jan, '07 , 16:00	Mi , 03 Jan, '07 , 16:30	Tatjana	1
Do , 04 Jan, '07 , 08:15	Do , 04 Jan, '07 , 08:45	-	1
Do , 04 Jan, '07 , 08:30	Do , 04 Jan, '07 , 09:00	-	2
Do , 04 Jan, '07 , 15:00	Do , 04 Jan, '07 , 15:30	Tatjana	1
Fr , 05 Jan, '07 , 07:00	Fr , 05 Jan, '07 , 07:30	Tatjana	1
Fr , 05 Jan, '07 , 09:00	Fr , 05 Jan, '07 , 09:30	-	3

Figure 6: Time manager, users' view. All time slots are visible but only the name of the current user is shown.

Improved flexibility and remote control possibilities

As mentioned above, because of the system resource limitation, the controller board of the robot cannot perform demanding operations. But if we let the PC (high speed computer) perform the time consuming operations, the system is optimal. Collecting information, analysing information, and returning commands is a classic period of robot controlling (Figure 8). In our new robotic system, one period is shown as following:

The robot has just two tasks. One is to collect the environmental information; the other is to perform the commands, e.g. drive the motor. Information collection about the environment includes different sensor signals, motor information, and image data (through web camera), and then sending all data back through wireless interface to the PC. On the software structure, all information is described as different data structure and is returned from the library functions through the interface to the high-level program. The students can write their own high-level program to analyse this feedback information and then decide on the reactions of the robot. The reactions are also described as data structure and used as function parameters. After these functions being invoked, the commands are sent to the robot. The robot gets commands via wireless interface and run them.

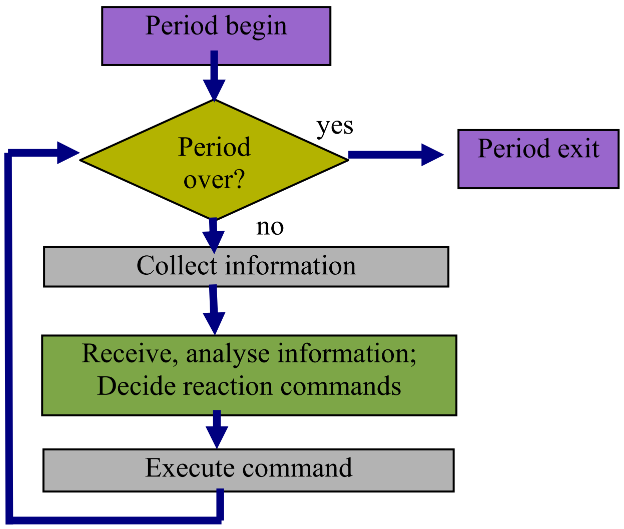


Figure 8: Classic period of robot controlling

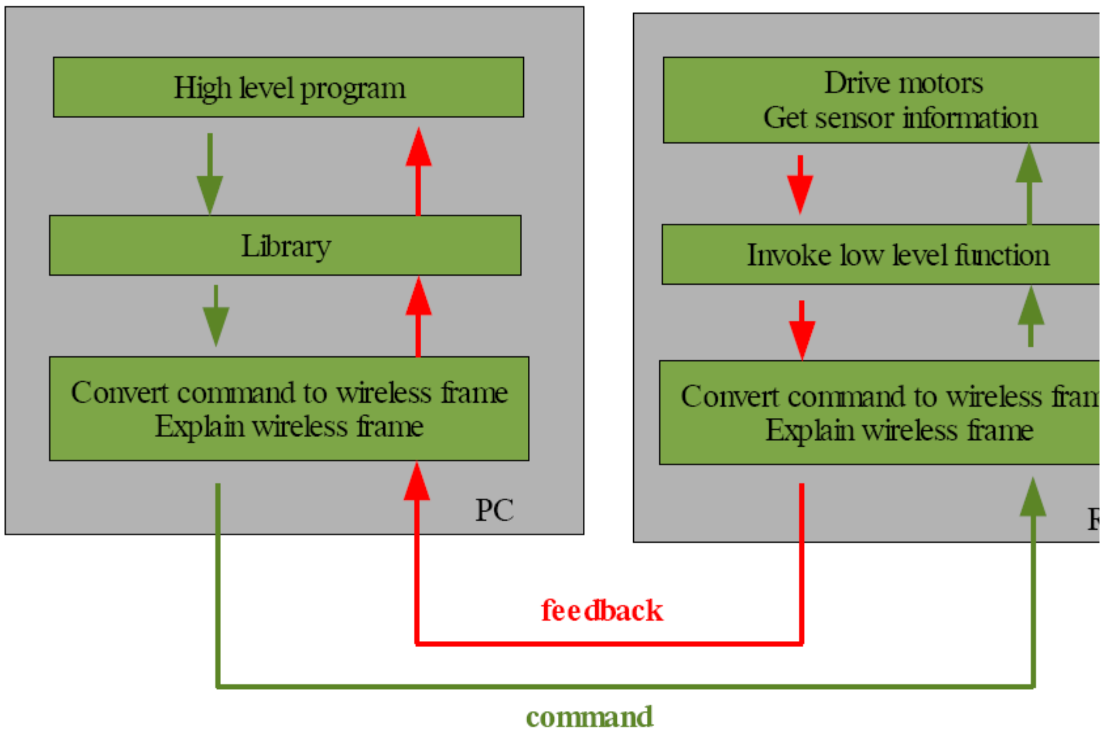


Figure 9: Running processes in one period. The red arrow is the sensor information.

The whole process of one period is described in Figure 9. The green arrow depictss the driving commands. The right side of Figure 9 shows the program which runs on the robot. We call it low-level program, while the left side depicts the high-level program which runs on the PC.
Because the PC and the robot communicate through a wireless interface, we have got rid of the remote control problem. We can also manually invoke commands to remote-control the robot.
The flexibility of the system is improved by usage of the robot proxy. The proxy can run on a PC which is connected to the controller board or on the embedded onboard system. This guarantees enough computation power for any task.

Improved simulation capabilities
The new robotic system has a simulation environment. The simulation environment is also a part of the GenRob system. It 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. They can follow the lecture based on their little experience on robots at that time. They can test whatever they want in a visual environment. Step by step, they will clarify their ideas and learn how to realize different functionalities. The discussed system hierarchy is just a general system hierarchy. More details of the improvements will be given in the next sections.

4. Detailed software hierarchy of the new robotic system

4.1. High-level program

On the PC side, we have built a normal library with the C language. Students can use the functions from this library to assemble their own program. Actually, these basic functions include two parts as shown in Figure 9. The bottom-up interface is the output data stream and input data stream. So the first function of the library is to convert the reaction commands to a special data structure, then send them out in an output data stream through the wireless interface. The second task is to receive the input data stream, analyze it and show it to the user. Meanwhile, routine functions like communication control, watchdog monitor, or checking the input on the high-level program are running.
For educational purposes, the high-level library and interface should be a universal platform. There are many students who don't have enough knowledge about C/C++, but they are good at Java. For these students we also provide a Java library in addition to the C library. Figure 10 shows the library structure from our new education telerobot system. There are two types of library, one is a basic library , the other is an extendible library.

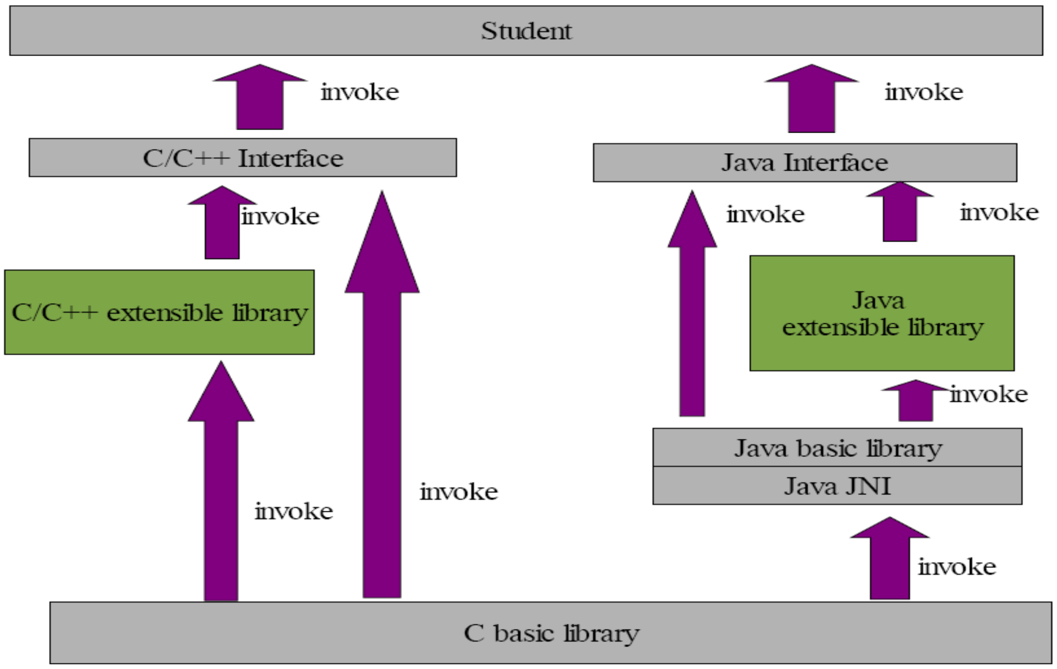
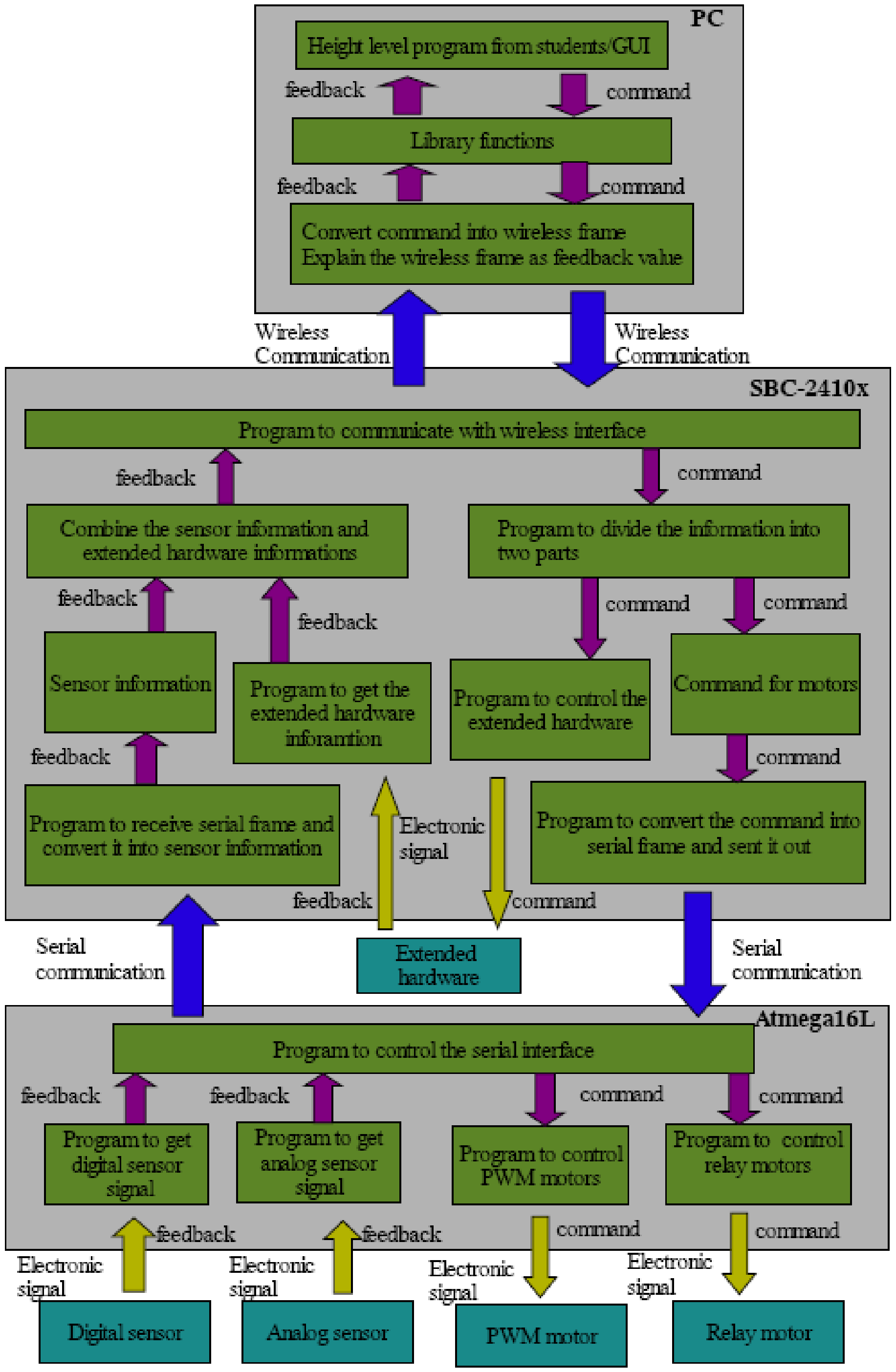


Figure 10: Library Structure

Basic library
The students cannot write their program without the basic functions. As mentioned before, it provides a security interface for students to use. This library is made by the system designer, who takes care of the whole system, and it should not be changed except by the system designer himself.

Extendible library
The extendible library is an open library which can be edited and added to by the students. Firstly students can get experience in arranging the robot work flow. For example, if the robot finds a light through optical sensors, the robot will move towards the fire source and stop in front of it. Then it will use another motor to drive the fan to kill the fire. This work flow can be arranged by students as one entirely closed function or as several subfunctions which can be added to the extendible library. After

that, they will have learned how to optimally organize work flows for a robot. Furthermore, students can optimize such functions, if they find better algorithms or better work flow models.
Secondly, existing functions can be directly invoked from the high-level program. Students do not need to write the old function. After students finish their own high-level program, they can use normal C/C++ or Java compilers to compile them, and then run the program on the PC operation system, just like the normal C/C++ or Java program as shown in Figure 11.


PC
Height level program from students/GUI
feedback
command
Library functions
feedback
command
Convert command into wireless frame
Explain the wireless frame as feedback value
Wireless Communication
Wireless Communication
SBC-2410x
Program to communicate with wireless interface
feedback
command
Combine the sensor information and extended hardware informations
Program to divide the information into two parts
feedback
feedback
command
command
Sensor information
Program to get the extended hardware inforamtion
Program to control the extended hardware
Command for motors
feedback
command
Program to receive serial frame and convert it into sensor information
Electronic signal
Program to convert the command into serial frame and sent it out
feedback
command
Serial communication
Extended hardware
Serial communication
Atmega16L
Program to control the serial interface
feedback
feedback
command
command
Program to get digital sensor signal
Program to get analog sensor signal
Program to control PWM motors
Program to control relay motors
feedback
feedback
command
command
Electronic signal
Electronic signal
Electronic signal
Electronic signal
Digital sensor
Analog sensor
PWM motor
Relay motor
Figure 11: Program hierarchy.

4.2. Low-level program

The low-level program runs on the robot. It also contains a library and the functions can be invoked by the high-level program. The low-level program hierarchy is more complex than the high-level program. Firstly, the program on the embedded system board (SBC-2410x) communicates with the PC via wireless interface. Using a serial port interface, the embedded system communicates with the low-level micro controller at the same time, as shown in Figure 12.

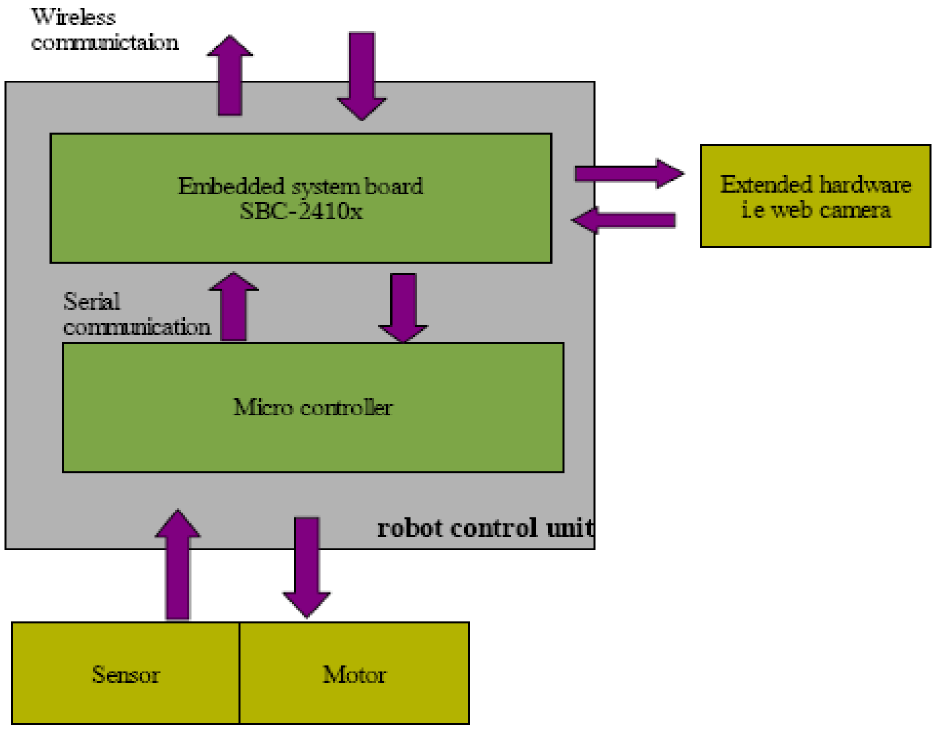


Figure 12: The communications structure.

Two programs are running: the level-1-program on the embedded system board and the level-2-program on the micro controller.

As this embedded system board can run a Linux operation system, the level-1-program consists of the following subprograms:

1. Communication with the wireless interface. There must be a routine program such as data stream control, converting the information from the PC to the wireless transmit data stream, or converting the data stream to useful information for the lower program.
2. A function to divide the received information into two parts. One is the program for the embedded system itself and the other is for the micro controller board.
3. A program to combine the sensor information coming from the lower micro controller and the useful information from the extended hardware such as a web camera, and to convert data to a special data structure , which will be sent to PC.
4. A program to control the hardware of the embedded system board, such as a web camera, to get the picture, or the sound from a microphone etc.
5. A program to convert the motor commands into a serial transmit frame and send them out through a serial port. This program has to include a routine such as time-out control, sum-check control etc. This part guarantees the communication between the embedded system board and the micro controller board.
6. A program to explain the sensor information, which is received from the micro controller

board, as well as the program above.

The level-2-program consists of the following subprograms:

1. Control of the motors.
   There are four motors on the robot. Two of them driven by PWM signals run with different velocities and directions. The other two are relay motors.
2. Collecting the sensor data.
   Sensor parts include digital and analogue sensors. The programs collecting the analogue sensor information and the digital sensor information have different mechanisms. In order to get a stable sensor signal, filters are used for every sensor.
3. Control of the serial port.
   After accepting the serial data frame, this program will explain this frame and then invoke the corresponding functions to drive the motors and collect the sensor signals. Then the program combines the sensor information, timer, motor information or alarm status etc, and converts it into a serial frame, which will be sent to level 1. In order to guarantee serial communication, some routines such as time out, checking the sum etc are invoked in this program as well. The lower part in Figure 7 shows the level-2-program hierarchy

5. Progress and Difficulties

5.1. Progress

To date, the framework for the software system has been developed. 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.
In total, functions which have been achieved include:

1. ***All functions on level 2***
2. ***All functions on level 1 with an incomplete GUI***
3. ***All educational tasks such as following a line, finding a target, as mentioned in our proposal***
4. *The better GUI is on the way.*

On the other hand, some difficulties we mentioned in the former report have been solved. We improved the usage-stability of our own new motor and sensor bricks. Through a great number of tests, currently all new bricks are 100% stable for use.
According to the requirements of this project, the tele-access capability has been met using RS232. The CCD input is used for image collecting; the onboard high-level controller sends the images to another GUI by a wireless communication interface.

5.2. Difficulties and solutions

At the moment, the difficulties include two parts:
1. Image transferring communication
Even though our hardware system can realize all these functions, at the moment, we still wonder if the communication velocity is fast enough for image transferring.

2. GUI part
To date, the user database server including a user-administration GUI and user verification interface have been implemented and tested. The GUI is the last part, consisting of a commands input area, a sensorial information area, a status information area, a CCD camera and general information. The students can start a task automatically or send commands directly to the hardware onboard. The images from CCD cameras are used for monitoring at the first step. In future, we can analyze them for further information.
All of the functions above are not too difficult to achieve. The only bottleneck is the simulation part. It is very important for the students to attend this part of the lecture due to their lacking experience of robots at the moment. They can test whatever they want in a simulation environment. Step by step, they will clarify their ideas and learn how to realize special functionalities. However, for an educational project, the goal is not to perfectly simulate reality, but to build and program a real robot. In any case, building a simulation scenario and a 3D simulated robot on the GUI in an educational project is difficult given the current state of the art of technology.

6. Future work
Based on the current achievements, we will focus on the difficulties, especially on implementing the GUI part later.
Firstly, the communication interface part will receive more attention according to the image transfer. Furthermore we will keep the possibility to analyze the image data open for the future.
The software will be extended. The simulation part of the software is very important for a tutoring system. It will be a focus in this project from now.
All future work will deal with system integration.

7. Conclusion
With the support of the ELCH, a practical course "Mobile Robots" with simulation functions and Tele-robot-access functions intended for Bachelor/Master studies is currently being 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