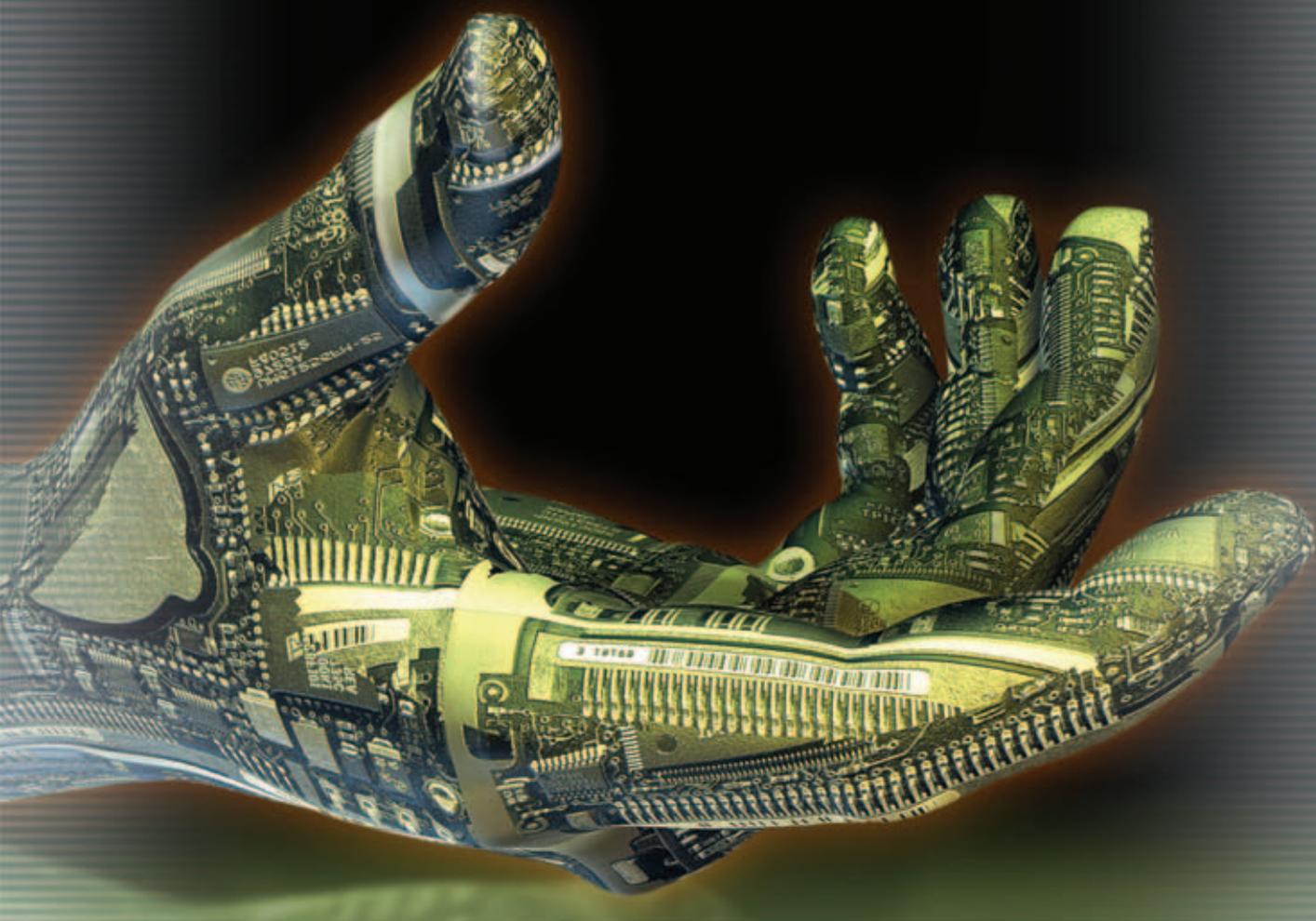


AT&P JOURNAL **2** *plus* 2010



Robotics in Education

*reviewed slovak professional
magazine for scientific
and engineering issues*

Robotika vo vzdelávaní

**recenzované periodikum
vedeckých a inžinierskych
publikácií**

Robotika vo vzdelávaní

Robotics in Education

Odborný garant

Ing. Richard Balogh

Slovenská technická univerzita v Bratislave
Fakulta elektrotechniky a informatiky
Ústav riadenia a priemyselnej informatiky
Ilkovičova 3, 812 19 Bratislava, Slovensko
e-mail: richard.balogh@stuba.sk

Technical guarantee

Ing. Richard Balogh

Slovak University of Technology in Bratislava
Faculty of Electrical Engineering and Information Technology
Institute of Control and Industrial Informatics
Ilkovičova 3, 812 19 Bratislava, Slovak Republic
e-mail: richard.balogh@stuba.sk

Vydavateľ Publisher

HMH s.r.o.

Tavarikova osada 39
841 02 Bratislava 42
IČO: 31356273

Spoluzakladateľ Co-founder

Katedra ASR, EF STU

Katedra automatizácie a regulácie, EF STU

Katedra automatizácie, ChtF STU

PPA CONTROLL, a.s.

Partnerské organizácie AT&P journalu AT&P journal partnering organizations



www.atpjournals.sk

AT&P journal PLUS2 2010

prof. Ing. Alexík Mikuláš, PhD., FRI ŽU, Žilina
doc. Ing. Dvoran Ján, CSc., FCHPT STU, Bratislava
prof. Dr. Ing. Fikar Miroslav, FCHPT STU, Bratislava
doc. Ing. Hantuch Igor, PhD., KAR FEI STU, Bratislava
doc. Ing. Hrádcký Ladislav, PhD., SJF TU, Košice
prof. Ing. Hulkó Gabriel, DrSc., SJF STU, Bratislava
prof. Ing. Jurišica Ladislav, PhD., FEI STU, Bratislava
doc. Ing. Kachaňák Anton, CSc., SJF STU, Bratislava
prof. Ing. Krokavec Dušan, CSc., KKUI FEI TU Košice
prof. Ing. Madarász Ladislav, PhD., FEI TU, Košice
prof. Ing. Malindžák Dušan, CSc., BERG TU, Košice
prof. Ing. Mészáros Alojz, CSc., FCHPT STU, Bratislava
prof. Ing. Mikleš Ján, DrSc., FCHPT STU, Bratislava
prof. Dr. Ing. Moravčík Oliver, MtF STU, Trnava
prof. Ing. Murgaš Ján, PhD., FEI STU, Bratislava
prof. Ing. Rástočný Karol, PhD., KRIS ŽU, Žilina
prof. Ing. Schreiber Peter, CSc., MtF STU, Trnava
prof. Ing. Skyva Ladislav, DrSc., FRI ŽU, Žilina
prof. Ing. Smieško Viktor, PhD., FEI STU, Bratislava
doc. Ing. Šturcel Ján, PhD., FEI STU, Bratislava
prof. Ing. Taufer Ivan, DrSc., Univerzita Pardubice
prof. Ing. Veselý Vojtech, DrSc., FEI STU, Bratislava
prof. Ing. Žalman Milan, PhD., FEI STU, Bratislava

Ing. Bartošovič Štefan,
generálny riaditeľ – president
ProCS, s.r.o.

Ing. Bodo Vladimír, CSc.,
riaditeľ – managing director
AXESS, spol. s r.o.

Ing. Csölle Attila,
riaditeľ – managing director
Emerson Process Management, s.r.o.

Ing. Horváth Tomáš,
riaditeľ – managing director
HMH, s.r.o.

Ing. Hrica Marián,
riaditeľ divízie A & D – head of A&D division
Siemens, s.r.o.

Jiří Kroupa,
DEHN + SÖHNE

Ing. Murančan Ladislav,
PPA Controll a.s., Bratislava

Ing. Petergáč Štefan,
predseda predstavenstva - chairman of board director
Datalan, a.s.

Ing. Pilňan Branislav
sales leader HPS
HONEYWELL s.r.o.

Ing. Tóth Andrej,
generálny riaditeľ - president
ABB, s.r.o.

AT&P journal

Evidenčné číslo: EV 3242/09
Košická 37, 821 09 Bratislava 2
tel.: 02/5026 1752 - 55
fax: 02/5026 1757
e-mail: info@atpjournalsk
www.atpjournalsk

Ing. Anton Géer
šéfredaktor - editor in chief
sefredaktor@atpjournalsk

Ing. Ildikó Csölleová
vedúca redakcie – editorial office manager
podklady@atpjournalsk

Bc. Zuzana Bakošová
marketingová manažérka – marketing manager
marketing@atpjournalsk

Ing. Branislav Bložon
odborný redaktor – editor
redaktor@atpjournalsk

Ing. Martin Karbovanec
odborný redaktor – editor
karbovanec@atpjournalsk

Peter Kanda
technický redaktor – DTP
dtp@atpjournalsk

Mgr. Bronislava Chocholová
jazyková redaktorka – text corrector

Obsah

Acrob – výučbová robotická platforma (len anglická verzia)	6
Richard Balogh	
Výučba robotiky na postgraduálnej úrovni:	
prednášanie pre študentov riadneho a dištančného štúdia (len anglická verzia)	10
Jenny Carter, Simon Coupland	
Autonómne vozíky ako aplikácia vhodná pre študentov	
Priemyselného inžinierstva a Manažmentu (len anglická verzia)	16
André Dias, Nuno Dias, Daniela Campos, Hugo Ferreira	
Robotour – exteriérová súťaž mobilných robotov (len anglická verzia)	21
Jiří Iša, Martin Dlouhý	
EDURO – mobilná robotická platforma pre výučbu (len anglická verzia)	26
Martin Dlouhý, Jan Roubíček, Tomáš Roubíček	
SyRoTek – Robotický systém pre výučbu (len anglická verzia)	31
Jan Faigl, Jan Chudoba, Karel Košnar, Miroslav Kulich, Martin Saska, Libor Přeučil	
Výučba mobilnej robotiky na FEI ČVUT v Prahe (len anglická verzia)	37
Jan Faigl, Tomáš Krajník, Karel Košnar, Hana Szücsová, Jan Chudoba, Vladimír Grimmer, Libor Přeučil	
Riadenie mobilného robota cez web (len anglická verzia)	43
Jaroslav Hanzel	
Predmet „Robotika“ na FEI ČVUT v Prahe	
– využívanie robotov LEGO pri výučbe základov riadenia (len anglická verzia)	46
Martin Hlinovský, Tomáš Polcar	
Lekcia s LEGO Mindstorms: od začiatočníkov po výučbu robotiky (len anglická verzia)	51
Martina Kabátová, Janka Pekárová	
Monokulárny navigačný systém pre súťaž Robotour (len anglická verzia)	57
Tomáš Krajník, Jan Faigl, Vojtěch Vonásek, Hana Szücsová, Ondřej Fišer, Libor Přeučil	
Vyučovanie humanoidnej robotiky na kurze v rámci počítačového inžinierstva (len anglická verzia)	63
Martin Mellado	
RoboTour ako naučené správanie založené na umelých neurónových sieťach (len anglická verzia)	69
Miroslav Nadhajský, Pavel Petrovič	
Využívanie platformy Lego Mindstorms pri výučbe priemyselnej automatizácie (len anglická verzia)	74
Carolyn Oates, Alois Zotl	
Otvorená platforma pre výučbu a projektové práce na pre- a postgraduálnom stupni (len anglická verzia)	79
Benjamin N. Passow, James Wheeler, Simon Coupland, Mario A. Gonara	
Prístup Robotika.SK k vzdelávacej robotike od základnej školy po univerzitu (len anglická verzia)	85
Pavel Petrovič, Richard Balogh, Andrej Lúčný	
Návrh a validácia robotického systému pre interaktívnu výučbu geometrie (len anglická verzia)	91
Lorenzo Riano, Martin McGinnity	
European Land Robot Trial (ELROB) – smerom k realistickým skúškam pre exteriérových robotov (len anglická verzia) .	97
Frank E. Schneider, Dennis Wildermuth, Bernd Brüggemann, Timo Röhling	
O vzdelávacom prístupe k behaviorálnemu učeniu pre robotiku (len anglická verzia)	103
Michel Tokic, Arne Usadel, Joachim Fessler, Wolfgang Ertel	
Niektoré didaktické problémy výučby robotiky (len anglická verzia)	109
Anton Vitko, Ladislav Jurišica, Andrej Babinec, František Duchoň, Marian Klúčik	
Stavba robotov ako motivačný nástroj pre študentov inžinierskeho zamerania (len anglická verzia)	113
Francis Wyffels, Michiel Hermans, Benjamin Schrauwen	
Arduino Etoys: programovacia platforma pre Physical Etoys (len anglická verzia)	117
Gonzalo Esteban Zabala, Ricardo Morán, Sebastián Blanco	
Archív	122

Contents

Acrob – an Educational Robotic Platform	6
Richard Balogh	
Teaching Robotics at the Postgraduate Level: Delivering for On Site and Distance Learning Student	10
Jenny Carter, Simon Coupland	
Autonomous Guided Vehicles Applied to Industrial Engineering and Management Studies	16
André Dias, Nuno Dias, Daniela Campos, Hugo Ferreira	
Robotour – robotika.cz outdoor delivery challenge	21
Jiří Iša, Martin Dlouhý	
EDURO – Mobile Robotic Platform for Education	26
Martin Dlouhý, Jan Roubíček, Tomáš Roubíček	
SyRoTek – A Robotic System for Education	31
Jan Faigl, Jan Chudoba, Karel Košnar, Miroslav Kulich, Martin Saska, Libor Přeučil	
Mobile Robotics Education at FEE CTU in Prague	37
Jan Faigl, Tomáš Krajník, Karel Košnar, Hana Szücsová, Jan Chudoba, Vladimír Grimmer, Libor Přeučil	
Web based remote mobile robot control	43
Jaroslav Hanzel	
Subject „Robots“ at the CTU FEE in Prague – using LEGO robots to teach the fundamental of feedback control	46
Martin Hlinovský, Tomáš Polcar	
Lessons learnt with LEGO Mindstorms: from beginner to teaching robotics	51
Martina Kabátová, Janka Pekárová	
A Monocular Navigation System for RoboTour Competition	57
Tomáš Krajník, Jan Faigl, Vojtěch Vonásek, Hana Szücsová, Ondřej Fišer, Libor Přeučil	
Teaching about Humanoids in a Robotic Course on Computer Engineering Studies	63
Martin Mellado	
RoboTour solution as a learned behavior based on Artificial Neural Networks	69
Miroslav Nadhajský, Pavel Petrovič	
Utilizing Lego Mindstorms as a Teaching Platform for Industrial Automation	74
Carolyn Oates, Alois Zötl	
An Open Platform for Teaching and Project Based Work at the Undergraduate and Postgraduate Level	79
Benjamin N. Passow, James Wheeler, Simon Coupland, Mario A. Gonara	
Robotika.SK Approach to Educational Robotics from Elementary Schools to Universities	85
Pavel Petrovič, Richard Balogh, Andrej Lúčný	
Design and Validation of a Robotic System to Interactively Teach Geometry	91
Lorenzo Riano, Martin McGinnity	
European Land Robot Trial (ELROB) – Towards a Realistic Benchmark for Outdoor Robotics	97
Frank E. Schneider, Dennis Wildermuth, Bernd Brüggemann, Timo Röhling	
On an educational approach to behavior learning for robots	103
Michel Tokic, Arne Usadel, Joachim Fessler, Wolfgang Ertel	
Some didactic problems of teaching robotics	109
Anton Vitko, Ladislav Jurišica, Andrej Babinec, František Duchoň, Marian Klúčik	
Building robots as a tool to motivate students into an engineering education	113
Francis Wyffels, Michiel Hermans, Benjamin Schrauwen	
Arduino Etoys: a programming platform for Arduino on Physical Etoys	117
Gonzalo Esteban Zabala, Ricardo Morán, Sebastián Blanco	
Archive	122

Acrob – an Educational Robotic Platform

Richard Balogh

Abstract

In our paper we describe the design of a new controller board for the mobile robot based on the Parallax Boe-Bot chassis. Disadvantages of the original Basic Stamp processor disappeared, more complicated tasks can be solved. As the board is compatible with the Arduino platform, also the open source development environment can be used. We describe the requirements, design process and technical parameters. Also some illustration examples are shown.

Keywords: controller, mobile robot, Arduino, robotic platform

Introduction

In our university we used for many years the commercially available mobile robots Boe-Bot¹ by the Parallax, Inc. for education [1]. They were used in some laboratory exercises for students of the Mobile robotics lectures, some additional lectures for students of Embedded systems, or Automotive control systems. We used this platform also for summer courses, student projects and for public presentations.

Our main problem with the Boe-Bot robot was with its controller unit. Although the Basic Stamp II with its programming capabilities is very reliable and useful for start up, our advanced students at the university were critical to use the Basic as a programming language for robots. They lack function definitions, program hierarchy, interrupts, parallel tasks, and direct access to the peripherals like timers, counters etc. Our experiences with the 8-bit RISC AVR processors by the Atmel and growing popularity of the Arduino platform leads us to the design of the completely new controller board for the robot.



Figure 1: The Boe-Bot mobile robot with the new controller.

¹ <http://www.parallax.com/Store/Robots/AllRobots/tabid/128/ProductID/302/List/1/Default.aspx>

The main goal was to achieve as much compatibility as possible. Not only the dimensions (which are essential to replace the board with the original one), but also the overall concept, connectors placement etc. were sustained. Now, we can use the robot with almost all original extensions of the Boe-Bot robot.

1. Components of the System

The Boe-Bot mobile robot [2] is a commercially available robotics kit by the Parallax, Inc. company. It consists of two geared motors mounted on an aluminum chassis, batteries and control electronics. On the motors are mounted two plastic wheels. The rear wheel is made of a drilled polyethylene ball. Mounting holes and slots may be used to add custom robotic equipment.

The robot is controlled by the Parallax's popular microcontroller Basic Stamp II and the Board of Education. It is a simple board containing a processor, power supply circuits, interfaces, connectors and a small experimental solderless breadboard. The Basic Stamp II processor can be programmed with the PBASIC language - simple, but powerful clone of the Basic language with the support of many specific peripheral devices [2]. Pros and cons of this platform were evaluated in details in [3].

Arduino is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. It's intended for artists, designers, hobbyists, and anyone interested in creating interactive objects or environments [4]. The microcontroller on the board is programmed using the Arduino programming language (based on Wiring) and the Arduino development environment (based on Processing) [5]. The hardware reference designs (CAD files) are available under an open-source license, you are free to adapt them to your needs. This is also the our case, we designed the completely new board retaining the compatibility with the platform.

2. Hardware

The new board is as much compatible as possible with the original Board of Education by Parallax Inc. [6]. The main differences are two: different processor and TTL serial interface without the converters.

For power supply we can use the battery box (with four primary or rechargeable 1,5V AA size batteries) or the wall adapter. The power switch has three positions. Except the standard on/off positions there is a special “development” position when motors are disconnected so the robot is not moving on the desktop during the debugging.

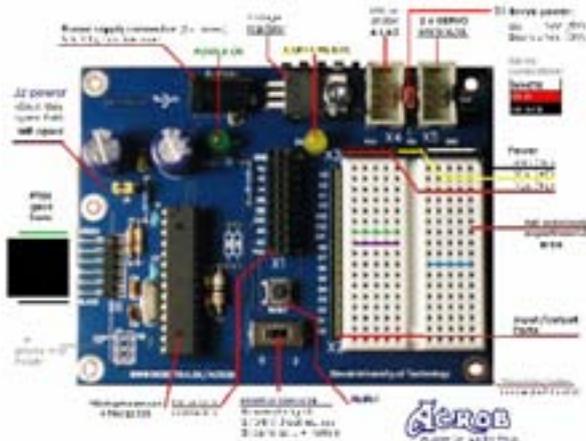


Figure 2: Description of the board.

On-board voltage stabilizator provides 5V for microcontroller and its peripherals. As the main processor the Atmel Atmega328P with pre-burned bootloader is used. It provides the user with 32kB of the program memory, 2kB of data RAM space and 1kB of the EEPROM space. The main area of the board is occupied with the solderless experimental breadboard which enables to connect different additional components. On its left side most of the I/O pins are available, on its top there is a power supply connector. The board also contains connectors for servomotors and two additional sensors with digital or analogue outputs. The dual line connector in the center of the board enables to connect standard Parallax's extension boards like the compass or the LCD modules.

ID	Purpose	Type
X1	Expansion connector for compatible with the Parallax AppMods.	2x10
X2	Access to the I/O pins	1x16
X3	Power supply for breadboard (Vcc, Vin, GND)	1x13
X4	Sensors (I2C bus and/or Digital/Analog Input)	2x3
X5	Servomotors (2xPWM)	2x3
X6	Serial / Programming interface (bootload)	1x6
X7	Power supply jack	3,5 mm
X8	In System Programming interface – ISP (MISO, MOSI, RESET,...)	2x3

Tab 1: Connector Description Table

See also the description of the comprehensive set of connectors for peripherals contained on the board in the Tab.1.

Programming and communication capabilities were increased comparing to the original Boe-Bot robot. We decide to have only the serial communication interface with TTL levels without any other converters on the board., so different converters can be used. We can use the standard FTDI Chips² USB cable or the SparkFun's FTDI Basic³ module for programming using the internal bootloader. Also we developed RS-232 level converter module to enable operation also with the standard serial interface (see the Fig. 3).

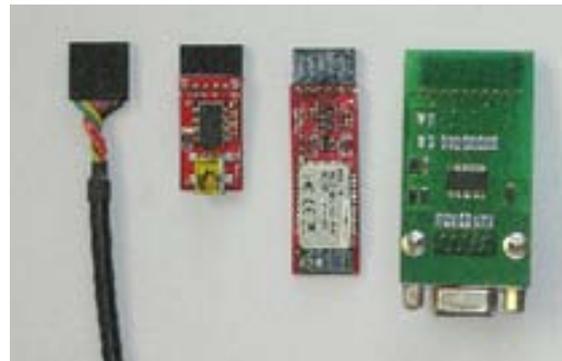


Figure 3: Programming and communication using the (from left to right) a) FTDI USB Cable, b) SparkFun FTDI Basic module c) SparkFun Blue Mate Bluetooth module d) custom made RS-232 module.

After the program loading, the interface is free for any user serial communication operations. This enables to connect e. g. SparkFun's BlueMate⁴ communication module to communicate with the computer or between the robots using the Bluetooth interface. On the board there is also connector for ISP programmer, so one can use any standard Atmel ISP programmer to burn the program into the processor. Together with the AVRStudio one can even debug, step and watch the programs written in assembler or avr-gcc languages.

The board can be used also as a standalone with the only connection using the FTDI USB cable. In such case user can power the board from the USB interface so no additional equipment is necessary. Such a configuration can be used for introduction to the embedded systems programming, explaining basics of the digital and analog inputs, outputs or built-in peripherals like the timers, counters, PWM and A/D converter. The schematics and printed circuit board were designed using the open source gEDA suite⁵. We used thru-hole components to enable student to build their own boards from the kits.

3. Software

For programming, the standard Arduino IDE can be used. Other methods include the assembly or avr-gcc languages integrated within the Atmel AVR Studio or using a set of command line utilities. We tested the environment on MS Windows XP operating system, but the Arduino IDE

2 <http://www.ftdichip.com/Products/Cables/USBTTLSerial.htm>

3 http://www.sparkfun.com/commerce/product_info.php?products_id=9115

4 http://www.sparkfun.com/commerce/product_info.php?products_id=9358

5 <http://www.gpleda.org>

should work also on the Linux and MAC OS systems. There is only one problematic point we found - during the installation process you need administrative rights to install USB drivers properly. This problem diminished in Windows 7 where drivers seemed to be already contained.

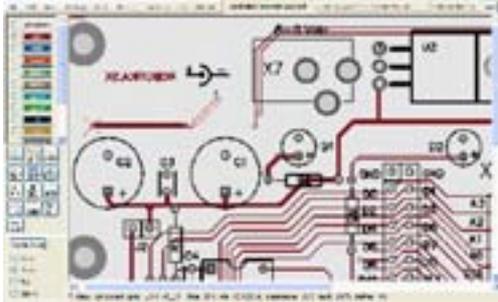


Figure 4: Design in the pcb program from the gEDA suite.

We prepared a set of basic programs to show an access to the peripherals. We start with the basic digital I/O (LED and switch), then move to the analog world – basic robot movements and analog sensor measurements. As the first analog sensor we find very useful Sharp distance sensors which are easy to connect and offer reliable results. Also their non-linear characteristics is challenging.

As the very first program we used the standard “Hello, World!” problem.

```
void setup()
{
  Serial.begin(9600);
}

void loop()
{
  Serial.println("Hello, World!");
}
```

After the compilation and burning the program into the processor using the bootloader, the user can see the result using the internal built-in terminal window (see Fig. 5). Sometimes the communication speeds didn't correspond to the real and characters were displayed incorrectly until it was changed.

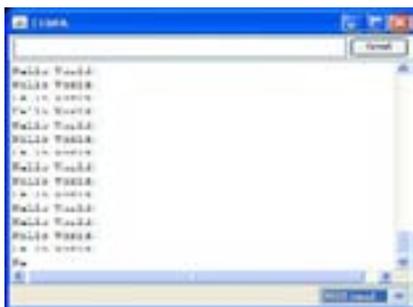


Figure 5. Hello World program in terminal window.

From the listing above it is clear that the programming is very straightforward and at the beginning no processor specific knowledge is required. Only important thing is to split the program to the two basic parts – `setup` (which is performed only once) and `loop` (which is then performed indefinitely – or better say, until not switched off or reprogrammed). As the Arduino language is built over the standard avr-gcc compiler, we can still use all its features and combine also standard approach e. g. direct access to the all processor registers:

```
TCCR0A |= _BV(WGM01) | _BV(WGM00);
OCR0A = 127;
OCR0B = 255;
```

Of course, the libraries can hide the internals from the user so no special knowledge is required. An example of library for servos to show basic robot movements follows:

```
#include <Servo.h> // this program uses the Servo library

Servo LeftServo; // create servo object to control both servos
Servo RightServo; // a maximum of eight servos can be created

#define FAST 50 // try to change these values during the test
#define SLOW 5

void setup()
{
  LeftServo.attach(9); // attach servo on pin 9 to the servo object
  RightServo.attach(10); // attach servo on pin 10 to servo object
}

void loop()
{
  // FAST FORWARD
  LeftServo.write(90 + FAST); // value 90 is in middle, i.e. stop
  RightServo.write(90 - FAST); // mirrored position
  delay(1500); // go fast forward for 1,5 s

  // SLOW FORWARD
  LeftServo.write(90 + SLOW); // test varying speed of movement
  RightServo.write(90 - SLOW);
  delay(1500);

  LeftServo.write(90 - FAST); // FAST BACKWARD
  RightServo.write(90 + FAST);
  delay(1500);

  LeftServo.write(90 + FAST); // ROTATE (PIVOT) RIGHT
  RightServo.write(90 + FAST);
  delay(1500);

  LeftServo.write(90 - FAST); // ROTATE (PIVOT) LEFT
  RightServo.write(90 - FAST);
  delay(1500);

  LeftServo.write(90); // STOP both motors
  RightServo.write(90);

  for(;;); // stop the program operation here
} /* End of Loop */
```

For a comparison – similar program written in an original Basic Stamp II language may look like this

```
{ $STAMP BS2 }
{ $PBASIC 2.5 }

counter VAR Word
pulseLeft VAR Word
pulseRight VAR Word
pulseCount VAR Byte

' Forward
pulseLeft = 850: pulseRight = 650: pulseCount = 64: GOSUB
Navigate

' Left turn
pulseLeft = 650: pulseRight = 650: pulseCount = 24: GOSUB
Navigate

' Right turn
pulseLeft = 850: pulseRight = 850: pulseCount = 24: GOSUB
Navigate

' Backward
pulseLeft = 650: pulseRight = 850: pulseCount = 64: GOSUB
Navigate

END

Navigate:

FOR counter = 1 TO pulseCount
PULSOUT 13, pulseLeft
PULSOUT 12, pulseRight
PAUSE 20
NEXT
PAUSE 200

RETURN
```

4. Support

We created the supporting page with all the necessary documentation at our robotics server⁶. Also we started with creation of the comprehensive manual with example programs and connection diagrams. One of the big advantages of the open source system is the large community of users adding their experiences to the whole system. As the board is Arduino compatible, we can immediately start to use an existing repository of examples, documentation etc. Let's say one want to connect ultrasonic detector SRF-08 to the robot. Very soon it finds not only few examples but also a connection diagram⁷ and even the whole library⁸ to use with this sensor. Just type the keywords 'SRF08' and 'Arduino' to your favourite search engine.

5. Evaluation

The new robot, called Acrob (**A**rduno **C**ompatible **R**obot) was tested and evaluated at some different events. We prepared the robotic introductory lecture for students of the Automotive branch of study. The main goal was to give them an idea of the mobile robots and its programming. During the two lectures the students were able to program basic movements and reactive behavior of the robots. For students of the Mobile robotics course we prepared similar lecture and the platform was also used as an example of differential driven platform. Also the infrared sensor distance detection was explained. Then we use the robots in a joined Austrian-Slovak lecture for secondary school students. During the lecture they were able to program the movements, connect and evaluate line sensors and distance sensor so they finished with the simple line-following robot with an obstacle avoidance. During the time of writing this paper were the robots tested intensively on the Summer School of Robotics and they were successful too.

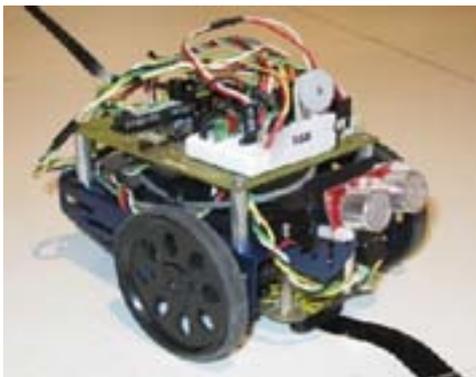


Figure 6: Prototype of the robot with line following sensor and ultrasonic obstacle detector at the Robotchallenge contest.

The overall concept was successfully tested also at the Robotchallenge Wien 2010 contest where the robot

⁶ <http://www.robotika.sk/acrob>

⁷ http://www.robot-electronics.co.uk/htm/arduino_examples.htm#SRF08%20Ultrasonic%20Ranger

⁸ <http://www.arduino.cc/playground/Main/SonarSrf08>

successfully (even very slowly) passed the line following category (see Fig. 6).

6. Conclusions

Presented robotic platform offers many capabilities. The main problem of the previously used Parallax's Boe-Bot platform – programming in Basic was successfully solved and programming is now possible both in assembly and C++ languages. The concept was proven on some robotic lessons for the university students, for students of secondary school and also at the summer school and robotic contests.

Acknowledgment

This project is a part of the CENTROBOT project funded by the EFRE program Creating the future (CBC SK-AT 2007-13). Author would appreciate also the help and support of the Parallax company.

References

- [1] BALOGH, R.: *Basic Activities with the Boe-Bot Mobile Robot*. In DidInfo 2008. 14th International Conference. FPV UMB, Banská Bystrica, Slovakia.
- [2] LINDSAY, A.: *Robotics with the Boe-Bot*. Student guide. Version 2.2., Parallax, Inc. Rocklin, California, 2004. ISBN 1-928982-03-4. Available on-line: <http://www.parallax.com>
- [3] HOFMANN, A. et al.: *Robotic Education Platform*. Literature Research and Evaluation. Technical Report, part WP2 of the Centrobot project. Wien, 2009. Available on-line: <http://www.centrobot.eu/en/aktivitaeten/robotic-education-platform>
- [4] ARDUINO Home Page. <http://www.arduino.cc/>
- [5] BANZLI, M.: *Getting Started with Arduino*. O'Reilly, 2009.
- [6] Board Of Education. Development / Educational Platform for the BASIC Stamp and Javelin Stamp Microcontrollers. Parallax, Inc. 2007. Available on-line: <http://www.parallax.com/Portals/0/Downloads/docs/prodboards/28150-BOE-Serial-v2.0.pdf>

Richard Balogh

Slovak University of Technology in Bratislava
Faculty of Electrical Engineering and Inf. Technology
Ilkovičova 3
811 09 Bratislava
E-mail: richard.balogh@stuba.sk

Teaching Robotics at the Postgraduate Level: Delivering for On Site and Distance Learning

Jenny Carter and Simon Coupland

Abstract

The MSc Intelligent Systems (IS) and the MSc Intelligent Systems and Robotics (ISR) programmes at De Montfort University are Masters level courses that are delivered both on-site and by distance learning. The courses have been running successfully on-site for 6 years and are now in the third year with a distance learning mode. Delivering material at a distance, especially where there is technical and practical content, always presents a challenge but the need to deliver a robotics module increased the challenges we faced significantly. There are two robotics modules though the second one is only available to those on MSc ISR. We have chosen to make the first robotics module, Mobile Robots, the focus of this paper because it was the first that had to be delivered and it is delivered to students on both programmes. This paper describes the rationale, delivery and assessment of the Mobile Robots module to students on the MSc IS/ISR with a specific focus on those students that are studying in distance learning mode. We believe it serves as a model for others attempting to teach robotics at distance.

Keywords: robotics, distance learning

Introduction

The courses are delivered mainly by the members of the Centre for Computational Intelligence (CCI) at De Montfort University. Their development enabled us to capitalise on the research taking place within the CCI and therefore on the strengths of the staff delivering the modules. It is generally preferred that staff members teach their special interests thus enabling research to support teaching and vice versa. There are significant benefits when a course is delivered by a team of people with a strong interest and research focus in the same areas.

Initial decisions were necessary to determine the content of the courses. There are a large number of topics that could be considered but the areas of fuzzy logic, neural networks, evolutionary computing, knowledge based systems and logic programming provide an array of tools and paradigms that encompasses much of what is considered to be computational intelligence and thus form the basis of the content. The ability to get mobile robots to react intelligently to their environment is a highly developed research field and it is one that is being tackled within the CCI; it also provides an ideal application area for applying the previously mentioned techniques and therefore mobile robots modules were included.

MSc ISR includes two mobile robots modules whilst MSc IS replaces one of these with a data mining module as an alternative application area for those less interested in pursuing mobile robotics work. In addition to the modules mentioned so far, a research methods module is delivered in semester 1 to ensure that students are equipped with the necessary skills to carry out literature searches, write project proposals and so on; and the Applied Computational Intelligence module enables students to pursue an appropriate area of their own interest in greater depth.

The remainder of this paper is structured as follows: Section 1 considers the background of the courses, their development and structure;

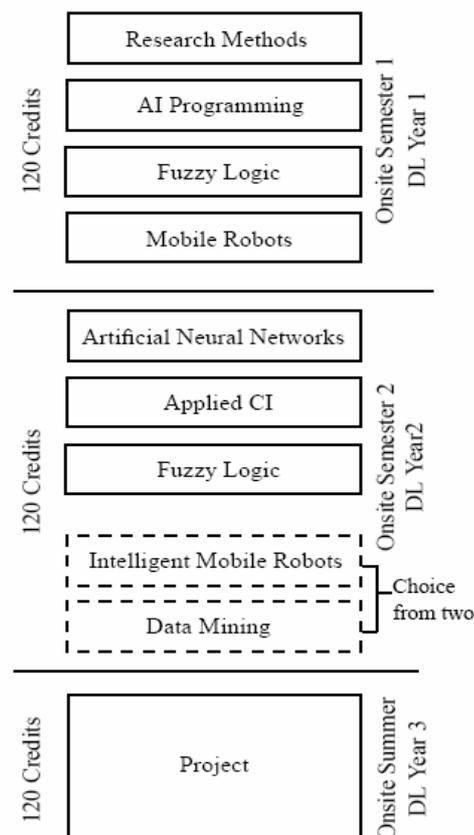


Fig.1 Course structure for MSc IS and MSc ISR

Section 2 the approaches to learning that we have adopted for the course, Section 3 gives a detailed account of the delivery of the Mobile Robots module and finally Section 4 draws conclusions from this work.

1. Background

The courses are delivered mainly by the members of the Centre for Computational Intelligence (CCI) at De Montfort University. Their development enabled us to capitalise on the research taking place within the CCI and therefore on the strengths of the staff delivering the modules. It is generally preferred that staff members teach their special interests thus enabling research to support teaching and vice versa. There are significant benefits when a course is delivered by a team of people with a strong interest and research focus in the same areas.

Initial decisions were necessary to determine the content of the courses. There are a large number of topics that could be considered but the areas of fuzzy logic, neural networks, evolutionary computing, knowledge based systems and logic programming provide an array of tools and paradigms that encompasses much of what is considered to be computational intelligence and thus form the basis of the content. The ability to get mobile robots to react intelligently to their environment is a highly developed research field and it is one that is being tackled within the CCI; it also provides an ideal application area for applying the previously mentioned techniques and therefore mobile robots modules were included.

MSc ISR includes two mobile robots modules whilst MSc IS replaces one of these with a data mining module as an alternative application area for those less interested in pursuing mobile robotics work. In addition to the modules mentioned so far, a research methods module is delivered in semester 1 to ensure that students are equipped with the necessary skills to carry out literature searches, write project proposals and so on; and the Applied Computational Intelligence module enables students to pursue an appropriate area of their own interest in greater depth.

2. Approaches to Learning

We aim to adopt an approach to our delivery of the courses that embraces modern technology in such a way that the students have appropriate learning experiences whether they are studying on-site or at a distance.

De Montfort already uses Blackboard as a platform for providing e-learning materials for all students and this is used extensively though not exhaustively in all faculties. It was therefore an obvious choice as the main platform for the MSc. Decisions about the best way to use Blackboard and which other resources to employ alongside it were necessary and as both on-site and distance students study the modules concurrently the experiences need to be as similar as possible.

Some practises have been adopted for all modules and this includes providing physical materials (e.g. textbooks, software, and other materials as necessary). We also record lectures and post them on De Montfort's streaming server (DMUtube). The students are able to view the lectures from within Blackboard and it has proved to be a popular method. Other methods adopted include sound over Power-point slides using tools such as Articulate Presenter; software demonstrations using screen and voice recorders.

[3] define the term 'networked learning' to describe a particular kind of web-based or on-line learning. Their definition of networked learning is "learning in which information and communications technology is used to promote connections: between one learner and other learners; between learners and tutors; between a learning

community and its resources". In adopting this idea of networked learning, it is important for us to make sure that we are not simply providing materials in a variety of forms but that the learning is networked i.e. there is human to human communication taking place within each module. One way that we do this is to make use of an assessed discussion board on our virtual learning environment (VLE). It is assessed based on the number of contributions over the semester rather than the quality of the content. We have found this to be very successful and it is clear that it helps to create a virtual learning community amongst our students. Such communities are identified as being important for student engagement in e-learning by [2]. Our experience of using this mechanism has shown that it encourages students to become more of a cohort through communicating with each other whether on-site or at a distance and it helps the distance students in particular to feel less on their own. The discussion board component is worth 10% on every module and it is this that encourages students to use it initially. We find that as they get used to using it they become more involved and often answer each other's questions and so on. Other practises used, though to a lesser extent, are blogs which are used for keeping project journals and also as a way of putting current students in touch with past graduates from the course; a Facebook group; and more recently wikis for sharing subject related ideas and student presentations.

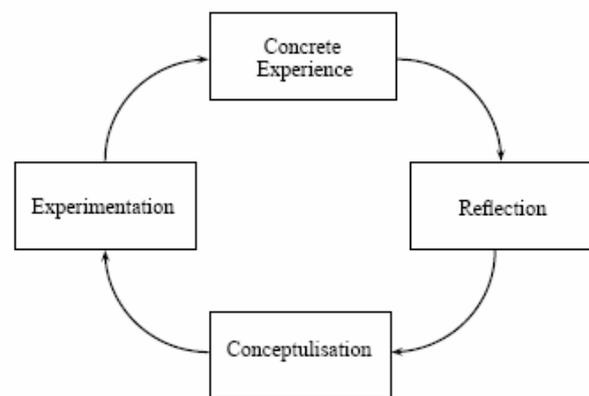


Fig.2 Kolb's learning cycle [6]

In order to deliver the course effectively it has been useful to consider approaches to learning and teaching in higher education more generally. Most of the modules include both theoretical and practical work and the assessments are usually open enough to allow the students to investigate appropriate topics in their own way thus there is an attempt to facilitate experiential learning as defined by [6]. Kolb suggests that learners acquire knowledge according to the learning cycle shown in Figure 2. A further example of this approach in practise can be seen in the design of the OU module, Artificial Intelligence for Technology in [5], [9]. Here they use a learning cycle that is derived from [6]. These can be considered to be constructivist approaches to learning where students construct their own knowledge through various experiences within (and without) of the course. We believe it to be very important for our students to draw on non-course experiences as many of them have work experience: for example, DL students are often in full time employment, there is a wide variety of first degree subjects amongst them and a significant number already have PhDs.

Due to these factors, often our students are interested in applying the knowledge gained from the MSc within their working environment or to their previous subjects or research area.

On-line learning in higher education is also considered by [1], who describes four levels of interaction as part of an on-line curriculum interaction model. Level 1 includes materials presented as text, Powerpoint presentations, videos etc. and relies on the students' own motivation. Level 2 has increased interaction amongst the students such as the discussion board activities used extensively on our MScs. Level 3 includes synchronous activities such as chat rooms and the final level, 4, is where the highest level of the e-learning community is offered and is where the learners and instructors engage in a variety of synchronous activities. Level 4 is achieved to some extent on our courses when we hold meetings, presentations, demonstrations (usually using Skype) with tutors and students. We plan to increase this as the course evolves further.

[2] defines pedagogical models for e-learning namely: open learning, distributed learning, learning communities, communities of practices and knowledge building communities. We believe that our approach incorporates aspects of the distributed learning model and to some extent, the learning communities model. Distributed learning is where the learners are in many different locations and can choose to study at times that suit them. Communication with the faculty staff and other students is through a variety of electronic means. Learning communities are groups of people who support each other in their learning activities and can be broadly defined as including "any social network or infrastructure that brings people together to share and pursue knowledge" [2]

The next section focuses on how we approach the delivery of the Mobile Robots module on the MSc.

3. The Mobile Robots Module

To be successful the mobile robotics module must combine hands-on practical work with advanced theoretical concepts. The teaching and assessment strategies have to work face to face and at a distance. For many students this module is their first exposure to programming robots and the first time they have come across the inherent challenges such as hardware limitations, behavioural debugging and dealing with uncertainty. To best support our diverse student population we have developed a clear delivery strategy which we believe serves as a model when delivering a first semester postgraduate robotics module. Our strategy is depicted in Figure 3.

3.1 Establishing a solid knowledge and skills base

Arguably the most important and probably the most difficult to teach at a distance part of the module is the first two weeks. It is vital that students come out of these first two weeks with the core knowledge and skills to make progress on the module. The students come on the module from a diverse set of backgrounds, some may have good knowledge of the topics covered in these first two weeks, others may have limited or no experience. For on-site students it is relatively easy to judge a student's starting level through body language and informal questions. For distance learners a different approach must be taken. We supply a range of learning materials in these early weeks, the compulsory material covers topics at a fairly high and

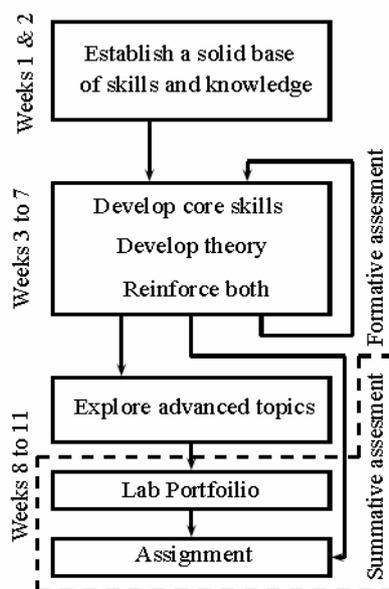


Fig.3 Teaching and assessment strategy for mobile robots

Abstracted level but contains pointers to deeper material which gives more detailed explanations and worked examples. Students are strongly encouraged to dig down in the material until they are confident in their understanding and are able and motivated enough to do this. To establish the core competencies needed by the students a set of multimedia materials are provided to the students. These materials cover what might be termed housekeeping issues, but are essential to progress in the module, topics:

- Building the robot model
 - Changing the batteries in the robot.
 - Updating the robot's firmware.
 - Basic operation of the robot.
 - Installing the BricxCC IDE. Using the BricxCC IDE: writing your first program, compiling, uploading and executing.
 - Installing GCC with OpenGL and OpenCV.
 - Setting up compiler short cuts and makefiles
- The media covering these topics are videos of lectures, videos of staff using the robots, video tutorials of software, lecture slides, documentary notes and textbook sections. An example taken from this material is given in Figure 4 which is taken from a set of instructions detailing how to modify the default Lego model for use in the module.

3.2 Developing Core Skills and Theoretical Underpinning

Through weeks 3 to 7 students are presented with a range of theoretical topics:

- Sensors and actuators.
- Low level control.
- Real-time programming.
- High-level control.
- Behavioural control architectures.

The content and delivery pattern of this phase of the module is cumulative by design: each topic depends on an understanding of the previous topic.

Ultrasound ranger and colour light sensors

These are the parts you need:



Put them together as shown below



This is the complete module, not yet attached to the robot

Fig.4 Instructions for robot modifications

From a technical standpoint this can be seen as a bottom up approach to teaching robotics. We start from a basic understanding of how a reading of an environmental phenomenon can be taken by a machine, through control strategies for simple actuators, programming issues associated with such devices to higher level, abstracted control strategies. One deliberately take this approach to avoid glossing over important issues and sources of uncertainties in mobile robots. We could take the opposite approach and begin with a high level view and then drill down to what is really happening. We have chosen the bottom up approach as it gives students an explanation for the idiosyncrasies of robot control from the outset. When the students come to write a high level control program, let's say obstacle avoidance using an ultrasonic range sensor, and the robot fails, crashing into an obstacle, the students already know what may have caused the fault. They are aware that different material reflect sound in different ways, that ultrasonics sensors have conical wavefronts not perfect straight lines and that perhaps the thread checking for obstacles is not run frequently enough.

One very important aspect of this phase of the module for distance learners is the high level of formative assessment and feedback given on a weekly basis. The students undertake a lab based piece of work every week which in some way gives a practical insight into the theoretical material. This lab work is assessed and marks and feedback are given to the students using the Blackboard virtual

learning environment. The grades are purely formative and give the students a clear measure of the level they are working at and what they can do to improve. It is important that the deliverables for these formative labs are short and concise or the level of marking quickly becomes burdensome. Clear guidelines are given to the students in this regards and lengthy submissions penalised. Two of these lab pieces form part of the summative assessment going in to the lab portfolio as discussed later on.

At the end of week 7, students have covered all the core aspects of mobile robots in theory and practice. They are aware of the issues that arise when working with robotics and have a practical experience of working with robots.

3.3 Exploring advanced topics

Having established all the key knowledge and skills the students need to meet the module learning objectives, we then take a brief look at some of the more challenging topics in robotics namely:

Robot/computer vision.

Collaborative robots.

Computational intelligence in robotics.

The second module, intelligent mobile robots (see Figure 1) covers in detail what most academics consider to be the advance topics in robotics: navigation, localisation and path planning.

These topics are not covered in the mobile robot module, where we look at this different set of advanced issues. Each of the subject areas listed above is covered by one weeks worth of materials.

The lab work on vision systems and collaboration is summatively assessed as part of the lab portfolio, giving the best students an opportunity to excel.

A significantly advanced piece of software is provided to the students on each of these topics. Since only one week is given over to each of these topics, it is unreasonable to expect any of the students to begin work on any of these topics from a blank sheet of paper.

For the robot vision lab the students are given a working face recognition program which uses hue masks [7] and morphology operators [8] to identify a human face. The software makes extensive use of the OpenCV library.

The students task is to choose an item for which they will modify the face recognition program so that it recognises this new object. The lab brief gives the best students the opportunity to showcase their technical skills and the knowledge of scientific method. Results from experiments showing classification rates and statistical analysis are not uncommon amongst the top 20%.

For the collaborative robot lab the students are given a simulated blackboard [4] server and four simulated clients who connect to the blackboard\footnote (not to be confused with the Blackboard virtual learning environment discussed in this paper) via TCP/IP. All the networking is taken care of and the students' task is to decide what information should be transmitted and when that information should be transmitted. Students at the lower end of the spectrum tend to struggle with this work, although most get a pass mark. Students at the high end take the work much further implementing multi-threaded communication systems and advanced visualisation tools.

The final topic covered on the module, computational intelligence and robotics, is only covered at the theoretical level. The didactic material covers areas where computational intelligence has been shown to be useful in

robotics with examples from the literature and from work in our own lab.

3.4 Assessment

Assessment of robotics work is generally challenging, these challenges are compounded when assessing work from distance learners.

Our assessment rationale is clear -- we assess each student against clear set of learning outcomes.

On completion of this module, the student should be able to: Demonstrate a comprehensive understanding of the principles and techniques used in building and controlling autonomous mobile robots by the design and implementation of adaptable controllers for autonomous mobile robots on a real robot system.

Demonstrate a comprehensive understanding of the theoretical principles of the techniques used in building and controlling autonomous mobile robots and of the advances that are being made in this field.

The scale of assessment must clearly differentiate between pass and fail and between pass and distinction.

To pass, a student must demonstrate that they have met the learning outcomes.

To achieve a distinction students must meet the learning outcomes, show high levels of skill in the controller design and implementation and demonstrate a deep theoretical understanding of the issues covered on the module.

This summative assessment against these criteria is done with two submissions, the lab portfolio and the assignment.

The lab portfolio contains work from weeks 4, 6, 8 and 9.

These labs allow the students to demonstrate a breadth of understanding: week 4 covers low level programming and sensors, week 6 covers behavioural control, week 8 covers vision systems and week 9 covers collaborative robotics.

Weeks 4 and 6 labs are relatively straight forward and weeks 8 and 9 are more challenging. It is important that the portfolio covers a wide breadth of topics and that the assessment allows the students to demonstrate their theoretical understanding of the work covered. The assignment is submitted after the main teaching period.

The students have to build a robot controller to perform line following using a behavioural architecture of their choice.

This allows each student to demonstrate all the technical knowledge they have acquired throughout the module i.e.

They must choose an appropriate architecture and justify that choice.

They must build a controller to behave in line with the specification.

When the controller fails they should show an understanding of why the failure occurred.

They should attempt to measure the performance of their controller, choosing appropriate metrics. Some of these are easily assessed through documentary evidence, however controller performance needs to be demonstrated through the real-time operation of the robots. For on-site students this is done through formal demonstrations, for distance learning we offer them a live video demonstration (usually via Skype) or allow them to submit a video of their controller on their robot with audio commentary on the robot's performance.

3.5 Student Performance

Table 1 gives the student numbers and pass rates on the mobile robots module over that past three years and table 2 distils these numbers in to fail, pass and distinction rates for on-site (OS) and distance learners (DL). It seems that study at a distance presents no barrier to students achieving the highest standards on this module, in fact a slightly higher

rate of distance students achieve a distinction on this module.

	07/08		08/09		09/10	
Numbers	OS	DL	OS	DL	OS	DL
Enrolled	5	0	0	6	9	6
Failed	1	0	0	0	0	0
Pass	1	0	0	0	5	4
Distinction	2	0	0	5	4	2
Deferred	0	0	0	1	0	0

Tab.1 Student performance 2007-2010

	OS	DL
Failed	8%	0%
Pass	46%	36%
Distinction	46%	64%

Tab.2 Student performance rates 2007-2010

4. Conclusions

Delivering courses at a distance is a topical area. With the many available mechanisms for interacting with learners electronically there are a number of choices to be made regarding the approach to take. In this paper we have described some of the decisions we made when developing the MSc Intelligent Systems and the MSc Intelligent Systems and Robotics for on-site and distance delivery. We have provided a case study of how this applies to one of the most practical modules, namely, Mobile Robots.

We have discussed our strategy for the delivery of this modules namely: firm basis of practical skills, build theory and practical with frequent feedback and give space to those most able to push their skills and knowledge as far as they wish to. The pass (more so the distinction) rates give over the past three years show how successful this final point has been, particularly for distance learning students. We believe that by following this model it is possible to teach a technical, practical subject through a distance learning model, and shown that a lack of contact is no obstacle for well motivated and determined students.

The module and the course are successful and sustainable with a total of 55 students currently enrolled (11 on site, the rest as distance learners). The course continues to evolve as the available technologies improve; additionally we gather feedback from our students regularly, using the responses to inform future developments. We hope to continue in this way ensuring that our students benefit from a carefully crafted course that makes appropriate use of current e-learning research and associated technology.

References

- [1] CHAVES,C.: On-Line Course Curricula and Interactional Strategies: The Foundations and Extensions to Adult e-Learning Communities, European Journal of Open and Distance Learning, 2006.
- [2] DABBAGH, N.: Pedagogical Models for E-Learning: A Theory-Based Design Framework, International Journal of Technology in Teaching and Learning, 1:25-44, 2005

[3] GOODYEAR, P., BANKS, S., HODGSON, V, McCONNELL, D.: *Advances in Research on Networked Learning*, pages 1-10, Springer, 2008.

[4] HAYES-ROTH, B.: A blackboard architecture for control, *Artificial Intelligence*, 26:251-321, 1985

[5] HOPGOOD, A., HIRST, A.: Keeping a distance-education course current through elearning and contextual assessment, *IEEE Transactions on Education*, 50: 85-96, 2007

[6] KOLB, D.: *Experiential Learning: experience as the source of learning and development* New Jersey: Prentice-Hall, 1984

[7] SHAPIRO, L., STOCKMAN, G.: *Computer Vision*, Prentice-Hall, 2001

[8] TARSHA-KURDI, F., LANDES, T., GRUSSENMEYER, P.: Hough-transform and extended RANSAC algorithms for automatic detection of 3d building roof planes from Lidar data, *The Photogrammetric Journal of Finland*, 21(1):97-109, 2008

[9] Weller, M., Hopgood, A. Implementing a learning model for a practical subject in distance education, *European Journal of Engineering Education*, 22:377-387, 1997

Dr. Jenny Carter

Dr. Simon Coupland

Centre for Computational Intelligence

Faculty of Technology

De Montfort University

The Gateway

Leicester LE2 2RD

Phone: +44 116 2506449

E-mail: jennyc@dmu.ac.uk, simonc@dmu.ac.uk

Autonomous Guided Vehicles Applied to Industrial Engineering and Management Studies

André Dias, Nuno Dias, Daniela Campos, Hugo Ferreira

Abstract

This article presents a framework to an Industrial Engineering and Management Science course from School of Management and Industrial Studies using Autonomous Ground Vehicles (AGV) to supply materials to a production line as an experimental setup for the students to acquire knowledge in the production robotics area. The students must be capable to understand and put into good use several concepts that will be of utmost importance in their professional life such as critical decisions regarding the study, development and implementation of a production line. The main focus is a production line using AGVs, where the students are required to address several topics such as: sensors actuators, controllers and an high level management and optimization software. The presented framework brings to the robotics teaching community methodologies that allow students from different backgrounds that normally don't experiment with the robotics concepts in practice due to the big gap between theory and practice, to go straight to "making" robotics. Our aim was to suppress the minimum start point level thus allowing any student to fully experience robotics with little background knowledge.

Keywords: production line simulation, autonomous guided vehicles, teaching robotics

Introduction

Simulation environments achieve great success in robotics learning especially when the students are at the beginning of their studies. Simulation is very useful for several reasons: students are just starting to learn robotics, the hardware is not ready, the operations place is very far away or inaccessible (e.g. other planets), the setup is often not operational (and to make it operational it can be difficult and time consuming), or even when the developers only want to test some small things and don't want to wait all the time needed for starting up the robots.

When students are just beginning to learn robotics, dealing with the hardware is complicated and simulation can teach the basics of robotics without having to deal with hardware problems.

The input focus of this framework was an Industrial Engineering and Management Science course in which students are required to learn how a production line works as well as everything around it. The students need to study production lines using robotics transporters.

The aim is to use Autonomous Ground Vehicles to supply materials to a production line as an experimental setup for the students to acquire knowledge in the production robotics area. Students are expected not just to know the necessary concepts of industrial management but to understand how to apply them to a real-life problem/challenge.

The main objective is stock management in a production line using AGVs, where the students are required to address several topics such as: sensors, actuators, controllers and an high level management and optimization software. Introducing the students to robotics requiring almost no prior

knowledge on the subject was a challenge, therefore a two step approach was used: a simulation one and a practical one. There are several works presenting the simulation of a production line using AGVs [1][2], and several presenting studies about the dynamics of AGVs [3][4] but the solution presented allows the student to learn about the flow of materials in the production line, electronics and AGVs dynamics.

The paper is organized as follows: Section II describes the proposed architecture. Section III describes the experimental setup, developed to support the previous assumptions by having a two step approach: a simulation one and a practical one. Section IV provides some conclusions and discusses future work.

1. Proposed Architecture

The proposed architecture main objective was to develop a framework capable of managing the automated transport of materials in any given production line. In this framework it was necessary to keep in mind that first year industrial management engineering students have no prior knowledge of robotics and little knowledge in electronics, therefore, the proposed architecture had to be easy to interact with. The main architecture of our framework can be seen in Fig. 1.

The typical interaction used in the industry to manage the production is an Enterprise Resource Planning (ERP) which uses a backend database. Our approach was to use a database to interact with both the simulation environment and the application developed by the students. In this approach the students are expected to develop an application that interacts with the database according to their industrial management scenario.

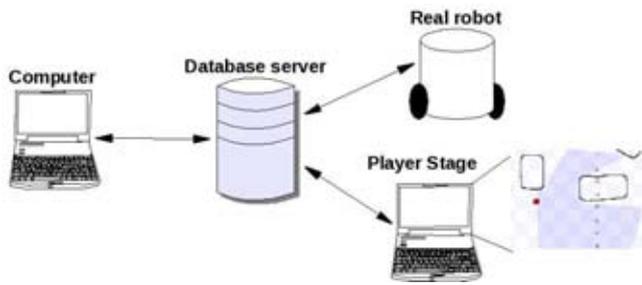


Fig. 1 Global Architecture

The industrial management scenario is an industrial plant with physical representation of the stock area (both of raw materials and finished products) and the number of existing production lines. All the supply and collect points are marked and tagged. The idea is that the students, using this scenario, would develop an application that determines the stock quantity of raw materials, the production rhythm (the maximum production rate of the production line is given), the stock quantity of finished products and controls the supply of materials to and from the production lines.

The supply of materials is done using AGVs. Therefore, the students need to take into account the AGV dynamics since it adds time restraints to the supply of materials which can lead to lower production rates. The control of the AGV dynamics and route planning and optimization is not an objective for these first year students. Moreover, the simulation platform was chose so that these features could be added in future work to be used by more advanced students.

All the location points where the materials are collected or supplied are marked in the scenario. This is done by adding tags in a database. This database is given to the students and has tags for every supply or collect point and the initial quantity of materials in each point.

The students application interacts with this database to implement their supply management strategies. This application uses a known open source tool from robotics community: Player project [6] (formerly known as Player/Stage).

This framework is composed by 3 layers:

- Player Project
- Database
- Application Program

1.1 Player Project

The Player project [6][5] is a multirobotic simulator (see Fig. 2) that uses a client/server model to manage the robot interface. It is possible to interact with a simple sensor or with an entire robot. There are several robots, devices and algorithms predefined in the original source but, if needed, new ones can be created.

The Player is capable to perform tasks by dealing with sensors and actuators interfaces (drivers) being simulated or from a real robot.

The Stage is a multirobot simulator in a two dimensional world. This software package integrates already a vast list of sensors models that can be easily modified and new sensors can be added.

The Stage can talk with the Player allowing robot simulation if there isn't any real robot available.

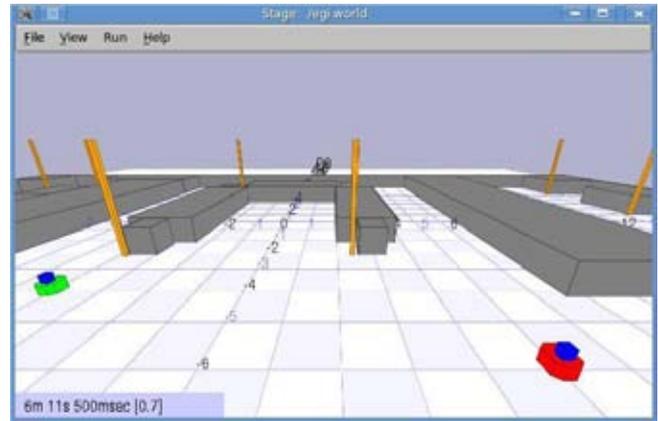


Fig. 2 The simulation scenario with two production lines in Player Project

The use of Player project allows the students and developers to avoid the time consuming task of developing the physics simulator and is a project with constant development, well established and flexible enough to add new robots such as AGV or robotic arms.

Using a powerful tool such as Player project enables the further development of this framework, allowing the students to edit the industrial plant, adjust the number of production lines and AGVs, control the AGV dynamics and perform route planning and optimization tasks.

1.2 Database

In a typical simulation environment such as the proposed Player project, the interaction with the simulator is made by using a UDP socket communication. This communication layer requires knowledge that is beyond the scope of the course in which this framework was implemented.

Therefore, a database was developed in an open-source platform, MySQL, that will be used as the information layer, allowing the simulation environment to interact with the students applications.

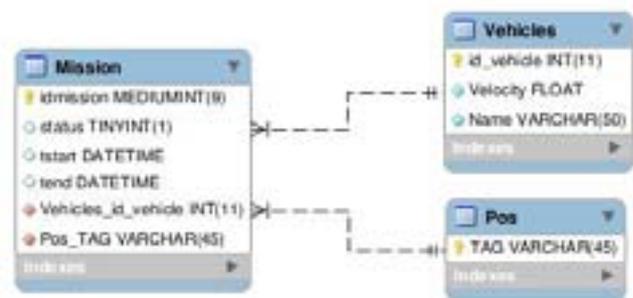


Fig. 3 Normalized implemented database

The database is composed by three tables (see Fig. 3) two of them are associated with the simulation environment (*Vehicles* and *Pos*) and the third to the dynamic of the vehicles in the product line (*Mission*).

The tables associated to the simulation environment are loaded with information regarding the AGVs available in each scenario and the supply/lifting end product positions in the warehouse.

The database also allows the interaction with standard platforms or hardware solutions designed by the students without having to extend their knowledge on wireless networks or other type of communications.

1.3 Application Program

The application program developed in GTK+ [8] is an interface that allows the student to initiate their simulation work (see Fig. 4). The GTK+ toolkit was chosen because it is open-source and cross-platform which is important considering the different operating systems that students use.



Fig. 4 Management Software

This application was developed to be user friendly, having only two main controls responsible for starting and stopping the simulation, encapsulating all the hard work that is needed by the simulation environment. It is responsible for managing the requests from the database and interact with the simulator producing all the data logs required. Some data logs are showed in real time, so students can understand what is happening in the simulator window, and other logs are output to a file as being final results from the simulation itself.

In order to manage more than one robot a multi-threaded application is used by allocating a thread to each AGV present in the simulation scenario. Each thread is responsible to access the database requesting the mission for the assigned robot and then providing the path to execute the mission. The next mission will only be executed after the last one is completed and all timestamps from the mission are recorded in the database.

2. Experimental Setup

The proposed architecture was made available to the students allowing them to optimize the production line by controlling the AGV's dynamics without any prior knowledge on robotics. This approach allows the students to develop management applications that are able not only to study the AGVs dynamics but also other important themes such as stock and supply chain management which are areas of interest in industrial engineering and management sciences.

Using knowledge acquired in the programming, algorithm and data structuring course, the students develop graphic applications that can parametrize both the missions and the AGVs by interacting with the proposed database available in the framework.

The students' application interacts with the database by sending tags with the information regarding the location where the AGV needs to pick up material or where it needs to deliver the material.

This information is read from the database and fed to the simulation environment where the students can visualize the AGV's movement but also obtain other useful information. The architecture allows the user to configure its' scenario by stipulating the maximum load and speed to the AGV's but also by introducing unique physical characteristics of the

AGV (like wheel diameter, etc) and adjust the AGV's reaction based on a physical model.

2.1 Simulation Environment

The first step is simulation, allowing the students to validate the AGV dynamics relatively to the material supplying. In this phase it is possible to extract important information about the whole system which includes: supply times, route optimization and inventory management.

After starting the simulation environment, that consist of the definition of the layout that is already available (given by the teacher), and the application program installed in a class room, the students are now able to interact with the database.

The layout that is developed by the teacher has a challenge for each student that consists in a representation of all stages of the supply chain process (raw materials, production line and finished product).

The black dots presented in the Fig. 5 are the tags where each AGV has to move. The example illustrates four spots (tags), two of them representing the supply materials (MP1_0002, MP1_0003) and the other two the production line supply points (LP1_0001, LP1_0002).

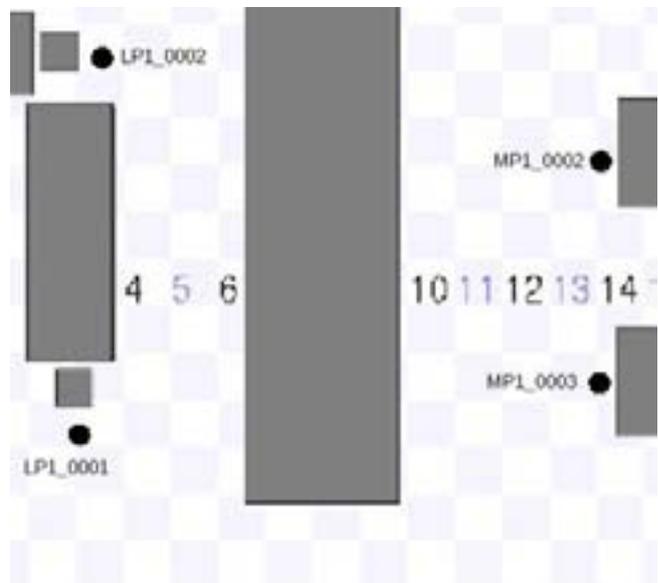


Fig. 5 A small layout example with tags. These tags are locations where the AGV has to move. The grey zones are areas where the AGV cannot pass.

These events (or tags) have to be generated according to the students previous stock management planning and it can also serve as testing ground for different strategies.

With this information, the students start to define the mission in the database. If the objective is to supply the production line LP1_0001, (Fig. 5) from the supply warehouse tagged MP1_0001 the students perform the following SQL command:

```
MYSQL_CONNECT(IP_MACHINE,USERNAME,PASSWORD)
INSERT INTO mission (Pos_Tag_id, Vehicles_id_vehicle)
VALUES(MP1_0001, 1)
INSERT INTO mission(Pos_Tag_id, Vehicles_id_vehicle)
VALUES(LP1_0001, 1)
MYSQL_CLOSE(sock)
```

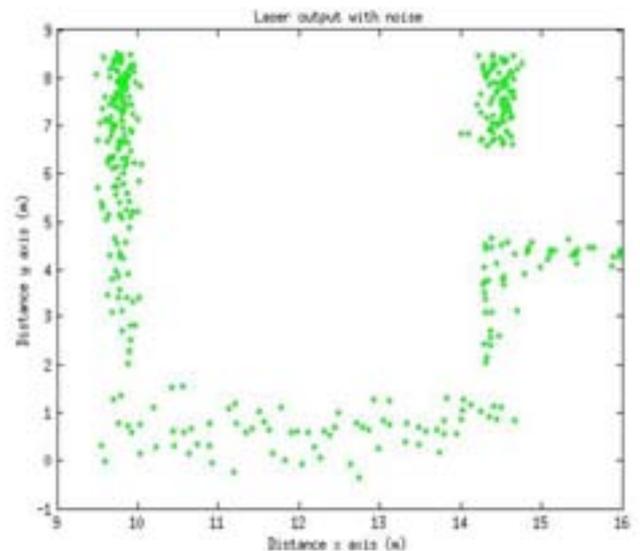
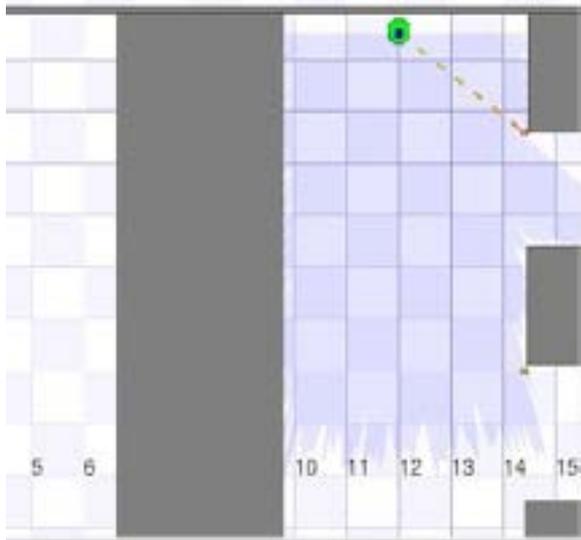


Fig. 6 AGV navigation with laser sensor (left). Environment mapping from AGV laser sensor (right)

The AGV loading, travel and unloading times are managed by the application program defined in the layout setup. Besides this results the students can also learn several robotic topics such as path planning and sensor navigation (see Fig. 6 and Fig. 7).

A major advantage from this simulator is the capacity to evaluate different type of sensors considering the proposed scenarios. Figure 6 represents a sensor behavior with its own characteristic measurement noise.

Students without in depth knowledge in electronics can choose different sensors for different scenarios by testing them, analysing their advantages and disadvantages from a practical point of view.

In Fig. 7 the AGV path and localization process that was simulated by the students can be observed. This is important because they can analyze the results and be able to evaluate different locomotion and navigation methods [7].

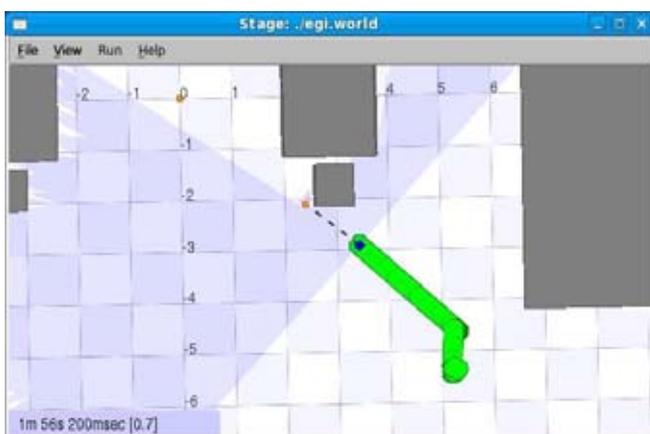


Fig. 7 This is an example of an AGV moving to LP1_0001 tag after receiving a command. The AGV path is also represented.

2.2 Robotic Platform

The second step is the implementation of a small scale robotic platform to be used by the students and that have identical physical characteristics to the simulation step.

This allows the evaluation of the software developed by the students in a practical case as well learning the behavior of sensors, actuators and controllers in a real scenario.

Each robot has a wireless communication system allowing the execution of the same instructions used in the simulation step.

The students can program the robots events the same way they did in the first step, therefore, this step adds no extra difficulty.

Since first year industrial management students don't have the necessary knowledge to implement the robots this step is intended only as a physical simulator of their production line stock management strategies. However it is an important stage to stimulate the students in achieving the best strategies and also to get them to take an interest in robotics.

The implementation of the robotic platforms is intended to be done by the students in the following years.

Conclusions

This article presents a framework for the students to acquire knowledge in the production robotics area using Automated Guided Vehicles.

This work allows the students to acquire skills in robotics area and supply chain management. The students are able to experiment with several practical scenarios and the presented simulation environment gives them an opportunity to see the results of their implementation.

This approach proved to be a good strategy to motivate the students to learn robotics and lead them to want to know more about this area.

Considering the impact of the proposed solution, we would like, in the future, to integrate the simulation environment with an e-learning platform so that it can be accessible by all students at school and at home. Also, the teacher/supervisor could easily add/change scenarios.

Further development of this framework to allow the students to have more control on the variables of industrial production planning was an initial idea and will be done in future work. The concepts necessary to take advantage of these new features are beyond the scope of first year students but this has great potential for more advanced students.

In another line of work, the students can develop their own small scale robot platform to interact with the present framework.

Currently we are working in the integration of Gazebo (a 3D multirobot simulator provided by Player project) in order to achieve more realistic feeling in the simulation output.

Acknowledgements

The authors would like to thank the School of Management and Industrial Studies and Porto Polytechnic. This work is sponsored by ESEIG and IPP. We would also like to acknowledge the support from Autonomous System Laboratory, LSA/ISEP/IPP.

References

- [1] KESEN, S. and BAYKOC, O.: "Simulation of automated guided vehicle (AGV) systems based on just-in-time (JIT) philosophy in a job-shop environment". In *Simulation Modelling Practice and Theory* 15, 272-284, 2007.
- [2] HSUEH, C.: "A simulation study of a bi-directional load-exchangeable automated guided vehicle system". In *Computers and Industrial Engineering* Vol.58, pp.594-601, 2010.
- [3] MARTINEZ-BARBERA, H. and HERRERO-PEREZ, D.: "Autonomous navigation of an automated guided vehicle in industrial environments". In *Robotics and Computer-Integrated Manufacturing* Vol.26, pp.296-311, 2010.

[4] SHAH, M., LIN, L. and NAGI, R.: "A production order-driven AGV control model with object-oriented implementation". In *Computer Integrated Manufacturing Systems* Vol.10 N.1, pp.35-48, 1997.

[5] Player project: <http://www.playerstage.sourceforge.net>

[6] GERKEY, B., VAUGHAN, R. and HOWARD, A.: "The Player Stage Project: Tools for Multi-Robot and Distributed Sensor Systems". In *Proceedings of the 11th International Conference on Advanced Robotics (ICAR 2003)*, pp.317-323, Coimbra, Portugal, June 2003.

[7] SIEGWART, R. and NOURBAKHSH, I. "Introduction to autonomous mobile robots". MIT, 2004.

[8] GTK+: <http://www.gtk.org>

André Dias
Nuno Dias
Daniela Campos
Hugo Ferreira

School of Management and Industrial Studies
Porto Polytechnic
R. D. Sancho 1º nº 981
4480 – 876 Vila do Conde
Phone: 00351 252 291 700
E-mail: adias@eu.ipp.pt, ndias@eu.ipp.pt,
danielacampos@eu.ipp.pt, hugo.ferreira@eu.ipp.pt

Robotour - robotika.cz outdoor delivery challenge

Jiří Iša, Martin Dlouhý

Abstract

In this paper, we present an international contest for autonomous robots: Robotour – robotika.cz outdoor delivery challenge. The main task is a navigation in real-world situations. First three years were held in park Stromovka, Prague, Czech Republic and raised an interest of many teams, media and general public. Last year, the contest started to migrate. To our knowledge, there is no similar European outdoor contest for fully autonomous machines. Note, that there are some common features with American Mini Grand Challenge and a younger Japanese Real World Robot Challenge. The rules of Robotour are described in more detail together with experience gained over the past four years – both from the organizers' and the participants' point of view.

Keywords: autonomous robots, outdoor, international competition

Introduction

Competitions such as Eurobot [1] and DARPA Grand Challenge [2] have repeatedly shown that both young students and senior researchers are attracted by competitive research environments. Autonomous robotics is a multidisciplinary domain which offers educational opportunities and interesting real-world research topics.

In 2004, the American Defense Advanced Research Projects Agency (DARPA) organized the first Grand Challenge. The goal of DARPA was to foster a research in fully autonomous vehicles. In the first year, only 11.78 km of the 240 km long route were completed by the best team. Already in the second year of the competition (2005), five vehicles finished the 212 km long route. This shows a tremendous impact the challenge has had on the field of fully autonomous ground vehicles.

Since 1994, the Eurobot competition attracts many young people (more than 2000 in year 2010) [3]. Eurobot has successfully shown how an international competition can be used to teach young people how to cooperate and how to develop complex systems.

In 2006, the Robotour – robotika.cz outdoor delivery challenge has been founded. In our opinion, the large gap in complexity between Eurobot-like competitions (e.g. RobotChallenge [4], Istrobot [5] and other) and competitions like DARPA Grand Challenge needed to be bridged. In about the same time, other organizers felt similar insufficiency and more competitions were born. Since 2003, Field Robot Event focuses on the agricultural automation [6]. Since 2006, European Land Robotic Trial allows research teams and industrial companies to demonstrate their unmanned outdoor systems in realistic scenarios and terrains [7]. One year after Robotour – in 2007 – Tsukuba Real World Robot Challenge (RWRC) took place in Japan for the first time [8]. Since 2009, a similar straight line outdoor challenge takes place in Písek, Czech Republic [9].

Robotour – robotika.cz outdoor challenge is focused on autonomous ground vehicles and their orientation

in the real-world outdoor environment. The robots perform a delivery task in complex environments of city parks. They are not allowed to leave paved roads. Participants of various background are welcome. In the previous years, students from high schools, university researches and hobbyists took part.

In this paper, we describe the Robotour – robotika.cz outdoor delivery challenge. General rules are covered in Section I. In Section II, we share experience obtained from the organizers' point of view. Reflections of the participants are captured in Section II.

1. Rules

1.1 Historical Overview

The rules for each year change slightly and the contest becomes more and more challenging every year. The unified theme of all years is robot's ability to autonomously navigate in outdoor environments and to move payload from one place to another. The robots have to be fully autonomous, which means that after a task entry they have to control themselves.

Since the first year, the basic requirement is to navigate on paved roads in the park without leaving them – similar to cars not leaving the streets. In the second year, a possibility of robot cooperation was introduced. In the third year, obstacles were added and robots had to deal with them successfully. In the fourth year, robots did not know exactly their start position and had to deal with obstacles more carefully.

The fifth year of this contest should be a next step towards smarter and more autonomous robots. In contrast to the previous years, the robots get only a map and coordinates of the destination. The robots should be able to navigate around the park even if they have never been there before. The map and the destination should be the only information the robots get before the start. Robot successfully solving this task

should be able to demonstrate its ability with a corresponding map in any park.



Fig.1 A simple map of the Lužánky park in Brno given to the participants in 2009.

1.2 Detailed Rules

Task

The task for the robots is to deliver payload in a given limit of 30 minutes to a destination as far as 1 km. Robots must be fully autonomous, not leave a road and choose correct path on junctions. The starting place, starting time and the destination will be the same for all the robots.

Map

Vector map of footpaths in a park will be based on a vectorization of an ortophotomap and teams could improve it further. The basic idea is taken from Open Street Map [10]. A robot is allowed to use only this shared map – all other maps are prohibited!

Robots

A team can deploy multiple robots this year, but only a single designated one is used to compute a score. Every robot must have an emergency stop button, which stops its motion. The button must be easily accessible, red and must form a fixed part of the robot (aka Big Red Switch), so it could be used in a case of a danger. The team must show that it is easy to manipulate with the robot – two people must be able to carry it several tens of meters. There is also a minimal size – robot has to carry 5l beer barrel (at least an empty one).

Leaving the Road

The robots are expected to stay “on the road” which means to stay on the paved passage ways. If any robot leaves the road, its trial ends. The team has to take care of their robot and remove it immediately.

Obstacles

There could be obstacles on the road. Besides natural obstacles like benches there could also be artificial obstacles. A typical (artificial) obstacle is for example a figurant, a banana paper box or another robot. Robots must not touch an obstacle. Contact with an obstacle means an end of a trial. The robot may stop in front of an obstacle and visually or acoustically give a notice. Note, that the robot has to detect, that the obstacle is no longer present.

Robots Interaction

Situations, in which a faster robot catches up with a slower one, will not be explicitly handled. The faster robot can handle the slower robot as an obstacle, i.e. avoid it or wait until the “obstacle” disappears. In general, the road rules will be respected: right of way, avoidance to the right, passing on the left.

Start

All robots will start from the same park road simultaneously. A minimum width of this road is 3 meters. The starting area for each team will measure approx. 1.5 x 1.5 meters. Starting areas will follow one after another on one side of the road. Within the starting area, each team can place its robot as they see fit. The order of the robots on the start is given by their results in the previous round (a better robot will be closer to the destination). The order in the first round will be given by the order of successful homologation. Robots start automatically via their internal timers. During the last minute before the start, no interaction with the robot is allowed.

Score

The team, whose robot manages to proceed along the route best, wins. The aerial distance of the last position of the robot (leaving the road, a collision or a timeout) to the destination is critical. For every meter towards the destination, a team gets one point. If the team carries a payload, its score is doubled (“points for the payload”). Each robot can carry only one “payload”. A 5l beer barrel (full) serves as a payload. In every round, a robot can obtain points at most equal to twice the aerial distance of the start and the destination.

Organization

The contest will consist of four trials for each team. The start and destination will be different for every trial. The selected destination will be announced to all teams 10 minutes before the start. The speed of the robots is not important (actually, it is limited to 2.5 m/s). All points gained during all trials will be summed together. The trial starts at a specified time and ends after 30 minutes. The robot must leave the starting area within 10 minutes of the start. If the robot does not move for 60 seconds its trial ends. Each team has to arrange for one person familiar with the rules that will be part of the referee team during the competition.

Homologation

A team can participate in the contest only if it is able to score at least one point. Another necessary condition is an ability to travel along a 10 meters long route fragment without a collision with any obstacle. The starting procedure will be tested (the automatic start) as well as the functionality of the emergency stop button. Usage of liquids, corrosive or pyrotechnic material as well as live beings is strictly prohibited. Every robot has to be accompanied by a team member, older than 18 years, who is fully responsible for the behavior of the robot.

Technical Documentation

Every team has to provide basic technical documentation about their robot (for presentations, general public and journalists). Three winning teams will be asked for a more detailed description for a website presentation and to make the entry of novices in the following years easier.

2. Organization

Robotour is organized as a three-day event (Friday to Sunday). Friday is dedicated to the testing, clarification of rule details and homologation. During the homologation, we want to make sure that robots are not dangerous, have a functional emergency stop button and are able to gain at least one point in the contest. Saturday is the contest day. Finally, there is a workshop on Sunday. It is after the contest, so the competitors have a fresh experience with their robots and algorithms. They are also not stressed any more and thus this is a good moment for sharing knowledge.

We started to enforce this three-day template after the first competition in 2006. That competition ended on Saturday and most teams left without letting us and other teams know what has worked and what has not. What was even more important was that teams left exhausted from the programming marathon and one team had a car accident on the way home. Since the following year, the workshop is mandatory.

The Robotour contest is relatively self-supporting and the expenses are minimal. There is no special playground – a public park is used instead. There is no need for renting a hall because the event takes place outside. To be precise, some room is necessary as a base for the teams especially in bad weather conditions. It is recommended to have a partner who provides this place, like Planetarium Praha in the first park Stromovka did. A good idea is also a combination with an exhibition of robots and a related technology parallel to the contest.

There is no registration fee, but the teams have to take care of catering and pay an accommodation.¹ Small items remain on the bill: leaflets printing, diplomas, cup for the winners, and a Saturday night dinner. The dinner is usually sponsored and the goal is to unite the teams and give them a chance to relax a little bit after the contest. Note, that prices are rather symbolic, which lowers expenses on one side and also reduces a potential rivalry between the teams.

2.1 Duties over the Year

The first task of the organizers is a precise specification of rules for the next contest. They are presented on the robotika.cz website in Czech and English languages. The core remains the same (autonomously navigate in a park) and the changes are usually a consequence of a discussion at the workshop and experience gained.

The second task is to ensure an affordable accommodation for a relatively large group of people (50 people needed accommodation in 2009). An agreement with a university dormitory serves well. The reservation must be performed usually a month in advance and that defines a clear deadline for the registration of the teams.

Finally, it is necessary to find an interesting park, manage permission for the contest day and find building with large enough room(s) for team base with many electric outlets.

2.2 Experience of the Organizers

There were couple lectures we have learnt over the last four years organizing Robotour (and previously several

¹ Accommodation is usually partially or fully sponsored.

years of organizing Czech Cup of Eurobot). The basic scenario was already mentioned and serves good and is worth a recommendation. What has changed over the years are two major trends: the number of teams is increasing and the task is getting more difficult. In the first case, we tried to find some optimal timetable of the rounds and we are still not satisfied. What suits the teams does not suit a general audience and vice versa. This year, we will start all the robots from one place simultaneously, which could be attractive for spectators, but may cause problems to many teams.



Fig 2 Robot of the R-Team (left) leading the allied robot of RobSys (right).

The task complexity is another issue. Beginners have a harder position to enter the contest every year. For 2010, we discussed a new category (WagonOpen), but we will probably cancel it. The reason is a new, for the beginners with outdoor robots highly recommended contest “Robotem rovně” (Robot, go straight!) in Písek. In Písek, the task is to navigate as far as possible on a 3 meters wide and 300 meters long park road. This is exactly the first stage which is necessary to enter the Robotour contest.

3 Reflections

3.1 Questions

To reflect an influence the competition has had on its participants, we have asked some of the past successful teams few questions:

1. What did you expect from the competition?
2. What did the competitions give you?
3. What were you disappointed with?

3.2 Asked teams

The following teams were asked:

- Propeler-team**, Opava: A group of high school students, who placed 2nd in 2006.
- LEE**, Prague: Researchers and students from Czech Technical University in Prague. Winners of the year 2008 and the year 2009.
- R-team**, Rychnov nad Kněžnou: A team of a high school teacher. Since 2010, he organizes RobotOrienteering in Rychnov nad Kněžnou. R-team finished 2nd in 2008 (in a coalition with the RobSys team, see Figure 2).
- Roboauto**, Brno: A self-funded group of researchers, which ranked 2nd in 2009.

•**Radioklub Písek**, Písek: Hobbyists and professionals, who also teach electronics in a club. Radioklub Písek got a 3rd place in 2009. Since 2009, the club organizes Robotem rovně (mentioned in Section II).

3.3 Answers

1. What did you expect from the competition?

•Propeler-team:

- The competition motivated us to build our first robot.
- Having almost no restriction on the dimensions of the robot allowed for a simple construction – We could use a notebook, get an image from a camera and use a bought chip to control the motor and the servo (we did not understand microchips and servos at that time).

•LEE:

- We wanted to see a comparison of several approaches to the mobile robotics.
- The competition gives us an opportunity to have our solution judged in an unbiased fashion.

•R-team:

- After Istrobot and Eurobot, I wanted to try something new.

•Roboauto:

- The competition served as a motivation to finish a functional version of algorithms and of the robot.
- We wanted to present our results to a general public.
- We expected to meet with a like-minded community.

•Radioklub Písek:

- After seeing the robots in 2007, we believed we could do better.

2. What did the competition give you?

•Propeler-team:

- We met people in the same domain of interest, saw their approach and other technology.
- Every year, we have a motivation to catch up with our first result.

•LEE:

- We have seen, how a relatively simple solution (by R-team) can solve a given task.
- We realized that the increasing accuracy of hardware and sensors can have a huge impact on the accuracy of simultaneous localization and mapping.
- We have been shown, how important it is to deal with the technical details and with the reliability of the robots.

•R-team:

- I have learned that even the hardware is not fully reliable. Indoor robots do not suffer from such problems.
- I realized how difficult the task is, even though I have expected some difficulties even beforehand.

•Roboauto:

- It has fulfilled our expectation.
- The competition gave us a practical experience with deploying a robot.
- We have got an inspiration for further improvements of the hardware and algorithms.
- We feel in touch with people with similar interests.

•Radioklub Písek:

- We realized the competition is not as simple as it seemed for the first look and few others.

3. What were you disappointed with?

•Propeler-team:

- We are not really disappointed: When the robot works, everything is fine.

- Answering the question “What does the robot do?” is difficult, when the task difficulty is not obvious.

•LEE:

- Although there is a lot written by the competitors at robotika.cz, every year someone new comes and repeats previous mistakes.

•R-team:

- In my opinion, the competition has become too difficult. Only one or two best teams can fully cope with the rules.

•Roboauto:

- Problems with a reliability and with a robustness are bigger than we have expected.
- We are disappointed with only a small media attention.

- We hoped to get an attention of potential sponsors or future team members, which has not happened so far.

•Radioklub Písek:

- We are sad that the cooperation of multiple robots is not encouraged any more. We have learned several interesting things doing that. On the other hand, as the competition evolves, it does not suffice to copy a solution from the previous year.

3. Summary

We have introduced Robotour – robotika.cz outdoor delivery challenge, its rules and their evolution over the time. We share experience gained while organizing several years of the competition and show several patterns worth following. The competition has been successful in attracting people to robotics and giving them an opportunity to learn. The contestants enjoy a chance to meet others, exchange ideas and compare their approaches in an independent manner. As the competitors note, while seemingly simple, the competition became difficult to participate in. This in turn led to a creation of two new robotic competitions in Czech Republic, which differ in the level of difficulty. Currently, there exists an evolutionary path for a person interested in robotics through these outdoor competitions up to Robotour and possibly even further.

References

- [1] Eurobot, “Contest for amateur robotics”, 1997.
- [2] DARPA, “Grand Challenge”, 2004.
- [3] David Obdržálek, “Eurobot 2010”, Robot Revue, May 2010.
- [4] Robot Challenge, “Indoor robotic competitions”, 2010.
- [5] Istrobot, “Contest for amateur robotics”, 2010.
- [6] Field Robot Event, “Agricultural robotic competition”, 2010.
- [7] ELROB, “1st European Land-Robot Trial”, 2006.

[8] Real World Robot Challenge, “Outdoor robotic challenge”, 2009.

[9] Robotem Rovně, “Outdoor robotic challenge”, 2010.

[10] Open Street Map, “Public mapping activity”, 2010.

Mgr. Jiří Iša

Charles University in Prague
Faculty of Mathematics and Physics
Malostranské nám. 25
118 00 Praha 1, Czech Republic
E-mail: isa<at>ktiml.mff.cuni.cz

RNDr. Martin Dlouhý, Phd.

MapFactor s.r.o
Štefánikova 24
150 00 Praha 5, Czech Republic
E-mail: md<at>robotika.cz

EDURO - Mobile Robotic Platform For Education

Martin Dlouhý, Jan Roubíček, Tomáš Roubíček

Abstract

Eduro is a modular mid-size mobile robotic platform designed as both a teaching tool for higher education and a research platform for academia and industry. In this paper we describe the technology used within the Eduro (indoor) and Eduro Maxi HD (outdoor) product lines. Both platforms are designed around a tricycle base with two differentially driven wheels and one caster wheel. The on-board electronics consists of smart sensors and actuators connected by a CAN bus. The main controller module is implemented as a single board x86- based computer running Linux OS. This platform participated in several competitions including Eurobot, RobotChallenge and Robotour.

Keywords: education robot, mobile platform, CANopen

Introduction

Eduro is a generic robotic platform intended for education and research. It was initially created as a teaching tool for Charles University, Prague in 2007, further development has continued independently at the initiative of the development team. At that time, none of the commercially available robots met the low cost/high performance requirements posed by the University's educators.

The Eduro platform is a successor of older platforms Berta, Daisy and Explorer. Berta - a triangle-shaped robot with vacuuming extension - won the 1st Annual Cleaning Contest in 2002, Lausanne, Switzerland [1]. The same triangular base was used in Daisy, a robot which ranked 7th at Eurobot 2003 in La Ferte Bernard, France [2]. Finally, the outdoor prototype Explorer - a 4-wheel waterproof robot - was demonstrated on Robotour 2006 in Prague [3].

The basic idea was to develop a platform that is highly modular on three levels: mechanics, electronics and software. The mechanics is designed as a construction kit with numerous mounting holes with pressed nuts. The electronic is based on a set of independent modules connected via a CAN bus. Finally, the low-level software modularity is achieved through the CANopen protocol and high-level modularity is facilitated through Player devices.

This paper is structured as follows: Section 1 describes the hardware platform in more detail. Section 2 outlines the software. Finally, examples of various configurations are provided in Section 3.

1. Hardware

1.1 Mechanics

The base of the robot is a construction module that includes the battery and motors. Modules such as the caster wheel or the control panel with buttons and indicators are attached to the base. These modules are made from aluminium

profiles and sheets and have many mounting points for simple extending.

The Eduro is not waterproof by default but outdoor versions can be optionally sealed against dust and water.

Rugged plastic wheels are used in the indoor robot design. Such wheels have very good contact properties while they are still sufficiently sturdy for reliable encoder measurements. In outdoor scenario we tested two sets of wheels. The first approach involves smooth inflatable wheels with shallow tread pattern as demonstrated in Field Robot Event 2010 in Germany. These wheels were found suitable for park roads and other easy terrain. The second option utilizes arrow shaped wheels commonly used for small ploughing tractors. These wheels are recommended for rough terrain. They performed very well on muddy terrain when tested on RoboOrienteering contest. In the case of uneven surface with steep slopes, 4- wheel-drive configuration becomes a necessity.

1.2 Drive

Current members of the Eduro product line use SMAC (Stepper Motor - Adaptive Control) drives. This is original Robsys technology for gearless drives, which are based on closed loop controlled stepper motors. The motors are attached directly to driven wheel for indoor robots (Eduro) or by simple belt transmission for outdoor robots (Eduro Maxi HD). This gearless solution is very durable and easily withstands operations by unexperienced students.

Simple speed control with interpolation is presently used. The control system sends speed commands periodically, speed is represented by a 16bit signed integer, where 1000 corresponds to one shaft rotation. The drive sends back an encoder value - 32bit signed integer, where 65536 means one revolution of the motor. The drive has preset software limits for maximal acceleration and speed. Smooth motion can be obtained for speeds between 1 cm/s to 2 m/s given 150 mm wheels (diameter). It is recommended to maintain continuous flow of speed requests such that the wheels remain in permanent contact with the ground. The drives have implemented a communication watchdog which stops motors if speed command is not received within a predefined period.

1.3 Processing Power

The brain of the robot is a single-board computer running Linux OS. The computer is equipped with AMD Geode CPU running at 500 MHz, 256 MB RAM, compact flash card, wi-fi, 3 Ethernet, 1 RS232 and 2 USB ports. RS232 port is dedicated for CAN bus connection via transparent RS232-CAN bridge. High data throughput without data loss is secured by real-time serial driver.

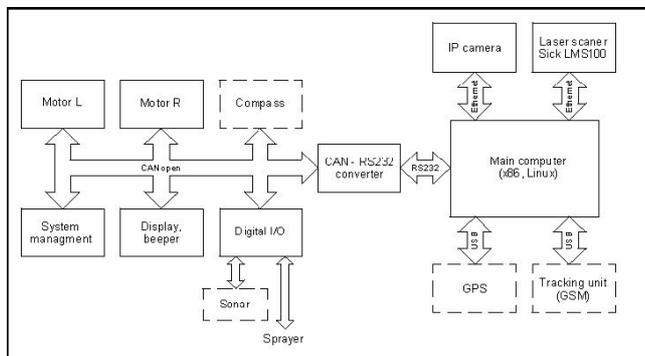


Fig.1 Hardware structure

1.4 Communication network

The Eduro uses CAN bus as its main communication network. All sensors and actuators with low data rate requirements are connected through the CAN. CANopen is the preferred communication protocol but other proprietary protocols can be used as well. CANopen is widely used in industry and hence many available sensors are directly compatible.

Cameras, laser range finders and other sensors with high data throughput are connected directly to the main computer via Ethernet or USB. Except for CAN and Ethernet, I2C and 1-wire Dallas buses are used in robots. I2C is a widespread interface for low-cost sensors, therefore it is supported. However I2C is not designed for large distances (I2C = inter-integrated circuits), therefore it is used only for short local buses. The 1-wire is used for a diagnostic network and advanced power management. Distributed power switches, thermometers, battery chips and other simple sensors and modules are connected via the 1-wire bus. I2C and 1-wire bus are connected to the CANopen network through the gateways.

1.5 Energy source and power management

The power supply is provided by sealed lead acid batteries. The outdoor version Eduro Maxi uses two 12 V/8 Ah batteries while for smaller indoor robot one third of the capacity is sufficient. The whole robot uses a single power source to simplify management. The motors are powered from 24 V supply branch, directly from batteries. The main computer, CAN network and most of sensors are powered from a stabilized 12 V branch. The auxiliary 5 V power supply is present for simple connection of the low cost sensors.

A standard off-the-self charger allows continuous charging while the robot is in operation (e.g., code debugging). When compared to other platforms it allows several hours of autonomous operation and swapping batteries is usually not necessary although it is possible.

An important part of any mobile robotic platform is power management. Energy is a limited resource and thus it needs to be monitored regularly. Two level power management is used in robots. The base is a standalone electronics providing basic function as charging, voltage monitoring and

power distribution. An optional module is connected to the CAN bus and adds remote monitoring and advanced functions. The module sends messages about system voltages, temperatures and other important information. When the voltage falls below the given threshold, temperature rises or other exception occurs, the module can automatically blink LEDs, turn on the beeper or even turn off motors independently on the main computer. The thresholds as well as the consequent actions can be preconfigured from the control software via CANopen. RF remote key is an invaluable accessory to the power management module and allows the operator to turn the robot off in case of an emergency.

1.5 Sensors

This section describes sensors which are used in robots and are connected via the CAN bus. A robot often includes other sensors such as cameras, laser range finders, a GPS unit, etc. These sensors are connected directly to the main computer via Ethernet or USB.

1) Compass: A compass is a part of the inertial unit. Currently, we use a two-axis compass HMC6352 from Honeywell. It is a one chip solution with I2C bus, however the chip is not visible from CAN. The data from compass and other sensors are periodically polled by the CAN module, processed and only then forwarded to the central unit. The azimuth readings from the sensor are converted into 1/100th of degree and sent over CAN bus as 16bit integer. The update rate is 20 Hz.

The sensor itself is represented as one of the layers in the "sandwich" of inertial unit, other layers can include an accelerometer or a gyroscope. The HMC6352 is only a twoaxis magnetometer, therefore tilt compensation is not possible. We plan to integrate a three-axes magnetometer to facilitate tilt compensation in the future.

The inertial unit including the compass is mounted on top of the pole away from sources of magnetic fields and ferromagnetic objects. The module itself is covered by a plastic case and no steel parts are used. During experiments we observed substantial changes in sensor readings caused even seemingly minor attachments to the pole such as a small umbrella, therefore caution is needed. A presence of ferromagnetic objects can be compensated by the system but that requires recalibration and usage of non-linear transformation.

2) "Sharps" distance sensors: "Sharps" distance sensors are cheap IR triangulating sensors for distance measurement. They are often used for obstacle detection and simple navigation in indoor. They have an analog or binary outputs with various operation ranges. The analog or binary signal is routed to universal CAN I/O module SC-DM04. This module has four analog or digital inputs and four digital outputs. The module can have additional function, for example outputs for RC servos or switch array decoder.

3) Sonar: Sonar is another sensor which can be connected via the universal I/O module SC-DM04. With special firmware the module behaves as a pulse decoder coming from sonar. The decoder accepts the SRF05 module from Devantech or compatible. There is also the option to connect the sonar with I2C interface via I2C to CAN gateway.

4) IR beacons: IR beacons were originally designed to facilitate precise robot navigation into the docking station but they can be used for wide variety of other applications. The transmitter consists of a circular IRED array. It transmits coded omnidirectional signal. The beacon has selectable code and signal intensity.

The receiver is also of circular shape with IR photodiodes attached on the perimeter. It can evaluate distance and angle for up to four beacons. The angle is calculated from the ratio of photodiodes currents and distance from intensity. Angular precision is 2-3 degrees and intensity corresponds to logarithm of distance in approximately 32 steps. The readings are reliable in most environments up to 3 meters.

The IR beacon system was tested both indoors and outdoors. It was presented at Eurobot 2009 as a sensor for absolute localisation on the playground and for opponent detection. Outdoor application was demonstrated at Robotour 2008. The set of one transmitter and two receivers facilitated reliable robot colony guidance. The following algorithm was actually simple enough so one of CAN modules was used for the control.

5) Bumpers: Robots often require various bumpers for object and collision detection. There are several options. The simplest one is set of micro-switches connected to digital I/O module. The status message is sent immediately after change (with limited frequency), and regularly once every second. Another option is to use digital sharps. They are contactless sensors with detection range of 5 cm and 10 cm. The output is again digital and is handled the same way as with micro-switches.

6) Other modules: The set of available sensors and actuators is much wider and grows over the years. Among those not mentioned is a thermometer which is useful for gyro offset compensation and as a guard for battery charging. A light gate can be configured together with a servo module for automatic gripper action. Ultra bright LEDs can provide light for a camera in darkness.

1.6 User Interface

Eduro has a simple user interface, primarily used for Eurobot contest. There is a set of color LEDs, a selection switch, an easily accessible emergency stop button, recently an alphanumeric display and a beeper were also added.

The emergency stop feature deserves an extra note. Eurobot contest rules require that in case of an emergency pressing the emergency button disconnects all powered components - typically drive motors - from the power source. In reality, this simple solution would not stop the robot due to inertia. The implemented algorithm first sends stop commands to motors and shortly after that it disconnects the power. This solution at least slows down the robot.

2. Software

Application software can communicate with the base platform on several levels. The most commonly used include high-level standard Player interface and low-level direct access to CAN bus via RS232-CAN bridge. Another option is to leverage the set of Python library modules and functions which can be used for quick prototyping and was successfully used in most of this year's contests (see section with presentations).

2.1 Player

The Player/Stage (P/S) project [4], [5], [6] has been hosted on sourceforge since 2001 and it has become a de-facto standard interface for mobile robotic platforms. P/S is an open source project that originally targeted Active Media robots. However, the current set of supported platforms and devices is much larger, mostly thanks to the open source

nature of its distribution which allows it to be easily extended to new machines.

The Eduro platform started supporting Player 2.1 in 2008 due to the interest from the development team members and collaborators who were familiar with this system from their work on other projects. Even though some of these contributors stopped using this system and moved to proprietary Python code due to problems with binary incompatibility between versions and bugs in even simple tools, we plan to support Player 3.0 on all Eduro platforms.

2.2 Pyromania

While it may seem unwise to build robotic control around a scripting language like Python, we found this approach to be quite appropriate and plan to keep leveraging it for even larger and more complex systems.

The time-critical control routines in Eduro are implemented through dedicated CAN modules. Computationally intensive tasks such like image processing can run in separate threads using Python's binding to OpenCV [7] or, if necessary, in separate programs written in more efficient languages (e.g., the C language). Even in these scenarios, Python remains present in its role of the integration language.

One of the major features, which Player lacked, was simple portability between Windows and Linux operating systems. We developed code for both platforms since limiting ourselves to only one would limit its appeal to potential users.

2.3 Direct control

The lowest level of robot control can be realized via direct access to CAN bus through serial line and RS232-CAN bridge. Programming on this level requires basic knowledge of CAN and CANopen protocols respectively as well as familiarity with detailed specification of incoming and outgoing messages for all modules.

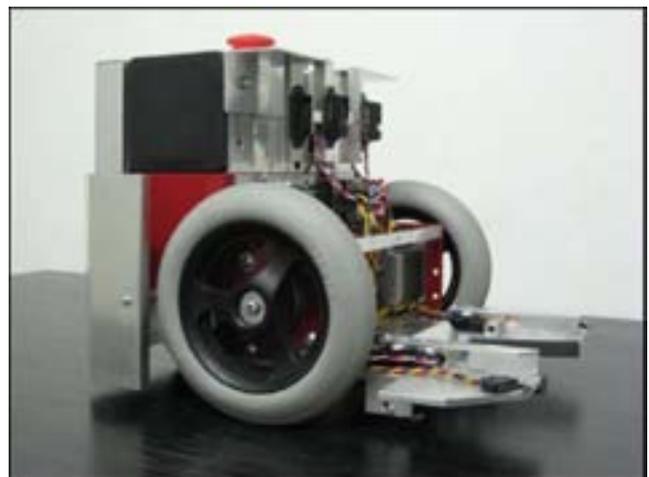


Fig. 2. Eduro prepared for Eurobot 2008 contest

3. Configuration Examples

3.1 Eurobot

Eurobot [8], [9] is an annual international indoor competition for autonomous robots. Robots compete in solving a specific task that differs year to year but generally involves reaching certain goals within an operating space of about 2m x 3m and within 90 seconds time limit.

The Eduro platform participated in three Eurobot events using the same base but varying mechanical attachments designed for that year's specific tasks. In "Mission to Mars"-themed event in 2008, this attachment was an automatically triggered gripper. This gripper was implemented using servos, a lightgate module connected to the CAN network using bumpers, digital Sharp's distance sensors (boundary detection) and analog distance sensors (feeder and opponent detection).

In 2009, the task involved building "temples". That year the attachment was a simple passive plowshare while an IR beacon system was used for opponent detection. The same system was also used for global Monte Carlo Localisation via triangulation.

In "Feed The World"-themed event in 2010, the Eduro platform was equipped with a ball collector in front of the robot. The previously used modules were enhanced with a beeper and an alphanumeric display. The beeper was used to generate acoustic warnings in case of inconsistency between localisation detected by the beacons and the color of the team. The alphanumeric display was used to show the selected strategy.



Fig. 3. Eduro on Robot Challenge 2009 contest

3.2 Robot Challenge/Puck Collect

Videos showing Eduro's participation in Robot Challenge 2009 and 2010 contest [10] in Vienna are available. This event's theme and rules stay the same every year. The Eduro platform fits best in the "Puck Collect" category. In this competition, the goal is to collect red and blue pucks scattered around a white playing field (2.8m x 2.8m) and carry them to the "home base" (colored squares 0.7m x 0.7m located in opposite corners).

Eduro was equipped with a U-shaped passive collector so pucks were collected when the robot moved forward or turned in place. Dropping the pucks was implemented through backup motion. IP security camera with wide fish-eye lens was used for color recognition. Finally, long range Sharp's (1.8 m) were sensing the border of the playground and also facilitated localisation services.



Fig. 4. Eduro Maxi HD on Field Robot Event 2010

3.3 Field Robot Event

The outdoor version of Eduro (Eduro Maxi HD) participated on several outdoor competitions. In Field Robot Event [11] held in 2010 in Brawnschweig, Germany the robot was expected to perform various farming-related tasks in a mature corn field. The robot was equipped with an IP camera, LMS100 laser scanner, a compass, a GPS unit and other modules previously used in indoor competitions (e.g., beeper, display, user panel). A sprayer was connected to Eduro Maxi with a 3pin connector. Two logic outputs independently controlled the spraying operation on the left and right of the robot.

For freestyle part of the event, the robot was equipped with a VTU10 tracking unit on loan from MapFactor [12] which in addition to tracking also facilitates two-way communication via a GPRS modem. The unit was attached to the robot via an USB port through which it accepted remote commands (GPS waypoint where the robot should autonomously navigate).



Fig. 5. Authors and Eduro Maxi HD on RoboOrienteing 2010

3.4 RoboOrienteing

A week after the Field Robot Event the same robot participated in RoboOrienteing event [13] in Rychnov nad Kněžnou, Czech Republic. This contest is similar to better known Robo-Magellan [14]. In both cases, the robots receive GPS coordinates for the starting point, waypoints and the end point and are expected to autonomously travel through the terrain between these points.

For this event the Eduro platform was equipped with tractor tires. Sonar was added in order to achieve better obstacle detection of benches and low placed tree branches.

4. SUMMARY

In this paper we introduced Eduro, a robotic platform designed for education and research. Its modular design was proven successful through high rankings in numerous international competitions - 1st place in Professional Task of Field Robot Event 2010, 2nd place in Puck Collect at Robot Challenge 2009, or 2nd place in Czech Eurobot National Cup 2010. The Eduro platform has attracted enough interest for us to start its serial manufacturing planned for the end of 2010.

REFERENCES

- [1] Cleaning contest, "1st International Contest For Autonomous Cleaning Robots," <http://www.service-robots.org/applications/cleaning.htm>, 2002.
- [2] Daisy, "quarter-finalist of Eurobot 2003," <http://robotika.cz/robots/daisy>, 2003.
- [3] robotika.cz, "Robotour 2006 - robotika.cz outdoor challenge," <http://robotika.cz/competitions/robotour2006/>, 2006.
- [4] Player/Stage, "device interface and simulator," <http://playerstage.sourceforge.net/>, 2001.
- [5] Brian P. Gerkey, Richard T. Vaughan, Andrew Howard, "The Player/Stage Project: Tools for Multi-Robot and Distributed Sensor Systems," Proceedings of the International Conference on Advanced Robotics (ICAR 2003), June 2003.
- [6] Richard T. Vaughan, Brian P. Gerkey, "Really Reusable Robot Code and the Player/Stage Project," Software Engineering for Experimental Robotics, 2006.
- [7] OpenCV, "image processing library," <http://www.sourceforge.net/projects/opencvlibrary>, 1999.

[8] Eurobot, "Contest for amateur robotics," <http://eurobot.org/>, 1997.

[9] David Obdržálek, "Eurobot 2010," Robot Revue, May 2010.

[10] Robot Challenge, "Indoor robotic competitions," <http://robotchallenge.org>, 2010.

[11] Field Robot Event, "Agricultural robotic competition," <http://fieldrobotevent.de>, 2010.

[12] MapFactor, "VTU10 tracking unit," <http://mapfactor.com/>, 2010.

[13] RoboOrienteering, "Outdoor robotic competition," <http://www.vosrk.cz/roboorienteering>, 2010.

[14] RoboMagellan, "Outdoor robotic competition," <http://www.robothon.org/robothon/robo-magellan.php>, 2000.

Martin Dlouhý

Charles University in Prague
Faculty of Mathematics and Physics
Malostranské náměstí 25, 118 00 Praha 1
Czech Republic
E-mail: md@robotika.cz

Jan Roubíček

RobSys Czech Republic
jr@eduro.cz

Tomáš Roubíček

RobSys Czech Republic
tomas.roubicek@robsys.cz

SyRoTek – A Robotic System for Education

Jan Faigl, Jan Chudoba, Karel Košnar, Martin Saska, Miroslav Kulich and Libor Přeučil

Abstract

This paper presents an insight to ideas and the current state of the project *SyRoTek* – *System for a robotic e-learning* that aims to create a platform for students' practical verification of gained knowledge in the fields of Robotics and Artificial Intelligence. A set of real mobile robots is being developed in order to provide remote access to real hardware for enrolled students. The advantage of the real system over a pure virtual simulated environment is in realistic confrontation with noise and uncertainty that is an indivisible part of the real world. In such a system, students can acquire in deep understanding of main studied principles in an attractive form, as students (especially future engineers) like to control real things. On the other side, this can be a potential issue if an accessibility to the system have to be guaranteed in 24/7 mode. In *SyRoTek*, robots are designed with special attention to long-term and heavy duty usage. Moreover, safety mechanisms are realized in several layers of the proposed software architecture that provide access to robot control and sensors. In addition, a support for semi-autonomous evaluation of students' solutions of their assignments is a part of the system.

Keywords: artificial intelligence, robotics, e-learning

Introduction

Computers have been domesticated in the education process during last decades. Simulations of real processes can be easily realized and students can gain better (and faster) understanding of main studied principles. However, the real world tends to be more complicated than a pure virtual environment mainly due to noise and uncertainty. That is why it is important to engage real robots in the education. Even through it is not hard to control a simple robot, the final robot behaviour mostly depends on the real environment. It is known fact that early ideas of Artificial Intelligence clash with complexity and uncertainty of the real world. Therefore, it is very useful to confront algorithms with reality during students labs. Maintenance of real robots that are easily used by students can be very costly, thus so-called virtual laboratories have been investigated and developed by robotic groups. The advantage of these laboratories is that the Internet access allows to control a real robot even from students' homes or dormitories.

Several robotic systems with remote users' access have been realized since nineties, once the Internet becomes available. Early systems allow control of hardware devices in the tele-operating manner [1, 2, 3, 4, 5]. One of the first integrated robotic system for e-learning is the project ARL Netrolab [6], started at University of Reading in 1993 [7]. The used mobile robotic platform consists of robotic manipulator, sonars, infrared range finders and a set of cameras. Netrolab provides access to the robot control and sensors. The measured sensor data have been stored for further analysis. The follow-up project allows control a small rover in an environment simulating a surface of Mars [8]. Probably the most complex system have been developed in the project RobOnWeb [9] at Swiss Federal Institute of Technology in Lausanne (EPFL) [10]. Five fundamental services of web interface have been defined: chat, video, robot control, virtual robot representation, and logging. In

the project REAL [11], four frames are used to provide a remote access to an autonomous mobile robot. The first frame provides the basic access to the laboratory and reservation system. The second frame realizes a tele-operated access to the robot. The additional frame enables possibility to use user's navigation module (written in C programming language) to control the robot. During the autonomous robot navigation, sensor data are collected by user's module and stored in the dedicated user space for further processing. The last frame represents module of a distance learning. A combination of a simulated environment with reality has been applied in the project LearnNet [12, 13]. The VRML technology has been used to model the real environment at the user side, while only coordinates of objects are transmitted over the Internet. This technique avoid necessity to transmit large video files of a real environment, thus it is suitable for low-bandwidth networks. A set of robots has been accessible for users in the project Virtualab [14]. Several cameras monitored a play-field and a user can use a combination of several views to get better overview of the robots movements. The robots can be controlled remotely via the ActiveX technology or by a program in C++, Delphi or Java programming language. An open source solution based on the Player/Stage framework [15, 16] has been planned in another project of a virtual robotic laboratory [17], which unfortunately seems to be no longer active.

The aforementioned projects are only a small selected set of representative projects that deal with the remote access to real hardware devices. Lot of other projects can be found, However, the main concepts are pretty much similar and have been proposed in the aforementioned approaches. The main differences can be found in the used technologies that are improved over time and in a combination of several concepts in order to find the most suitable solution for particular requirements. Also new systems provide additional features that came from new technologies and progress in forms of e-learning education process.

SyRoTek – System for a robotic e-learning is one of the current systems, which is similar to other projects. It shares many ideas of the previous systems, but it has been designed with different aspects that provide additional features over the previous systems. In this paper, we describe the main ideas and concepts of the system, which enable contingency to use the system in regular students labs related to Robotics and Artificial Intelligence as a field to verify learned theories and to gain practical experience with real robots.

The paper is organized as follows. The basic overview of the system and its architecture is described in Section II. Description of the designed robotic platforms and developed hardware parts is presented in Section III. Access to the system from user's point of view is described in Section IV. The essence of the SyRoTek e-learning part can be considered in a concept of assignments, which is described in Section V. Finally, remarks and the current progress status are presented in the conclusion.

1. System Overview

SyRoTek consists of an arena with real autonomous mobile platforms, communication infrastructure and the main control computer accessible from the Internet. The overview of the system is shown in Fig. 1.

Robots are placed inside an arena with dimensions of 3.5 x 3.8 m including docking stations with a robot battery charging system. Several cameras support visualization of the real scene and creation of video records that are provided by a video server. Estimation of robots positions is crucial in various robotic navigation tasks, also it is useful for evaluation of user's assignments, thus a localization module based on processing of an image from the camera placed above the arena has been developed. The main control computer provide access for users from their workstations to SyRoTek through the Internet.

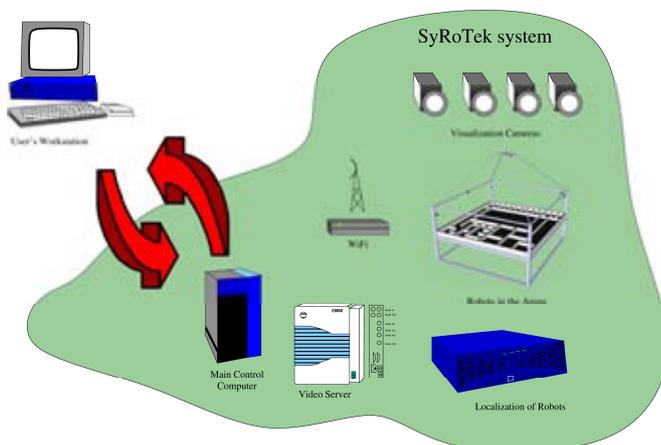


Fig. 1: SyRoTek system overview

The architecture of SyRoTek consists of three main layers: the low-level hardware layer, core layer, and user interfaces, see Fig. 2. The hardware layer is a set of firmwares for micro-controllers and drivers for specialized devices (e.g. laser rangefinder, camera) that are used to collect data from sensors, to control the robot, and to watch the power system of the robot.

The core layer provides basic functionalities of the system and consists of several modules. The system module ensures safety and accessibility of robots from other parts of the system. The task module represents a set of supporting objects for tasks, e.g. realization of dynamic changes in the environment, tasks evaluation. The user module serves as

the main access point to the system for regular users. It realizes an interface between SyRoTek-core and selected end user communication protocol through which a user controls a robot and reads sensors data.

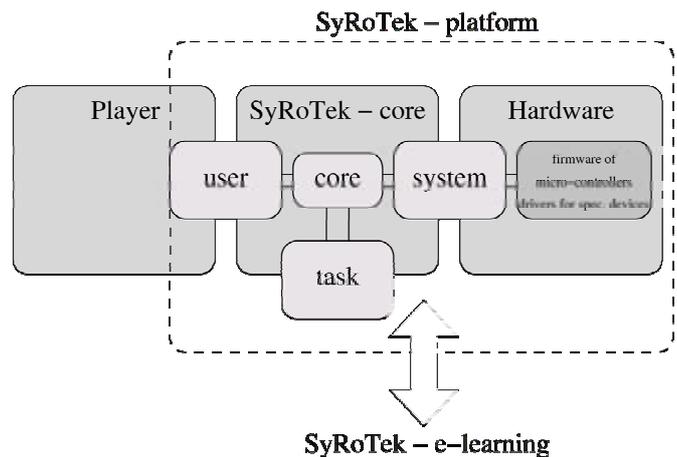


Fig. 2: SyRoTek architecture overview

The layers represent the so-called SyRoTek-platform that is hardware components and necessary software, which provides independent access to the components. The end user of SyRoTek will not be in direct connection with the SyRoTek-platform internal interfaces. Instead, another interfaces are provided. This abscission is realized due to the following reasons. At first, it allows selection of already known and used (by robotic community) abstractions and interfaces to hardware devices, in our particular case the Player [16] framework has been selected. Moreover the hardware part of SyRoTek is considered to be used in longer horizon than currently selected technologies for the current web based remote access to the e-learning part of the system. Thus, the separation of the SyRoTek-core from the presentation layer allows possible further replacement of the web pages by modern technologies, e.g. using visual impressive presentation based on new HTML5, CSS3 features, new toolkits like silverlight [18] or another Adobe Flash technology replacements.

The whole system is implemented as a set of services that provide access to particular functionalities of the system: robot and hardware parts, web pages, visualization and development tools. Besides, a set of maintenance tools and services are part of the system. The set comprises monitoring and notifications of status changes, power management, shutdown policies and emergency actions, like self-docking in the case of a low power. All these services are designed to improved reliability of the whole system and possibly avoid system damage by an improper usage.

2. Hardware Description

The hardware components of SyRoTek consists mainly of a closed play-field called arena and a set of mobile robotic platforms. All obstacles are removable and a part of them can be controlled remotely. The robots have been designed for a long-term and heavy duty usage. The arena is placed in a university computer lab, see Fig. 3. Although the system is designed for a remote access, students can directly see the robots.

A schema of the robot is depicted in Fig. 4. The robot is called S1R and its body consists of the main chassis and an optional front module. The robot has differential drive realized by two Faulhaber 2224 motors with a gearbox

(20/86:1) and the magnetic encoders IE2-512. The on-board power is provided by six Li-Pol Kokam 2400 mA-EHD-30C

A dedicated MCU is used to wrap particular interface to be sensor bus compatible. Even though this unification requires

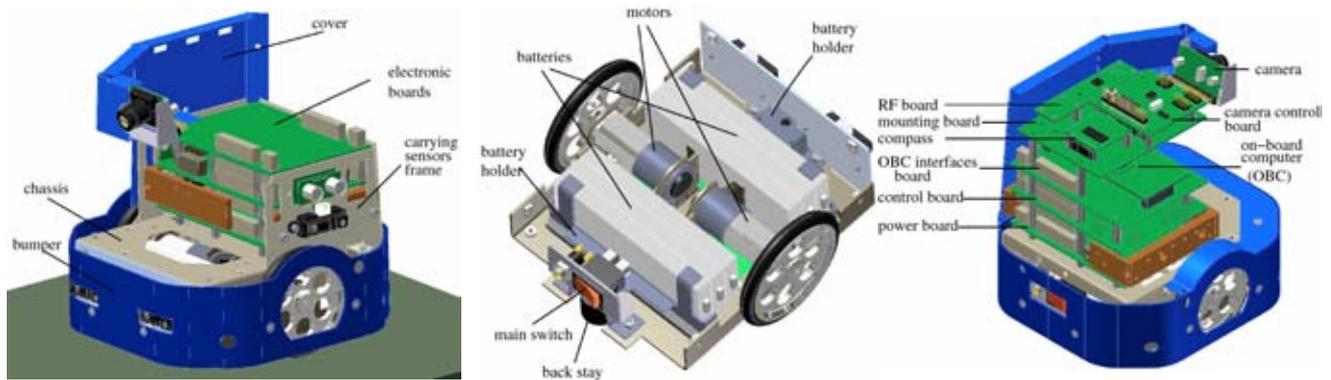


Fig. 4: Schema of the SyRoTek robotic platform - S1R

cells with nominal voltage 3.6 V connected in serial¹, thus the real voltage is in the range from 18.0 V to 24.6 V. The *power board* provides the main on-board voltage 5 V using the power regulator LM2596 - 5V/2A and the Atmel ATmega 2560 Micro-Controller Unit (MCU). A battery charger based on LTC4008 is integrated to the *power board*. The motors are controlled by the *control MCU* (cMCU) that is Hitachi H8S/2639 operating at 20 MHz placed on the *control board*, the maximal velocity of the robot is designed to be around 0.35 m/s. The on-board computer (OBC) is the Gumstix Overo Fire module with ARM Cortex-A8 OMAP3530 processor unit operating at 600 MHz and running the Linux kernel in version 2.6.x. The so-called *sensor bus* based on the I²C bus is used to connect the *power board* and additional sensors to the OBC while cMCU is directly connected to OBC via dedicated asynchronous serial interface. A dedicated MCU called *bridge* is used for interfacing sensor bus to SPI of OBC. In order to guarantee data packet delivery time from the control computer to OBC a dedicated RF module is planned to be used, probably based on Nordic nRF24L01. Besides, WiFi can be used to transmit a large amount of data.

additional MCU, it is advantageous from the software point of view. A unified communication mechanism can be used with various devices, and to transmit data from sensors to OBC and the main control computer, see schema of the communication between sensors and users in Fig. 5.

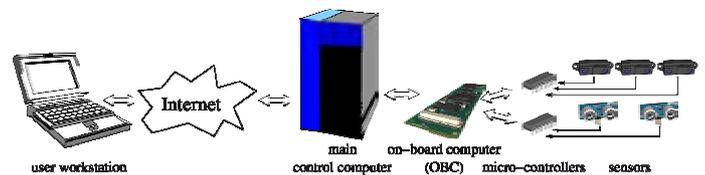


Fig. 5: A schema of a communication between sensors and users

Additional sensors, e.g. the front sensor module can be connected to the sensor bus, or directly to the OBC. Nowadays, two types of the front sensor module are available, see Fig 6. The first one is equipped with three sonars (Devantech SRF10) and three infrared range sensors (Sharp GP2Y0A21Y), the module is connected to the sensor bus. The second one uses the laser range finder Hokuyo URG-04LX and it is connected to OBC via the USB interface.



Fig. 3: SyRoTek arena

The chassis serves as carrier of basic sensors of the surrounding environments: five infrared range finders (Sharp GP2D120), three sonars (Devantech SRF10), floor sensors (twelve infrared sensors) and the intelligent camera module CmuCam3 [19]. The range sensors are directly connected to cMCU, while other sensors are connected to the sensor bus. Sensors of the robot internal states including the compass (Philips KMZ51) and the encoders are connected to cMCU. Besides, temperatures are measured in various places of the robot body, and currents to the motors are measured as well in order to provide the so-called software bumpers.



Fig. 6: Two types of front sensor module

¹ Based on real experiments, the battery pack provides energy for around eight hours of a continuous robot moving without additional power saving techniques.



Figure 7: Three S1R robots during exploration

Three robots S1R during the exploration task are shown in Fig. 7. Notice the patterns on top of the robots that are used by the localization system to estimate the current positions of the robots.

3. User Access

Three types of user access can be found in SyRoTek: web, remote shell, and data (video streams and sensors data). The web access can be considered as a primary gate to the system. It provides basic description of the whole system, account creation request, reservation system, maintenance of a user profile, courses and particular assignments. A more detail description of this part of SyRoTek is dedicated to Section 4. In this section, the next types of accesses are described.

SyRoTek is focused on an e-learning in robotics, particularly it aims to provide support of knowledge transfer of foundations of several robotic problems and also practical verification in various robotic tasks. It means that a student can use real robots to verify the learned principles in a real practical application. So, the student is requested to create a program that is able to navigate a real robot in an environment.

The practical orientation of the robotics steers SyRoTek to provide support of software development process oriented to robotics. The best practice in robotic development is an initial creation of an algorithm or a control program that is verified against simulation, which is typically much faster process than with a real hardware. Moreover, a program that is able to navigate a mobile robot is often consisted from various components, and the complete program can be quite complex. Thus, it is advantageous if a student can use already available components. Also a good hardware layer abstraction is a plus in order to create a simple program that can be easily transferred from a simulation to real robots. These considerations are the main reasons why the Player/Stage framework [16] has been selected as the main SyRoTek user interface. The Player has a hardware abstraction based on a set of interfaces and devices that are proven by more than ten years of history by several robotic researchers around the world. The Player can be accompanied by simulators Stage or Gazebo. The Player follows a client/server concept in which the user application is a client that is connected to the server (player) via TCP connection. The server provides interfaces representing particular devices, which can be real devices or simulated ones. So, the system can be used in various configurations, e.g. a server running at user's workstation or at a robot, which is remotely accessible.

3.1 Robot Access Module (robacem)

Even though the Player is flexible enough to be used in a robotic application, it does not provide required functionalities of SyRoTek. The main issue arises when an authorization to particular sensors have to be granted, e.g. if an evaluation or monitoring of user's application performance have to be realized. The authorization is not a part of the Player at all. When user's application is connected to the Player server, only one program is able to actively control the robot (its motors) by a dedicated serial interface, e.g. RS232. In such a case, the robot will be inaccessible for system services, which is not desirable. In addition, a user can accidentally send a command that can navigate a robot into forbidden areas. Such a situation cannot be handled in low levels firmwares, because robot surrounding environment have to be taken into account, so a high level action monitor is required. From the other point of view, an evaluation can be based on different sensors, e.g. a robot position from the global localization systems, that can be abused by a user to quickly solve the given assignments. Therefore, to authorize access and to guarantee accessibility to the robot for authorities (like monitoring and maintenance services) an additional component called ROBot AcCEess Module (robacem) is used in SyRoTek-platform.

Robacem represents a robot at a particular computer. The S1R robot uses OBC that is connected with the main control computer via WiFi or dedicated low-bandwidth radio channel with guaranteed transport delays. Therefore two robacem modules are running for each robot in SyRoTek: at OBC and at the main control computer. Robacem allows simultaneous and independent access of system monitoring services and Player servers, which are accessible from user applications. A basic schema with possible places where users' applications can be executed is shown in Fig 8. The connection between user's application running at the main computer and the player server at OBC (represented by the red arrow) is possible. However, it can be used only with special attention. During preliminary experiments, a client application connected to the player is able to generate very intensive traffic, which significantly reduce the response of WiFi connections to other robots. Therefore, such a connection can be used only if additional bandwidth limits are involved, e.g. restriction of a connection bandwidth. Otherwise a user can cause degradation of system functionality.

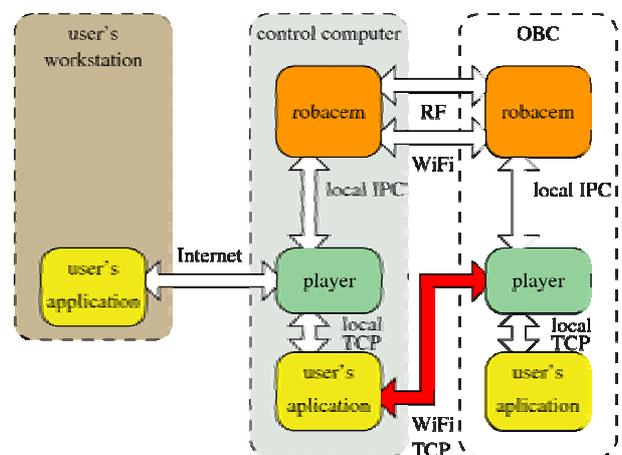


Fig. 8: Connections of process with robacem modules

3.2 User's Remote Access

Student's program to control mobile robots requires necessary software development tools that have to be installed at a user's workstation, which can be tedious. Therefore a remote access to the main control computer, which is fully configured, is allowed. A user can use secure shell (ssh) or secured graphical access by ssh tunneling of XDMCP. These protocols are easy to use within standard installation of Linux based distributions or other unix based systems and they do not require additional proprietary software. Moreover, a remote process execution can be configured in such a way that a user does not recognize a difference between local and remote execution at a glance.

The remote shell access is advantageous in a situation when user's program requires low transport delays, which cannot be guaranteed in a case of a low bandwidth Internet connection. The shell does not have high requirements, and the user is able to execute or even develop her program remotely with slow connections.

3.3 Data Access and Visualization

The best way how to access to the robot is a connection of user's application to the player server running at the main control computer. Our pilot experiments indicate that a connection with 512 kbit/s bandwidth provides sufficient comfort, if video streams are not required.

Video transmission requires an additional bandwidth that is why it is considered as an independent communication channel. In a robotic application, data from real sensors are processed in order to generate the most suitable action. It is very useful if data are visualized and combined with a real view of the scene. We consider the Stage simulator (in version 3.x) as a base of our visualization systems. The simulator provides models of sensors with particular visualization, therefore we enhance it by consideration of several views that can be combined with videos of the real scene. An example of such a visualization is shown in Fig. 9.

A user can use our modified stage simulator as a visualization of the real situation in the arena. According to her Internet connection, she can select particular video streams from several cameras mounted in the SyRoTek arena and various quality (resolution and bandwidth) of videos. Moreover, videos can be recorded during user's application execution and together with recorded data they can be used for debugging or as a proof of program functionality in the assignment evaluation process.

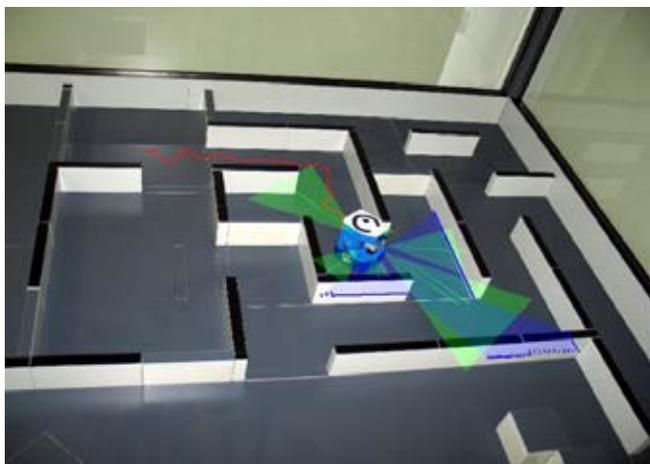


Fig. 9: Visualization of the arena and real sensor data

4. Assignments

SyRoTek as an e-learning system is considered to be practically (task) oriented, due to its relation to real robots. The studied principles of the related domains can be demonstrated in reality by moving a robot in the arena. From this perspective, the essence of SyRoTek lies in robotic tasks. Besides, the supplementary materials can be presented to students in standard ways, e.g. in a form of web pages.

In our first ideas and concepts (based on the previous and current virtual laboratories) we have planned to use one of the already available web based e-learning systems, particularly Moodle [20] has been considered as the most suitable candidate. Later, we recognized that a practical part of assignments (robotic tasks) is tightly related to the software development process of an application to control real mobile robots, which is not a part of general systems for Content Management System (CMS), or Learning Management System (LMS). Such systems can be customized, but most of the specific functionalities of SyRoTek have to be implemented from scratch, which can be more costly (due to general system API) than a creation of a simple specific (single-use) system. Based on this premise, we have reconsidered necessity of a general CMS and instead of primary usage of such a system we use direct description of tasks according to the SCORM 2004 definition [21]. Specific information related to the robotics, resp. SyRoTek, are stored in the Learning Object Metadata (LOM), therefore it can be eventually used in any system that supports SCORM 2004. A relation database has been selected to store the tasks definitions. Its main advantage is relatively cheap creation of copies of assignments and fast access to the definitions that are crucial properties of the desired feature of SyRoTek that is an individualization of assignments.

E-learning systems are sometimes denoted as impersonal. In SyRoTek, we use current technologies to create a support for more personal relation between a teacher and his course students. An individualization of a particular task for each student enables capability to reflect current knowledge of the student and his focus to the most relevant parts of the problem. Such an individualization needs a set of supporting modules that substitutes particular sub-tasks of the assignment and are helpful to quick and targeted knowledge transfer to the student. Initial versions of these modules are part of the system, but further student's implementation of particular assignments can be used in future.

4.1 Courses and Tasks Concepts

Courses can be divided into three categories in SyRoTek: introductory, intermediate, and advanced. The first category are courses to afford fundamental algorithms in key robotic domains like simple robot control, reactive behaviours, dead-reckoning, sensor processing and path&motion planning. In these courses, students are also introduced to the provided SyRoTek functionalities. The intermediate courses are based on Top Assignments (TA) that comprise from several fundamental problems. These courses are organized to guide students to acquire knowledge of necessary fundamental algorithms in order to solve TA of the course. The advanced courses are similar to the intermediate courses. The difference is that the advanced courses aim to solve the selected TA itself.

Two groups of TAs can be defined: basic and advanced. The basic TAs are typical problems in robotics and artificial intelligence, which are well studied or well described, e.g. simultaneous localization and mapping, inspection,

exploration, coverage, pick&delivery. The advanced TAs are hard problems, for which it is expected that students will either study literature to find some approximate solution or they will creatively develop its own approach. These problems are typically designed as multi-robot tasks where cooperation and coordination of robots play an important role, e.g. games like pursuit-evasion, capture the flag or treasure hunt.

4.2 Task Evaluation

From the e-learning point of view, teacher's access to SyRoTek is also important. The system allows specification of constraints under which a task can be solved by the particular students. The system supports verification of the task in semi-autonomous manner. A teacher can write a module that is simultaneously executed with student's program within a dedicated period for submission. Such a module monitors behaviour of student's program to control the robot or it can dynamically change environment according to the robot behaviour, e.g. an evader controlled by student's program can be pursued by a different program in pursuit-evasion scenarios. An output of the student program can be automatically processed to verify student's results. A performance of the robot behaviour is captured and video is created for the teacher to support evaluation of student's solution.

Conclusions

The SyRoTek project is in the second half period of solution, therefore this paper presents only the main ideas, concepts and preliminary results. First robots have been created and concepts of the user access to robot functionalities have been verified in selected robotic tasks. These experiments support the main ideas of the proposed concepts, however it also show possible communication issue related to limited bandwidth of the used WiFi infrastructure. The issue can be solved by additional restrictions of the direct users access to a robot in order to guarantee desired quality of accessibility for other users. Thus, it is not a drawback, as it will improve the overall reliability of the system.

The further development will concern to finalization of robots hardware, creation of an initial public access to system, and preparation of supplementary materials. It is expected that SyRoTek will be open in trial application for users at the end of the year 2010.

Acknowledgements

The work presented in this paper has been supported by the Ministry of Education of the Czech Republic under program "National research program II" by the project 2C06005.

References

- [1] "<http://www.telegarden.org>," 1997, (accessed 27 July, 2010).
- [2] "<http://www.telescope.org>," (accessed 27 July, 2010).
- [3] "<http://www.iai.uni-bonn.de/rhino/tourguide>," 1997, (accessed 27 July, 2010).
- [4] "<http://www.cs.cmu.edu/afs/cs.cmu.edu/Web/People/Xavier>," 2001, (accessed 27 July, 2010).
- [5] "<http://robotoy.elec.uow.edu.au>," (accessed 17 July, 2006).

[6] "<http://www.arl.rdg.ac.uk/research/netrolab>," 2003, (accessed 27 July, 2010).

[7] G. McKee and R. Barson, "Netrolab: a networked laboratory for robotics education," IEE Colloquium on Robotics and Education, April 1995.

[8] "<http://www.redrover.reading.ac.uk/RedRover/index.html>," (accessed 26 September 2006).

[9] "<http://asl.epfl.ch/research/projects/RobOnWeb/robOnWeb.php>," 2002, (accessed 27 July 2010).

[10] R. Siegwart and P. Sauc, "Interacting mobile robots on the web," in Proceedings of the 1999 IEEE International Conference on Robotics and Automation, May 1999.

[11] E. Guimarães, A. Maffei, J. Pereira, and et. al, "Real: A virtual laboratory for mobile robot experiments," IEEE Transaction on Education, vol. 46, no. 1, February 2003.

[12] "<http://www.learnnet.de>," (accessed 27 July 2010).

[13] I. Mas'ar, A. Bischoff, and M. Gerke, "Remote experimentation in distance education for control engineers," in Proceedings of Virtual University 2004, Bratislava, Slovakia, December 2004.

[14] "<http://www.robotika.sk/projects/virtuallab>," 2008, (accessed 27 July 2010).

[15] "<http://playerstage.sf.net>," (accessed 27 July 2010).

[16] B. P. Gerkey, R. T. Vaughan, and A. Howard, "The player/stage project: Tools for multi-robot and distributed sensor systems," in In Proceedings of the 11th International Conference on Advanced Robotics, 2003, pp. 317–323.

[17] "<http://vlab.pjwstk.edu.pl>," (accessed 26 September 2006).

[18] "<http://www.silverlight.net>," (accessed 27 July 2010).

[19] "<http://www.cmucam.org>," (accessed 27 July 2010).

[20] "<http://www.moodle.org>," (accessed 27 July 2010).

[21] "<http://www.adlnet.gov/Technologies/scorm/SCORMS Documents/2004dition/Overview.aspx>," (accessed 27 July 2010).

Ing. Jan Faigl, Ing. Karel Košnar, Ing. Martin Saska, RNDr. Miroslav Kulich Ph.D. and Ing. Libor Přeučil CSC

Czech Technical University in Prague
Faculty of Electrical Engineering
Department of Cybernetics
Technická 2
166 27 Prague 6
{xfaigl,kosnar,saska, kulich, preucil}@labe.felk.cvut.cz

Ing. Jan Chudoba

Czech Technical University in Prague
Faculty of Electrical Engineering
Center of Applied Cybernetics
Karlovo náměstí 13
121 35 Prague 2
chudoba@labe.felk.cvut.cz

Mobile Robotics Education at FEE CTU in Prague

Jan Faigl, Tomáš Krajník, Karel Košnar, Hana Szücssová, Jan Chudoba, Vladimír Grimmer, Libor Přeučil

Abstract

In this paper, we describe concepts and main ideas of the labs of the Mobile Robotics course at Faculty of Electrical Engineering, Czech Technical University in Prague. Besides, we present our gained experience from three years of teaching of the course. We consider students' contact with real hardware and real sensor data as the most important part of mobile robotics as the mobile robot can quickly lose information about its position in contrast to stationary robotic manipulators. Thus, the autonomous navigation is a crucial problem. Moreover, a computer simulation cannot substitute complexity of reality, such as noise, imperfect measurements and random events. To achieve our desired pedagogical goals we have decided to develop a new small platform that will be based mostly on off-the-shelf components and it will have sufficient computation power to use the Player robotic framework. The labs are organized into four consecutive assignments with increasing complexity and a final assignment that combines particular students' results from the previous tasks. The final assignment is to develop an algorithm that navigates the mobile robot in order to create a topological map of the environment and reuse this map for later autonomous navigation.

Keywords: robotics, e-learning

Introduction

The course "Mobile Robotics" is an optional subject of the Technical Cybernetics study program at Faculty of Electrical Engineering (FEE), Czech Technical University in Prague (CTU). The course assumes a small student group because of the necessity of individual and personal contact between students and teachers. When the course has been opened at summer semester in the year 2008, it had less than twenty enrolled students. In this paper, we describe the main concept of the course labs, selected solutions and gained experience from three years of the course.

The course is taught within fourteen weeks of the semester and is organized into the same number of lectures and labs. The lectures are dedicated to theoretical description of basic principles of navigation of autonomous mobile robots. These lectures cover relatively wide range of topics from motion control, sensor data processing and path planning, to environment modeling, localization, mapping and simultaneous localization and mapping (SLAM) techniques [1]. It is clear that only the most important ideas and concepts of mobile robotics can be presented within the limited time of the course. Therefore, the core of the transferred knowledge is in understanding of fundamental principles and issues of the mobile robot navigation in a real environment.

The principles discussed at the lectures are practiced during the labs assignments with real robots. As a part of our lab concept, we had to select the most suitable robotic platform for our desired pedagogical goals. The desired platform had to be affordable, allow a reconfiguration of sensors, provide enough processing power and be robust enough to be used by inexperienced students.

Based on our previous experience in the field of experimental research of navigational algorithms for autonomous mobile robots, we have chosen to create a

small platform, even that similar platforms (in the sense of robot dimensions) have been available in the market. To allow reproduction and reuse of our platform by others, we have decided to use off-the-shelf components, integrated the platform in the Player [2] system and published documentation, construction plans and software on our web pages [3].

The main reason for our choice was based on the fact that marketed solutions are too simple and insubstantial, e.g. LEGO Mindstorm or Fischertechnik ROBO Mobile Set, do not provide enough processing power or cannot be extended by advanced sensors, e.g. Rogue Blue ERS, Arrick Arobot Mobile Robot, Carper Rover OOPic-R Combo, Rogue ATR Base, Kit, Lynxmotion 4WD1, Inex Interactive C Robot Kit V2.0, AIRAT 2, Surveyor SRV-1, Hemisson, Khepera, or are too expensive, e.g. Pioneer 3-DX, Koala.

The main idea of the labs is to properly setup the robot sensor system and to develop an application that is able to control a mobile robot in order to create a map of a real environment. At first, students are introduced to the field of mobile robotics in four consecutive assignments with increasing complexity. In these assignments, students adopt methodology to create an application controlling a real mobile robot. After that, the students are given a general description of the final task, which is exploration of unknown environment. In this task, the robot should plan its actions in order to create a complete map of the surrounding environment. Such map should provide enough information for effective path planning to destinations given by a human operator. Specification of the final task is general, in fact, it is impossible to fulfill such a general task within the course labs. Therefore, the students are requested to specify restrictions and conditions in which their robot will be able to fulfill the exploration task.

The rest of this paper is organized as follows. The developed mobile platform called MORBot is described in

Section 1. The developing environment and concepts of students' work on their applications are presented in Section 2. The assignments, their main purposes and challenges for students are described in Section 3. Remarks and ideas for further course improvements based on gained experience from the three runs of the course are presented in the conclusion.

1. MORBot Platform Description

The aforementioned requirements lead to design a new robot, which was inspired by the research platform GBot [4] and robots for the Eurobot competition developed in the Gerstner Laboratory [5, 6]. The robot we needed for the course had to be smaller than these platforms, because of the space restrictions (the course takes place in an ordinary computer lab). Moreover, it had to be simpler and easier to use. Due to the target application of the platform, a motion on a flat surface had been considered as sufficient.

The design has been constrained by our preliminary budget. The required cost of particular components for a single robot has been targeted to be less than one thousand euros. The main idea, concepts and preliminary components have been suggested by authorities, however the final robot design and construction has been realized by students in two Bachelor's [7, 8] and one Master's thesis [9].

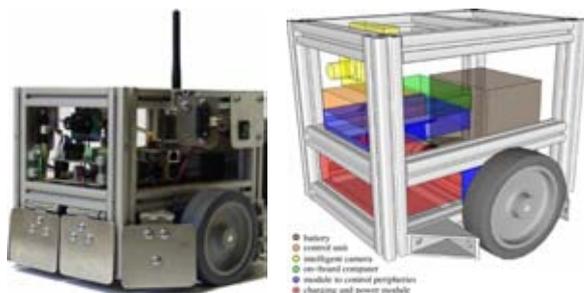


Fig. 1 The MORBot platform and its main components

The robot hardware consists of an aluminum chassis, power, motion, sensor and control subsystems, on-board computer and an intelligent camera, see Fig. 1. The hardware subsystems are interconnected by several buses, see Fig. 2. Its software composes of micro-controller units (MCU) firmwares and the Player server from the Player/Stage framework [2] providing hardware abstraction for users.

1.1 Robot hardware

The skeleton of the robot is composed of interlocked X-shaped aluminum beams (Item profiles). These are firm enough to support robot devices and provide reliable shock protection. The skeleton dimensions are 16.0x22.0x18.5 cm and the robot circumference is about 85 cm. The total weight of the robot (including battery and sensors) is about 5 kg.

The power subsystem is composed of a 12 V, 5 Ah sealed lead-acid battery, a charging control board and a voltage stabilizer, which provides 5 V for additional electronic boards.

The motion subsystem is based on the differential drive Devantech RD01 and two supporting rollers mounted at the back side of the robot, the maximal forward velocity is about 0.8 m/s. The motor driver MD23 controls speeds of both wheels and counts pulses from the motor IRC sensors. The counter values are sent to the control board via the I²C bus.

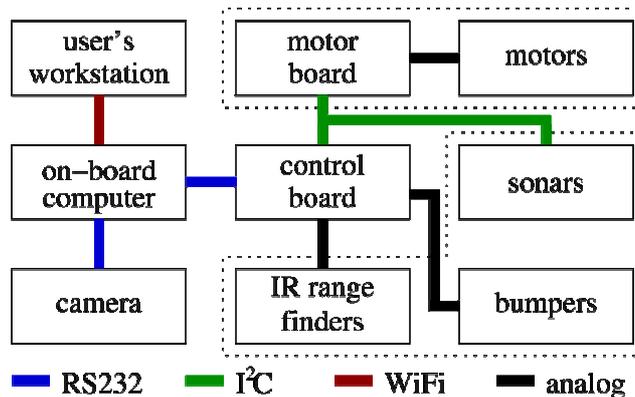


Fig. 2 Robot subsystems scheme

The sensor subsystem consists from these sensors: four Sharp GP2D120 (IR) rangefinders, two Devantech SRF10 sonars and seven mechanical bumpers. The IR rangefinders have detection range between 0.04 and 0.30 m and the sonars detection range is 0.1-4.0 m. The IR rangefinders provide an analog voltage signal needing to be further processed and the sonars SRF10 supply the measured distance in centimeters via the I²C bus. Positions of the sonars and the IR sensors are not fixed and students can reposition them according to their intentions and the particular assignment. The mechanical bumpers are based on microswitches covered by metal plates, see Fig. 3.

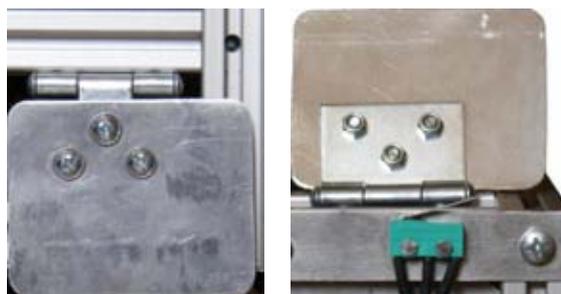


Fig. 3 The mechanical bumpers

The control subsystem works as an interface of the motion and sensor subsystems to the on-board computer. The control board uses the Atmel ATmega 168 MCU to continuously gather data from the sensors and the motor control board. The MCU estimates the robot position from values of the motor board IRC counters. The current status of the sensors and the estimated position are provided to the on-board computer via the RS232 interface on request. Moreover, the on-board computer issues commands, which set speeds of the motors. Control subsystem ensures safety of the robot independently on the on-board computer. When a frontal bumper is pressed, the control unit prohibits forward movement and vice versa, thus preventing damage to the environment and to the robot.

The primary purpose of the on-board computer is to run the Player server, which interfaces the robot devices and sensors. Moreover, the computer allows using more advanced sensors like cameras or laser rangefinders. The computer is based on the Gumstix Verdex XL6P [10] motherboard with two expansion boards: the netwifimicroSD and interface board called PortBoard [9]. The motherboard utilizes the Marvell PXA270 processor running at 600 MHz, 128 MB RAM and 32 MB of internal FLASH RAM with installed Linux operating system in version 2.6.x. The expansion boards provide a micro SD card slot, Ethernet and WiFi interfaces, three serial ports and I²C and USB buses.

An intelligent camera, the CMUcam3 [11], is installed on top of the robot and is connected to the on-board computer via

the RS232 interface. The camera itself is capable of recognizing and tracking objects with distinctive colors [12]. The on-board computer specifies a color of the searched object and the camera starts to send object position in its image coordinates.

The fact that the robot is not completely covered is appreciated by students, because they have a good overview of the robot inner structure. Moreover, they can change placement and configuration of particular sensors and realize that sensor configuration must be reflected in software controlling the robot.

2.2 Robot software

The software of the robot is divided into three layers. The control software (firmware), which runs on the control board MCU, gathers sensory data, estimates the robot position and provides an interface to the robot motors. The image analysis software running on the CMUCam3 can also be considered as a part of the firmware. The second layer provides abstraction of hardware devices and it is realized by the Player server running on the on-board computer. The last layer is students' application itself, which is a client application to the Player server.

One of the advantage of the Player system is the support of various devices like the CMUCam3, IR rangefinders, sonars, motors and odometric systems. However, due to our own design of the control board, we had to develop our own driver for it [8]. The Player system offers an easy-to-understand tutorial on driver development, thus implementation of a new driver did not take a long time. A student, who wants to use the robot, can access its on-board computer remotely via a WiFi connection. The student program connects to the Player server running on the robot can retrieve sensor values and set robot angular and forward velocities.

Documentation needed to build the robot including component list, electronics board schemes and software is accessible through the web pages [3].

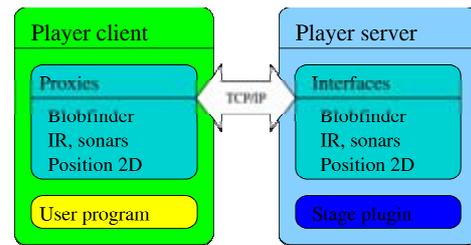
2. Developing Environment of Students' Assignments

The practical part of the labs is based on the Player/Stage framework [2], which uses the client/server architecture. The Player/Stage framework is widely used, well documented, free and open-source [13]. The basic principle of the Player/Stage framework is shown in Fig. 4.

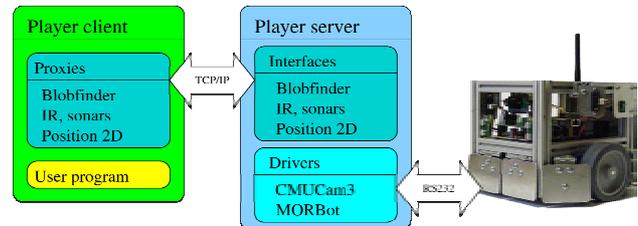
The Player server provides a unified networked interfaces to robotic sensors and actuators. Programs, which control the robot, connect to the Player server as network clients. Therefore, the robot control programs can be written in any programming language and can run on any computer with a TCP/IP connection to the robot. The Stage is a simulator plugin to the Player server, thus a developer without a real robot can use it to substitute the real hardware and environment by simulation. It is another great advantage of the Player system, because it provides straightforward deployment of the program verified in the simulator to the real robot.

These properties allow students to work at home, thus prior to the school labs they can verify their programs. Consequently students can spend more of their time at the lab by consultations of found issues with a teacher and practical verification with a real mobile robot. The students can either download and install the Player/Stage on their computers or use the standard unix-based graphical remote

access protocol to run the Player/Stage on a university server.



(a) Connection to the Stage simulator



(b) Connection to the MORBot

Fig. 4 A basic schema of the Player/Stage framework

One of the encountered issues is the fact, that not all students are familiar with alternative development tools and concept of the client/server architecture. They tend to prefer the particular development environment adopted by their previous experiences (typically not well mentored). To allow fast introduction to the development tools, we have prepared a skeleton of user's client application. The skeleton composes of several files written in C++, which define classes representing the robot and its devices. Some methods of these classes are empty and are supposed to be implemented by students in the lab assignments. For example, the robot class defines (empty) methods, which correspond to the robot behaviours, e.g. `moveTo(x,y)` or `explore()`. The files with C++ codes are complemented by recipes ("Makefiles"), which prescribe how to build sources and create the executable binary file. Thus, only a minimal set of tools (gcc compiler and gmake build system) is required. After several practical applications, students reported that this framework is comfortable, straightforward and easy to use, especially within a remote session.

This positive feedback is very important for us, due to the fact that students are introduced to the programming language Java in the first years of their study at FEE, CTU. The Player framework is used with the C/C++ interfaces and for some students it is difficult to learn a new programming language. Even that students do not become experts in C/C++, they recognize that for a certain task a new (not already known) tool can be more appropriate. This helps to students to realize that the essence of programming does not lie in mastering one particular language, but rather in knowledge of algorithms and systematic principles.

Another possible issue of the used Player/Stage framework is a requirement of the unix-like operating system to install and use the framework. Even that the recent Player version 3.x supports Windows, unix-based operating systems are advantageous for students, because the development tools are part of standard installation, e.g. the gcc compiler or the gmake build system. However, if a student would refuse to install a unix-like system on her computer, she can install the freely accessible Xming program [14], which provides access to the fully configured university server with all the necessary tools.

Students are organized in teams with three or four members, and the students are encouraged to actively use

a Version Control System (VCS) for the program sources, in particular the Subversion system [15]. The VCS allows not only comfortable and convenient organization of source codes but also provides information on how students work during the semester. It is not surprising that most of the students increase intensity of work at the end of the semester. We use reports of students' activity to emphasize the importance of continuous work, see Fig. 5 for an example. The reports also provide us a valuable feedback on how students deal with the assignments. Moreover, teachers can access and review students' source codes and suggest corrections by adding notes to the source. This absolves us from tedious management of e-mail attachments.

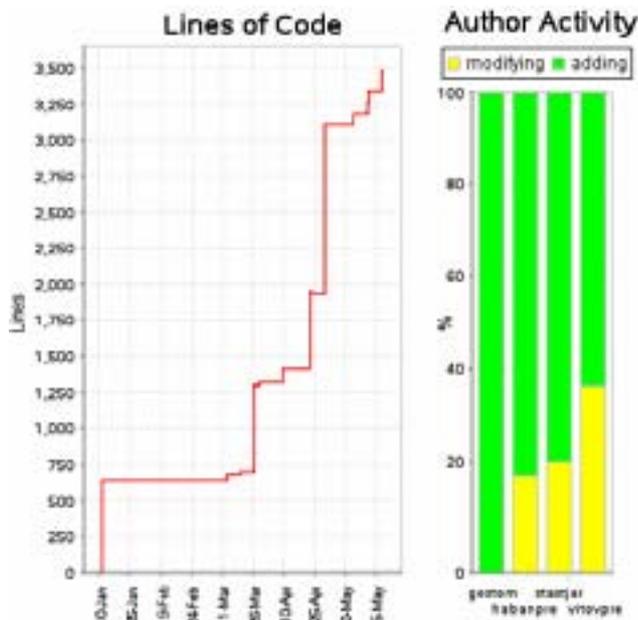


Fig. 5 Example of a students' VCS activity report

The VCS and the provided framework allow easy download and compilation of students projects. Just two commands are needed to bring the up-to-date program version and create an executable file, i.e. `svn update` and `gmake`. This significantly reduces teachers' load when examining students' programs and allows to focus on really important issues.

3. Labs Assignments

The main objective of the labs is to make the students understand the nature of uncertainty in mobile robotics systems. The objective is materialized in assignments to create programs that will provide an intelligent behaviour to the MORBOT platform. The students should learn that uncertainty in measurements and action results can be dealt with by means of feedback loops realized by controllers, which compute angular and forward velocities of the robot from sensor values and gather information about the robot surrounding environment. In addition, the students should learn that a proper decomposition of a mobile robotic task should be carefully chosen to avoid the pitfalls of uncertainty.

The labs consist of fourteen sessions (one per each semester week) that take place in the university computer lab. One session lasts 90 minutes and it consists mainly of contact time with the teachers. Students can also contact the teachers individually during consultation hours. Besides, it is expected that students spend additional time working at home, or at lectures. Students are introduced to the labs and course organization during the first session. The last

session is dedicated to evaluation of the final assignment. The final objective of the assignment is to create a program that will control the platform in order to explore and map an environment and it should be solved within four weeks. Prior to this task, four simpler assignments have to be solved by students, two weeks are devoted for each task. At first, the students have to implement a simple position controller of a mobile robot and use ranging sensors to detect obstacles. In the third assignment, they have to extend the previous solution to a "Bug" type algorithm. The fourth task serves as a basic introduction to image processing, resp. visual navigation. These assignments introduce students to the usage of the Player/Stage framework, robot control, sensor data processing and fundamental robot skills and behaviours. Description of assignments is presented in the following subsections.

3.1 Robot Control

In the first assignment, students create a simple control algorithm that will navigate the mobile robot to a certain position in an environment without obstacles. Their control application has to determine forward and angular velocity of the robot based on the desired position and the current position estimated by the odometry. In this assignment, the students familiarize with the development tools and software framework. Moreover, they learn how to implement a simple controller of a mobile robot and encounter the first problems caused by the uncertainty in robot position. They also realize that a position from the odometry is defined in a local frame of reference and depends on the starting position of the robot.

3.2 Obstacle Detection

The second assignment is an extension of the first one into an environment with an obstacle. The robot has to use its rangefinders to prevent collision with objects in its path. The students have to decide, where to place range-finding sensors and design an algorithm, which processes real sensory data. The students are requested to deal with real sensors and to consider how the sensor output is related to detectable obstacles. Therefore, they have to design filters that deal with sensor noise, non-linear characteristics of the IR sensors [16] and false sonar echoes. A measurement of the output characteristics is necessary to find a more precise transformation of sensor output to the obstacle distance. The assignment is fulfilled if a robot stops before an obstacle and continues its motion once the obstacle is removed.

3.3 Collision Avoidance

In this assignment, the students get familiar with the subsumption architecture. The students are requested to create an implementation of a reactive navigation algorithm of the "Bug" class [17] that will control the robot in order to reach a given location in an environment with several obstacles. The robot combines algorithms from the previous tasks with new behaviour in one program for more complex collision avoidance. In the case that an obstacle is detected, the new behaviour has to actively circumnavigate the detected obstacle and recognize if the desired destination is not reachable. The implemented algorithm does not have to follow the "Bug" algorithm specification exactly. Instead of that, the students are motivated to modify a particular algorithm to satisfy the assignment constraints. Therefore, the students realize that even simple algorithm can be efficient if it is adjusted to the particular real scenario.

3.4 Visual Navigation

The students have to implement a reactive navigation algorithm that will control the robot in order to reach an object of the selected color. The students do not have to implement image analysis procedure, but they use the CMUCam3 [11] camera, which provides image coordinates of objects with the specified color. As before, the assignment can be solved by two feedback loop controllers. One transforms object coordinates in the captured image to the robot angular speed and the second computes the robot forward speed from IR sensors and sonars. Similarly to the previous tasks, the students can use the Stage simulator before deployment of the algorithm to the real robot. An example of the camera, IR and sonar simulation is shown in Fig. 6.

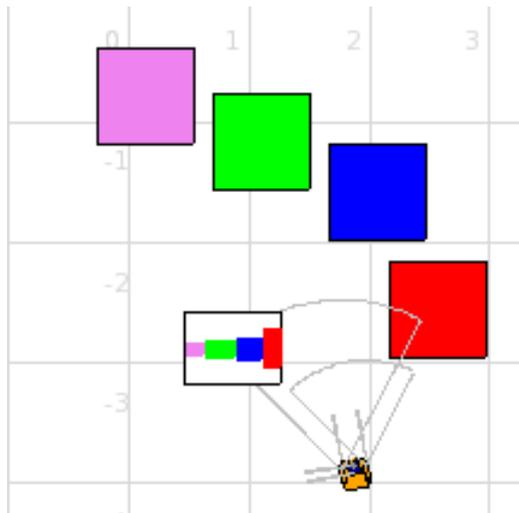


Fig. 6 An example of sensor simulation in the Stage simulator

3.5 The Final Task - Topological Exploration

Finally, the students have to combine and extend algorithms implemented in the previous assignments to make the mobile robot capable of exploration of unknown environment. The robot has to create a topological map of the environment and use this map for planning and reasoning. An operational environment of the robot contains two types of objects - visual landmarks and obstacles. The visual landmarks are boxes with distinguishable colors and the obstacles are white walls or gray boxes. An example of an environment and a created topological map is shown in Fig. 7. A particular related real environment is shown in Fig. 8.

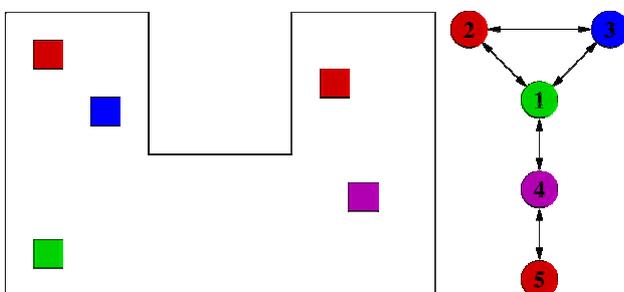


Fig 7 An example of an environment and a created topological map

The time needed to create an exploration algorithm, which will work at any situation, is much longer than the time dedicated to the assignment. Therefore, the students have to specify constraints under which their algorithm will be able to successfully finish the exploration. Examples of such constraints are “a box with some color is not visible from a

box with same color” or “at least one box is visible from robots starting position”. This helps students to understand the complexity of real world and to select between more complicated, less reliable solutions, and more robust solutions with clearly specified constraints. An important part of the assignment is a discussion why students assume particular constraint, and how the problem could be solved if such a constraint cannot be assumed.



Fig 8 An example of the real environment with MORBot

4 Conclusion

The presented ideas and concepts of the labs have been realized within three runs of the Mobile Robotics course. During the years of the course we gain practical experience and collect valuable observation and remarks from students. A part of them have been presented in the above sections of this paper. However, the part of them that are inspiring for further course improvements are presented in the following paragraphs.

One of the interesting observation is that even though a fully configured computer for development is accessible for students, they prefer to solve the assignment at their own laptops rather than using university computers. The students also tend to use multiple operating systems instead of Windows only installation and they are pretty much familiar with modern Linux based operating systems. However, we realized that for inexperienced users an installation of the Player/Stage can be quite difficult. It is mainly due to inappropriate dependency libraries provided by the used operating system (particular Linux distribution) in its standard installation. Thus, having a prepared “Robotic Linux Distribution”, which will allow live usage from a CD or a USB FLASH drive, will be a great advantage.

We also observed that students hesitate to ask even a complicated question because they are afraid the question can be considered stupid. Sometimes, they stuck on simply solvable problem because they do not ask. Therefore the pro-active approach of the teacher is advised. The teacher can request the students to show their progress and go through the students’ codes with them.

Besides, we recognized two additional possible improvements. At first, students appreciate a mechanism that will inform them if they are behind the labs schedule, i.e. some kind of automated assignment evaluation. However, an important aspect of such evaluation has to be taken into account. It might leads to reduction of the assignments to solutions which are aimed to satisfy the submission automaton only. In such a case, the creativity and encouragement to do extra work beyond the basic assignments would be suppressed.

The second improvement is related to the used CMUcam3. The camera image is not directly accessible, thus students cannot see how particular color is changed under various illumination in real-time, which increases the difficulty to understand the problem of color recognition. This is in contrast to the main advantage of an intelligent camera that provides abstraction from the image. Even though the used on-board computer provides sufficient computation power for an eventual image processing, the main issues is in the USB interface (version 1.1), which does not support sufficient bandwidth to use a regular webcam with a raw image and sufficient frame rate and resolution.

We plan to address these improvements in further years of robotic course at FEE, CTU.

Even though not all students finish the course, it is beneficial for them, because they can take advantage of the gained knowledge in the field of mobile robotics in their bachelor or master theses. Before the Mobile Robotics course was established, we had to spend several hours with each student to introduce him/her into the field. Students that have taken the course, have sufficient background knowledge, and therefore they can focus on the core of their theses problems. These students have significantly faster progress of their theses solutions. Thus, they can bring their ideas into effect and be at or beyond the current state-of-the-art.

Acknowledgments

The work presented in this paper has been supported by the Ministry of Education of the Czech Republic under program "National research program II" by the project 2C06005 and by the Grant Agency of the Czech Technical University in Prague, grants No. SGS10/185.

References

- [1] DURRANT-WHYTE, H., BAILEY T.: "Simultaneous Localisation and Mapping (SLAM): Part I The Essential Algorithms," IEEE Robotics and Automation Magazine, vol. 2, 2006.
- [2] GERKEY, B. P., VAUGHAN, R.T., HOWARD, A.: "The Player/Stage Project: Tools for Multi-Robot and Distributed Sensor Systems," in Proceedings of the 11th International Conference on Advanced Robotics, 2003, pp. 317–323.
- [3] "The MORBot Project." <http://imr.felk.cvut.cz/morbot>.
- [4] CHUDOBA, J., MÁZL, R., PŘEUCIL, L.: "A Control System for Multi-Robotic Communities," in ETFA 2006 Proceedings, Piscataway: IEEE, 2006, pp. 827–832.
- [5] FISER, O., SZÜCSOVÁ, H., GRIMMER, V., POPELKA, J., VONÁSEK, V., KRAJNÍK, T., CHUDOBA, J.: "A Mobile Robot for Small Object Handling," in EUROBOT 2009 - International Conference on Research and Education in Robotics. Paris: LAMPA, Arts et Métiers ParisTech, 2009.
- [6] KRAJNÍK, T., CHUDOBA, J., FIŠER, O.: "A Mobile Robot for EUROBOT Mars Challenge," in Research and Education in Robotics: EUROBOT 2008 - Revised Selected Papers. Heidelberg: Springer, 2009, pp. 107–118.
- [7] GRIMMER, V.: "Platform for mobile robotics education (Bachelor's thesis)," 2008, (in Czech).
- [8] SZÜCSOVÁ, H.: "Software for mobile robotics education (Bachelor's thesis)," 2008, (in Czech).

- [9] HAVLOVIC, K.: "Periphery board for mobile robots," Master's thesis, Czech Technical University in Prague, 2008, (in Czech).
- [10] "<http://www.gumstix.com>," (accessed 28 July 2010).
- [11] "<http://www.cmucam.org>," (accessed 27 July 2010).
- [12] JANOUC, M.: "Embedded image processing," Master's thesis, Czech Technical University in Prague, 2008, (in Czech).
- [13] "<http://playerstage.sf.net>," (accessed 27 July 2010).
- [14] "Xming X Server for Windows - <http://sourceforge.net/projects/xming/>," (accessed 28 July 2010).
- [15] "<http://subversion.apache.org>," (accessed 28 July 2010).
- [16] "Sharp IR rangers - <http://www.acroname.com/robotics/info/articles/sharp/sharp.html>," (accessed 28 July 2010).
- [17] LUMELSKY, V.: Sensing, Intelligence, Motion: How Robots and Humans Move in an Unstructured World. Wiley-Interscience, 2005.

Ing. Jan Faigl, Ing. Tomáš Krajník, Ing. Karel Košnar, Bc. Hana Szücsová, Bc. Vladimír Grimmer, Ing. Libor Přeucil, CSc.

Czech Technical University in Prague
 Faculty of Electrical Engineering
 Department of Cybernetics
 Technická 2
 166 27 Prague 6
 Email: {xfaigl,krajnik,kosnar,szucsova,grimmer,preucil}@labe.felk.cvut.cz

Ing. Jan Chudoba

Czech Technical University in Prague
 Faculty of Electrical Engineering
 Center of Applied Cybernetics
 Karlovo náměstí 13
 121 35 Prague 2
 chudoba@labe.felk.cvut.cz

Web based remote mobile robot control

Jaroslav Hanzel

Abstract

The paper deals with the internet based robotics. The attention is focused on the proposal and implementation of the web based interface for the remote control of the mobile robot. The proposed system contains visual feedback to assistance the operator for safe navigation of the robot in dynamic environments. The control system utilizes the client - server architecture and is mainly implemented in the platform independent Java programming language.

Keywords: mobile robot, telerobotics, visual feedback control, Human Machine Interface

Introduction

With increasing use of the internet, the number of smart devices or systems dedicated to service, safety and entertainment is growing. These are composed of the distributed computer systems with use of the observation cameras, manipulators and mobile robots. As the idea of web robots or web-based robots is relatively new, it draws attention and interest of researchers. In addition to the control in hazardous environments, which are traditional telerobotic operations, internet extends the limits of real robots using robots in the areas known as telemanufacturing, teleeducation, telesurgery as well as a guide to a museum, in traffic control, space research, in the rescue operations during disasters, domestic cleaning or care. Although the internet provides for the teleoperations inexpensive and easily attainable information channel, there are many problems that must be resolved before the successful achievement of its real use. These problems are mainly due to the limited bandwidth and the arbitrarily large transmission delays that significantly affect the performance of telerobotic systems based on the Internet. For these reasons, it is necessary to equip the robot with a high level of autonomous behaviour. An intuitive user interface for operators is required for controlling the robot remotely.

Web based robotics uses a web browser for remote control of the robot and it differs from the traditional teleoperations in several aspects. The delay and throughput of the Internet are highly unpredictable, unlike traditional teleoperations, where the interfaces have known and guaranteed delays. Web based remote controlled robot also needs a high degree of resistance to the loss of the data packets. Web robots are controlled in most cases by people with little expertise and limited experience, unlike traditional tele-robots, which are operated by trained operators, and therefore their behaviour also become an important factor in the system design. Web robots deal with problems of a complex, dynamic environment in terms of the unpredictable delays in the network communication. Therefore their design and execution itself bring many challenges in addressing these problems.

This contribution deals with mobile robot control system via a web interface. The system should include a standard network protocol and interactive Human Machine Interface (HMI). Using a web browser, a remote operator can control

a mobile robot with visual feedback over the internet. Using an intuitive user interface allows internet users to control mobile robot and implement useful tasks remotely.

1. System design

Research on remote controlled systems deals with a new generation of network telerobotic systems for real use, such as telemanufacturing [1], teleteaching [7] and telemedicine [6]. These systems combine advanced networking technology with intelligent mobile robots [2], [4], [5]. Modern telerobotic systems should have several properties to enable their efficient and flexible use. Among those there is the requirement of the:

- universal interface for easy integration of different types of robots into the system,
- intuitive user interface and the adequate feedback,
- easy expandability of the system for adding more complex function,
- implementation of the cooperative approaches to solve complex tasks,
- high degree of autonomous robot behaviour and intelligence.

With the rapid growth of the internet, several available communication technologies are implemented in a networked environment. Current internet protocol used by web browsers is the Hypertext Transfer Protocol (HTTP). A Communication Gateway Interface (CGI) is attended to link the external applications with the web server. By means of a Hyper Text Markup Language (HTML) a requirement from the client to the server to start the process of executing a certain predetermined actions on the server can be specified. Dynamically generated HTML page can return results to the client. On the other hand, CGI has a number of shortcomings such as relatively slow speed of response. It must be also generated a complete HTML page with every client request. So this method of communication is not very suitable for remote control in real time. Contrariwise Java (object oriented programming language) offers the possibility to implement network connections and thus avoid restrictions of the CGI.

The relatively flexible and extensible approach for such tasks is to use a central server architecture [3], as shown in

Figure 1. All clients and servers are connected to a central web server. It is necessary to know the location of the web server and the reciprocal communication with each other through a web server. With this architecture all the video services and robot control services can either be provided for a single computer, or it may be possible to connect multiple computers. It is very easy to add more computers to control the robot and to process the graphical data or for the purpose of the control of more robots.

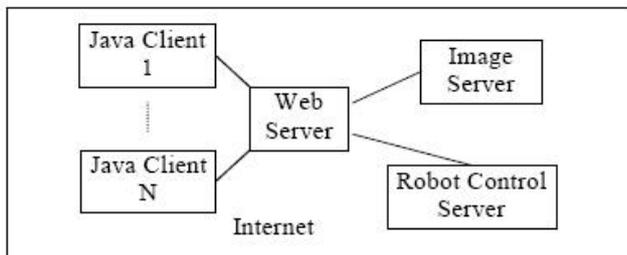


Fig.1 System architecture [3]

2. Hardware and software configuration

When designing the hardware structure of the telerobotic system, it is necessary to consider several factors related to the intended practical use of the system and it is also necessary to take financial possibilities into account. Figure 2 shows the proposal of a hardware system for the simple remote controlled robot.

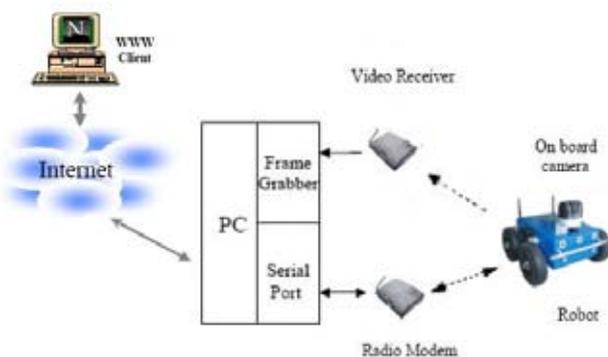


Fig.2 Configuration of the robotic system

Main host computer communicates with a mobile robot through a radio modem connected to a serial port. The main computer is connected to the network by standard network interface. The front part of the robot is equipped with a camera, that gives the user a clear view of the environment appearing before the robot. The robot can also be equipped with various sensors (eg ultrasonic, laser), which help to provide a more complex sight of the robot working environment. Video signal from the camera located on the robot is captured by the frame grabber of the main computer and it is sent to the client.

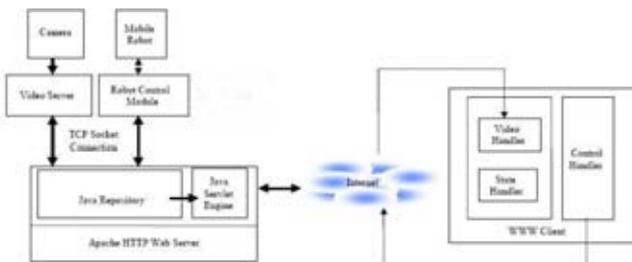


Fig.3 Software structure

As a web server the application Apache HTTP web server working on multiple platforms such as MS Windows or Linux is used. The entire software system consists of several

independent modules for optional services, each of which contains a server-side program and client-side Java applets. Java servlet in the Apache web server handles the communication between clients and servers, as shown in Figure 3.

3. Control and visual module of the system

Operating of the mobile robot is performed by the robot control module. In the primary stage of the implementation of the control module, certain basic functions such as the controls for the movement, change of the speed and stop function are inserted. More intelligent forms of the behaviour is possible to integrate afterwards.

When the system begins to function, the Java program will run and accepts commands sent from the client and controls the movement of the mobile robot by the radio modem connected to the serial port. The robot can be controlled at the same time only by one user and other users have to wait in a queue until the current operation is completed. At the same time the program sends the information from the robot, such as the ultrasonic sensor data and state of the robot, to the clients. In order to reduce transmission time, any information is transmitted in the form of the character strings and sent to all clients connected to the server. These strings are interpreted and displayed on the client side.

A key element of the mobile robot remote control is an image from a camera placed on the robot transmitted to the client side. The image quality and speed of transmission should be sufficient to provide maximum information in real time for the safe and efficient remote robot control. Number of projects dealing with the transfer of images via web are using server push technology. The video is composed from a stream of static images sent by the Java program via sockets to the Java applets. In this system, the images captured from the frame grabber are compressed into JPEG format by the software implemented in C + +. Subsequently these images are sent from the image server to the web server. Java program streams these JPEG images to all clients connected to this web server. On the client side Java applet restores the image after its receiving.

4. Web interface

Simple user interface is designed to provide basic information necessary for safe remote control of the mobile robot and it also provides the necessary basic controls. The user interface may consist of several Java applets, as shown in Figure 4.

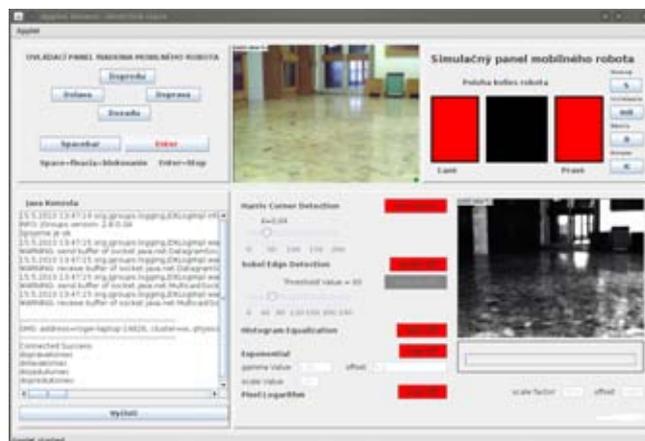


Fig.4 Implementation of the web interface

On-line instructions for the robot are processed by the control panel, which may be formed in the base version of the four directional buttons. The user can directly control the mobile robot by clicking on the direction buttons on the control panel, or by use of the keyboard for fast and complex control, such as change of speed or adjustment of the chosen speed, or eventually by input of the coordinates of the target. The image display applet shows a visual feedback in form of the continuous stream of JPEG images. The virtual environment applet can show some basic information about the mobile robot and workspace, and analyse the feedback information from the mobile robot for example in the form of the environment map. Users can monitor for example the obstacles near the robot, the traveled path and current position and speed of mobile robot.

The active user can control with this simple interface with visual feedback the movement of mobile robot. Other users can only track the visual and sensory feedback without the possibility to control the robot. These users have to wait until the first user logs off the network.

Conclusion

The aim of the paper is to analyse the options and outline a possible structure and implementation for a network telerobotic system for internet users, who can control a mobile robot in dynamic environment remotely from their home. The system allows internet users to control the mobile robot with utilization of the data obtained by the robot sensory system using a web browser. On the client side the obtained information is processed in order to encourage operator to safely control the robot. The visual feedback module provides fast image updates and presents a relatively credible real time visual information for the web users.

References

- [1] Bailey, M. J.: Tele-Manufacturing: Rapid Prototyping on the Internet, IEEE Computer Graphics and Applications, vol. 15, no. 6, pp. 20-26, Nov. 1995.
- [2] Burgard, W., et. all: The interactive museum tour-guide robot, Proceedings of the fifteenth national/tenth conference on Artificial intelligence/Innovative applications of artificial intelligence, Madison, Wisconsin, United States, pp. 11 – 18, 1998, ISBN:0-262-51098-7.
- [3] Sayouti, A., Qrichi Aniba, F., Medromi, H.: Control Architecture design for a Mobile Robot via the Internet, 9th International PhD Workshop on Systems and Control: Young Generation Viewpoint, 1. - 3. October 2008, Izola, Slovenia.
- [4] Schulz, D., Burgard, W., Cremers, A. B.: Predictive simulation of autonomous robots for tele-operation systems

using the world wide web, In IEEE/RSJ International Conference on Intelligent Robots and System, Victoria, B.C., Canada, October 1998.

[5] Simmons, R.: Xavier : An autonomous mobile robot on the web, In In International Workshop On Intelligent Robots and Systems (IROS), Victoria, Canada, 1998.

[6] Wright, D., Androuchko, L.: Telemedicine and developing countries, J Telemed Telecare, 1996, 2(2), pp. 63-70.

[7] Yuanchun S., et. all: The smart classroom: merging technologies for seamless tele-education, Pervasive Computing IEEE, April-June 2003, Volume: 2, Issue:2, pp. 47 – 55, ISSN: 1536-1268.

Acknowledgment

The work has been supported by VEGA grant 1/0690/09 and grant VMSP-P-0004-09. This support is very gratefully acknowledged.

Ing. Jaroslav Hanzel, PhD.

Slovak University of Technology in Bratislava
Faculty of Electrical Engineering and Information Technology
Institute of control and industrial informatics
Ilkovičova 3
812 19 Bratislava
Tel.: +421 (2) 60 291 864
E-mail: jaroslav.hanzel@stuba.sk

Subject „Robots“ at the CTU FEE in Prague – using LEGO robots to teach the fundamentals of feedback

Martin Hlinovský, Tomáš Polcar

Abstract

Since the beginning of the school year 2009/2010, new bachelor's program Cybernetics and Robotics offers a compulsory subject A3B99RO Robots in the first semester of study. This is a completely new type of subject provided jointly by three Departments at FEE CTU: Control Engineering, Cybernetics, and Measurements. The course with a limited number of lectures supporting required theoretical knowledge is focused mainly on independent laboratory work. The lectures are alternated by the departments mentioned above with laboratory exercises running in parallel. The assistants are supplied in laboratory teaching by promising and experienced master program students strengthening the concept of "learning by teaching".

Keywords: LEGO robots, feedback control, hardware, software

Introduction

Study of technical subjects could be very difficult for new students entering the faculty with different background. Traditional study of theoretical disciplines without any clear relevance to "real" problems decreases motivation of many students, particularly those having difficulties with advanced mathematics. Many students feel almost betrayed when they have to learn three years just lemmas and theory since they want to study robotics, not mathematics. The core objective of our new subject Robots is to explore, in friendly way, students' independent thinking, and creativity and work in team. Although the theory is limited we believe that our students will eventually improve their theoretical knowledge as well. They are not obliged to memorize theoretical formulas but they have to learn how to use them. They start with something like child game trying to move robot from one place to another. Then, to succeed in more complicated tasks, they soon recognize that there is something behind the curtain and they start to ask how to solve the problems looking for a solution actively. If they reach this stage they are "trapped" - and become excellent and highly motivated students of control engineering.

1. A3B99RO Robots: course organization

At the beginning of the semester the students are divided into small teams (4 to 6 students). Each team uses the basic set of the LEGO Mindstorms Education 9797, the set of the technical parts 9648 (additional passive components) and the network adapter 9833 (see Figure 1). The teams design and complete the mobile robot with implemented control and program it to fulfill the specified and well-revisable tasks. Eventually, the teams prepare for the final competition with their robot directly fighting the opponents in activities attractive for broad audience.

An essential element of the set LEGO Mindstorms Education 9797 and at the same time the "brain" of the robot is the central control unit known as LEGO © NXT Intelligent Brick (see Figure 2) with a matrix display 100 x 64 pixels, 4 input ports for connection of the sensors, 3 output ports for connection of the motors, a speaker with 8kHz sampling frequency, having a possibility of Bluetooth wireless

communications or an ability connecting to a USB 2.0 port. The intelligent brick and connected devices can be tested and partly controlled with the help of 4 buttons. Up to 3 servomotors can be connected to the LEGO © NXT Intelligent brick which can be used as sensors for rotational speed measurement as well. The touch sensor, the light sensor (giving the robot an ability to "see" by measuring the intensity of the light and even recognizing different colors), the sound sensor or ultrasonic sensors (enabling the robot an orientation in the space, to find obstacles and to determine the distance from them) can be connected as well.



Fig.1 Basic set of LEGO Mindstorms Education 9797, set of technical parts 9648



Fig.2 Intelligent LEGO © NXT brick and connected sensors

2. Programming LEGO robots

NXT-G - the programming language was named according to the programming language used by the LabVIEW program, developed by National Instruments, which is called only G. Abbreviation "G" comes from the fact that the programming language is graphical. Programs written in the NXT-G are thus built up of graphic blocks, with set up properties and subsequence, connected together. NXT-G is a joined product of LEGO and National Instruments and it is the basic programming tool for the LEGO MINDSTORMS NXT. The emphasis of the NXT-G is put on intuitiveness and simplicity of development environment including the programming process so that it can be used by primary school pupils with little experience in programming.

NXC - this text language derived from C language runs in the BricxCC on the standard firmware LEGO Mindstorms. It is very comfortable for those who want to program in both the NXT-G and the NXC because they do not need to upload new firmware after each change in programming environment. Working with the language abbreviating the phrase "Not Exactly C" is very comfortable and a programmer understanding at least the basics in C language becomes quickly familiar with the environment due to the almost identical semantics. Another advantage is that it is a freeware application. A disadvantage consists in complicated debugging of the programs. Unlike the NXT-G it is a purely textual programming without any graphics.

LeJOS-NXJ - The programming language distributed by Sourceforge is free and is available for Windows, Linux and MAC OS. Due to widespread expansion and knowledge of Java many users chose the LEGO MINDSTORMS LeJOS NXJ with its extensive libraries, which support interesting functions of the robot. The disadvantage is the necessity to change the firmware NXT which includes Java Virtual Machine replacing the standard LEGO firmware. LEGO firmware may be loaded into the NXT brick back using the LEGO software.

It depends on the students if they use one of recommended programming languages or use other ones (e.g. MATLAB toolbox developed at the University of Aachen (a product for users accustomed to programming in Matlab), RobotC (programming language based on C programming language), LeJOS OSEK (programming in ANSI C / C + +), or another one).

3. Solved tasks in the current school year

In the current academic year 2009/2010 students solved two tasks:

A) Follow the line - the aim of the students in this task was to build and program a robot that would independently, without any further assistance, pass along the black line marked on the mat as quickly as possible and stop at its end (see Figure 3).

Students do not know the path ahead, they know only the basic parameters of the runway and that the total length of the line will be approximately 10m. The line may be arbitrarily extended not crossing itself with a minimum curve radius 20 cm.



Fig.3 Task "Follow the line"

The students are completely free to design the robot provide they use only the parts from the borrowed sets. After three weeks of preparation, testing and software debugging two-rounded competition of all teams followed. Four teams that had reached the best time had advanced directly to the final competition (held at the end of the semester), where they subsequently competed for interesting and attractive prizes.

```
sledovani_cary.nxc
#define CARA 15
#define LOWSPEED 20
#define HISSPEED 100
#define F 20
#define I 0
#define D 100
#define DT 10

int required, actual, error, lasterr, integral, action, doleva, doprava;

task main()
{
  SetSensorLight(IN_3);
  required = (Sensor(IN_3) - CARA);
  lasterr = 0;
  integral = 0;

  do
  {
    OnFwd(OUT_BC, LOWSPEED);
  }
  while (Sensor(IN_3) > required);

  PlayTone(220, 500);

  while (true)
  {
    actual = Sensor(IN_3);
    error = required - actual;
    action = (P*(error + (I*integral*DT)/100) + (D*(error - lasterr)/DT))/10;

    if (action > 0) //musim doprava
    {
      if (action > HISSPEED) (action = HISSPEED);
      OnFwd(OUT_C, HISSPEED);
      OnFwd(OUT_B, HISSPEED);
    }
    else //musim doleva
    {
      if (action < -HISSPEED) (action = -HISSPEED);
      OnFwd(OUT_C, HISSPEED + action);
      OnFwd(OUT_B, HISSPEED);
    }

    integral += error;
    lasterr = error;
    //Wait(DT);
  }
}
```

Fig.4 Example of the program in NXC for the task "Follow the line"

The teams Jamais Contentés (Forever Dissatisfied) and Beer-go-home achieved the best times with 20.92s and 21.00s, respectively. The secret of success to achieve the best times was using PID controller for monitoring the line (example program in NXC see Figure 4).

B) Labyrinth - The aim of the second task was to build and program a robot that would independently, without any further assistance, pass through the maze from its beginning to its end as quickly as possible (see Figure 5).

The students were allowed to design robot quite arbitrarily only restricted by using the parts from the borrowed sets. Each robot had to pass through the maze from the start to the finish without any further assistance and external control. In the case of rules violation the team was immediately suspended from the competition. The minimum distance between any two maze walls was about 40cm. All maze walls were straight-line, 28 cm high, absent from any unforeseen bends and perpendicular to the bottom, i.e. there were no inclined walls. The time was measured by two light sensors located in the starting and finishing area. The total size of the maze was 330 x 160cm. The maze was built so that the shortest path between starting and finishing area was never coming back to the starting area, the passage led all the time to the target area. The choice of using sensors and control strategy depended solely on the individual teams. Of course, for the sake of orientation in the maze the robots could touch the walls. After four weeks of preparation, testing and software debugging again followed by a two-rounded competition for all teams the best fourteen went into the final contest (held at the end of the semester); their competitiveness was again strengthened by attractive prizes.



Fig.5 Task "Labyrinth"

The students solved the passage through the maze using different ways; here we describe three of them. The team called DREAM TEAM constructed a robot according to the motto "The power is in simplicity" because they were aware of the fact that the decision making of the robot represents the largest waste of time. During construction of the robot they focused on the hardware part so that the code was as simple as possible and thus faster processing the information from sensors. They created a mobile robot (see Figure 6) that touches the wall by one wheel all the time.

The program (see Figure 7) activated all three motors at one time (this ensured that the robot was permanently crashing into the wall which it went along). There was one decision element (ultrasonic sensor) in the program analyzing whether the robot went along the wall or not. If the robot went along the wall the motor, which had been slowed down due to turning round, accelerated and the robot was speeding up. Conversely, if the robot went away from the wall, the program had slowed the motor so that the robot was able to pass the curve most efficiently.



Fig.6 The robot of the team called DREAM TEAM

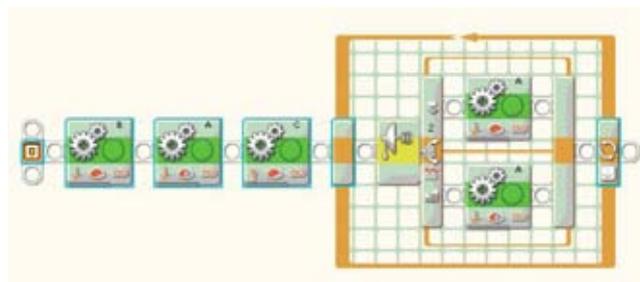


Fig.7 The program in NXT-G of the team called DREAM TEAM

The robot of the team called CENCUL'E was designed in a similar way (see Figure 8). Behavior of the robot was very simple (see program Fig. 9). The robot went straight on (motors A and B were switched on) and after hitting the wall it turned right using the motor C located in front. The ultrasonic sensor was used to detect a left curve. If the distance from the wall was more than 20cm the robot turned left (by decelerating the motor A). After re-approaching the wall the motor A was switched on again.

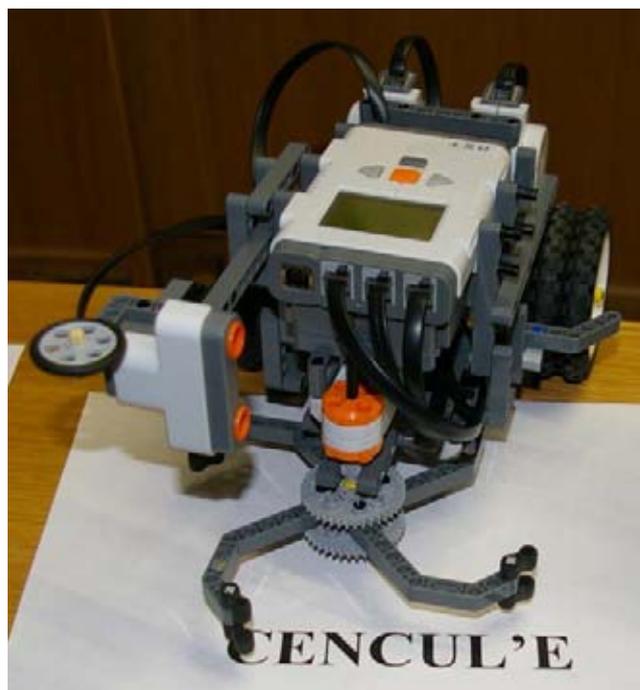


Fig.8 The robot of the team called CENCUL'E

```

bludiste_team_cencule.nxc

task main ()
{
  SetSensorLowspeed (IH_4);
  while (true)
  {
    if (SensorUS(IH_4)<20 )
    {
      OnFwd (OUT_A, 100);
      OnFwd (OUT_B, 100);
      OnRev (OUT_C, -100);
    }
    else
    {
      OnFwd (OUT_A, 100);
      OnRev (OUT_B, 0);
    }
  }
}

```

Fig.9 The program in NXC of the team called **CENCULE**

The best design from both construction and software point of view was developed by the team Jamais Contantés (see Figure 10). They used two driven wheels with individual drives (the motors were located in the opposite directions for the compact shape and the smaller inertia around the vertical axis), two focal points, the gear of 2, 4 to fast on big wheels and the side guide wheels for the case of an impact. The robot used ultrasonic sensors (measuring the distance from an obstacle in front of the robot) and the light sensor (keeping the distance from the side wall, placeable on either side).



Fig.10 The robot of the team called **JAMAIS CONTANTÉS**

The drive along the wall was solved in the code as one-level task with about 120 lines, due to the rapid passing a cycle using firmware 1.28 (allegedly many times faster than the original one) leading to the average cycle time of about 5ms. The program provided different speeds for direct drive, right and left turn and slowing down before the curve. Fast driving and right turn were controlled by two different settings of the PD controller using the light sensor. The code was optimized - due to long response the ultrasonic sensor measured the distance in front of the robot only if needed.

The following problems occurred during solving the task: low computing power, long reaction time of the ultrasonic sensor, small light sensor resolution, changing light conditions and big gap in the motor gear. Tuning the controller was possible only visually according to the behavior of the robot (it was not possible to store the values due to low computing capacity).

4. Final competition within the subject Robots

The final round which was a part of the introductory course Robots of the new bachelor's program Cybernetics and Robotics, was held on Friday, 11/12/2009 from 3PM in the Zenger's lecture-room at the CTU FEE in Prague (see Figure 11 and Figure 12). For the final competition the task "Labyrinth" was selected. Each team was allowed to use once again the basic set of LEGO Mindstorms Education 9797 and the set of technical parts 9648. Best eighteen teams from CTU FEE in Prague with two teams attending the robotic seminars in Gymnasium Voděradská (high school) measured their power in the final contest for attractive prizes. The winner was the robot which passed through the maze as the fastest one. The entire final competition was broadcasted on-line via Internet.



Fig.11 Final competition within the subject Robots



Fig.12 Final competition within the subject Robots

The parameters of the maze were announced just an hour and a half before the beginning, in order to allow finely tuning the programs of the mobile robots. In a two-rounded race both applause for the successful completion of the runway, bursts of laughter at the helplessness of a robot in the corners of the maze or even disappointment from the failure to complete the passage formed an exiting atmosphere.

Each team introduced some trick. The biggest hit was the robot which got over the walls (see Figure 13).



Fig.13 The robot which got over the wall

No one expected that the competing teams would reach such great times and the biggest surprise for us was when it became clear that students themselves have studied the basic theory of process control that would be lectured in later years, in order to gain an advantage over the other teams. We believe that without any doubt the subject makes learning more attractive. The whole atmosphere of the final round was quite exceptional. The winner of the contest became a robot team called "Cencuře" with the time of 11.455s, second place earned the team called "Jamais Contentés" and third place took already mentioned robot, which got over the walls. Interestingly, the first and second teams were separated only by two tenths of a second. And how did finish the teams from the high school? Team "Robíci" finished the competition at an excellent ninth place just one second after the winner.

And what were the prizes for the winners? The winning team received a trophy, a barrel of beer and a certificate for a thirty-minute sightseeing flight by a helicopter starting in the airport Praha-Točná going in the direction of Karlštejn, quarries Great America, Mexico and Little America. The second placed team received external hard disks and a barrel of beer, the team on the third place received flash disks and a barrel of beer, the teams on the fourth and the fifth place were awarded too.

Information about the current subject Robots, videos and photos from the final competition including the flight of the winning team can be found on the website <http://dce.fel.cvut.cz/roboti>.

Conclusions

The subject Robots is timed in the very beginning of the study, deliberately at the time when the students "know not-

hing". However, it represents a playful way how to learn the basic ideas of automatic control, cybernetics, robotics, measurement and signal processing. The initiative is raised by themselves during solving the practical tasks. Right at the beginning of study the students recognize the principles of the creative engineering and research work.

The aim of the course Robots is to excite an interest in the branch, its main ideas and opportunities, while encouraging students to ask and study. We hope that the course will give them enough motivation to pass the difficult mathematical and technical courses during their studies. Moreover, the course and final competition is a very effective way to show FEE as the one of the most progressive faculties in the field of Robotics and Control Engineering in the Czech Republic.

Finally, we are preparing an international competition with our German colleagues for the next year.

Acknowledgements

This research has been supported by the MŠMT FRVŠ project no. 33/100035/13135 Robots – the motivation subject of the new bachelor's program Cybernetics and robotics.

References

- [1] TROJANEK, P.: Usage of the LEGO robots in education, Prague 2009, Bachelor thesis, CTU FEE in Prague
- [2] RIEBER, J., WEHLAN, H., ALLGÖWER, F.: The ROBORACE CONTEST – Using LEGO robots to teach of the fundamental of feedback control, IEEE Control Systems Magazine, October 2004
- [3] MARTINEC, D.: Usage of the Lego Mindstorms Robot – Design and realization of the tasks, manual for programming in LeJOS-NXJ, Prague 2010, Bachelor thesis, CTU FEE in Prague
- [4] MOC, I.: Usage of the Lego Mindstorms Robot – Design and realization of the tasks, manual for programming in NXC, Prague 2010, Bachelor thesis, CTU FEE in Prague
- [5] BELIK, T.: Usage of the Lego Mindstorms Robot – Design and realization of the special tasks, Prague 2010, Bachelor thesis, CTU FEE in Prague
- [6] ROBORACE (in German) – online. Available: <http://www.ist.uni-stuttgart.de/roborace>
- [7] LEGO.com Mindstorms – online. Available: <http://www.mindstorms.com>

Ing. Martin Hlinovský, Ph.D.

Czech Technical University in Prague
Faculty of Electrical Engineering
Department of Control Engineering
Karlovo namesti 13
121 35 Prague 2
Phone: +420 22435 7477
E-mail: hlinovsm@fel.cvut.cz

Lessons learnt with LEGO Mindstorms: from beginner to teaching robotics

Martina Kabátová and Janka Pekárová

Abstract

In this paper we describe several lessons of educational robotics at different level of robotics experience. First one focuses on developing basic skills needed to successfully control a robotic model. The other one uses advanced programming skills to provide the communication between two programmable bricks. We also describe organization of the project in brief. LEGO Mindstorms kit was used as a learning platform for all activities. Addressing differences between beginners and experienced students, we list a few tips and recommendations how to execute quality robotic lessons.

Keywords: robotics education, LEGO NXT, programming, guidance, project

Introduction

Robotics presents an attractive introduction to the object-oriented programming or higher programming languages (see [2], [5], [10]), but it can be also used in the lower levels of education. Dealing with real robots has a high motivational effect – students visualize their robot as a toy [11] which behavior can be set according to the scenario where it is used. The experiences with robots are tangible although their design requires much abstract thinking. Finally, they enable rich varieties of interdisciplinary projects. Therefore we consider robotics to be a powerful tool for developing thinking. We pay special attention to the preparation of pre-service teachers who can enrich their teaching repertoire by the robots' use. We have been realizing the robotic seminar for them for several years. We try to explore the effective way for constructing students' comprehension of robotics. We often find our methodology similar to different courses worldwide.

1. How robotics is taught

Carnegie Mellon Robotics Academy [3] offers a special robotic course for educators. Here they learn more about:

- MOTION and CALCULUS (What is a robot?; Mindstorms hardware; Movement and rotations; Size, distance and movement; Abstract bridges; Challenge: go as close as possible)
- ROBOTIC SENSORS (Measure – Plan – Execute strategies overview; Touch, ultrasonic, light and sound sensor; View Menu; 'Wait for' block; Limits of the measurements; Tasks with sensors)
- DECISION MAKING (Repetition; Obstacle detection; Cycle, condition and conditional cycle; Line following; Setting the ride through obstacles; Iterative solution of the problems; Challenge: ride through obstacles)
- EDUCATIONAL ROBOTICS (Possibilities; Challenges; Robotics in your school)

Different approach to the educational robotics can be found in TERCOP project [1]. Teachers learn how to teach robotics in a constructionist way. Besides the basics of programming the kits, participants also get informed about constructionist learning philosophy and project-oriented classes. They analyze and assess the robotic model, suggest own assignment for their students and think over the organization of robotics projects. In the core lessons of the project they learn to measure with sensors and control the motors of the robot using the basic program blocks. Moreover, they learn how to check if robot works in the way prescribed in assignment and how to modify the program in order to fit assignment needs.

MIT Lifelong kindergarten applies four principles into their leisure time robotics workshops for children and families [13]: (a) Focus on **Themes** (not just Challenges); (b) Combine **Art** and **Engineering**; (c) Encourage **Storytelling** and (d) Organize **Exhibitions** (rather than Competitions). Authors of this learning approach aim to make robotics attractive to the as broad range of people as possible.

[9] puts emphasize on creating **cooperative learning environment** where **small groups** of students maximize each other's learning while working on robotics projects. In the proposed curriculum students at first work with ready model, in order to understand possibilities and limitation of robotic kit. They use programs prepared in advance, learn to modify them and after that try to design a functional robotic model on their own. Similar approach is presented in [14]. **Challenges** are taken in the end of each lesson in order to **ensure understanding** to the core concepts of the lessons.

Youth Engaged in Technology programme also combines **team building activities** with **demonstration of programming instruction** to the robot [6]. This course further contains necessary math to calculate gear ratios. Several exercises are focused on building and mechanical components of robots. Open-ended challenge completes the course syllabus.

[8] has decided to give his robotic course the form of open lab where students can spend **much time** on solving various robotic related tasks which encourages students to be more creative in their design and robot implementation. The assignments are close-ended and clearly defined, although the author recognizes the potential of the **open-ended projects in combination with the contests**.

2. Seminar Robotic kits in education in detail

We have been applying constructionist ideas and principles ([12]; [13]) in our seminar practice:

- **learning by doing, hands on activities** through own experiences – students build and program robotic model and test its functionality;
- **authentic success in finding the problems and their solutions** – students decide how the robot will work and choose their way how to achieve this, they select the topic for their project and explore programming possibilities;
- **the hard fun and playful learning** – robotic kits are in fact the toys, but solving some tasks with them can be quite complicated, the atmosphere on the seminar is relaxed and we try to help students learn in an entertaining way;
- **creative learning by designing and inventing** is included in the creating the robotic model;
- **combination of digital technologies as a building material together with art materials** – students can decorate their robots, make a costume for their models or produce some coulisses;
- **enough time** – the syllabus of the seminar is quite flexible, we can spend more time on activities that last „too long“;
- **freedom to make mistakes** and learn from them – students get space for their own, independent solutions in which they go wrong sometimes, we try to reveal the core of problem in common dialogue and help them to solve it;
- **teamwork, collaboration**, distribution of roles within the group, common work on problem solving – students learn how to organize teamwork, some assignments cannot be completed individually (e.g. to prepare the robot for the contest);
- **learning together** – it is not possible for us – the teachers – to be prepared for the whole range of troubles that can happen, we also solve novel tasks and learn new things together with our students.

The syllabus of the seminar keeps balance between closed tasks having the only solution and open-ended projects:

Week	Topic of the seminar	Tasks, solution, teamwork
1	Programming without computer?	Simple closed tasks, one solution, small group
	Creating simple program through NXT brick interface	
2	Introduction to Mindstorms programming	Simple closed tasks, one solution, small group
	Move, Display and Wait for block, Cycle, Condition	

Week	Topic of the seminar	Tasks, solution, teamwork
3 - 4	More on Mindstorms programming	Simple closed tasks, one solution, small group
		Procedures and variables; Parallel processes
5	Experiments with sensors	Single task mixed from small group work and common activity with all students
		Read/Write to file, variables
6	Communication between robots	Advanced tasks for small groups collaboration
		Send/Receive message
7 – 9	Our Project 1. Preparation for the robotic contest as an alternative	Open-ended project, many possible solutions, small group
10 – 12	Our Project 2. Exhibition and presentation of the models	Open-ended project, many possible solutions, small group builds single model for the common topic

Tab. 1 Syllabus of the seminar

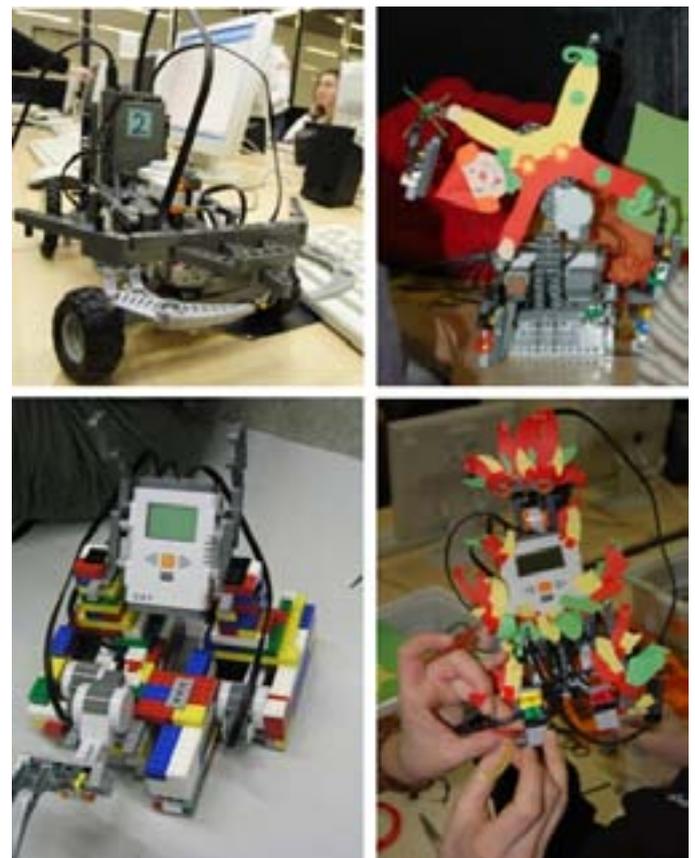


Fig. 1 Our Project – examples of outputs

We have sketched the way of organizing open-ended projects in [7]. Briefly, during previous terms students:

- built and programmed robotic elevator controlled by touch sensors (see also section V),
- suggested and realized robotic models for the Space and Playground topics,
- designed and completed moving pieces of a Spooky castle, Amusement park and Intelligent house.

We encourage students to do **pedagogical reflection** of their own robotic design. The part of robotic assignment is also a teacher checklist where they advice fictional teacher how they can realize similar robotic project – how much time is needed, special needs of hardware, previous programming skills, common problems and their solution. Although documentation of the work in this way isn't very popular among students, they are able to produce high quality description of their robotic device.

In the following sections we describe three lessons: (a) an initial lesson in Week 2 where students work in the programming environment for the first time; (b) communicating between robots in Week 6 and (c) design of the project-based activity The Lift. We want to show the continual progress from simple close-ended tasks that help the beginner to develop basic programmer skills to change the robot's behavior to more open-ended tasks that require more creativity and open space for more qualitatively different solutions. We think it is necessary to give participants the possibility to **design, build and program the robotic model on their own**. This brings additional effect: the pre-service teachers experience some common problems when designing the robots and can more effectively give advice to their own students in future.

3. Lesson one: controlling the robot

The students use the standard model of the robot (as presented in Robot Educator section of Mindstorms programming environment). We provide them by the set of clearly defined, simple tasks so that they get to know the possibilities of the iconic programming environment used to program the robot. During previous terms we noticed several problems and misconceptions common to various groups of students. Let's have a closer look at some of them.

Robo – archeologist Assignment: Choose one of the letters written on the stone.

- Teach your robot to move in the outline of the letter.
- Teach it to move in the outline of crocodile teeth:




In the very first assignment which should be solved in Mindstorms programming environment the students express high expectations of robot's possibilities. Many of them ask whether the robot should follow the black outline of the letter and some students even don't ask and try to program it. The main idea of the assignment is in fact much simpler: they should learn to use Move block and use the sequence of this type of block to produce the track of the robot in a fixed dimensions (we advise the students the blocks needed in solution on the edge of the assignment).

In trying to create the crocodile teeth track students find out that the angle used in Move block isn't the same angle in which the robot turns. The angle stands for the rotations of the motor – 180o means half of one rotation. Solution of the task is then often based on many trial-and-error experiments with the settings of Move block in order to create a desired outcome.

Our students are the experienced programmers when beginning to attend the seminar. At this point they suddenly find out that programming the real robots differs from programming virtual ones (e.g. a turtle in Logo programming language) – they have to consider physical

aspects of the model as well as the properties of the environment. When they program a robot that should move after whistling, some of them face the problem that robot will start moving immediately after the program launches (as the noise in room is often high). They discover the need to calibrate the sensors.

We always encounter students who try to program a continuous motion of the robot (no matter how outer conditions are). They soon question why the program containing only one Move block set to Unlimited steps doesn't work as they expected. This is the other difference between programming real and virtual creatures that needs to be explained explicitly and perhaps demonstrated in more guided instructional way.

Students learn to set robot's behavior depending on outer conditions – values measured by sensors in several real-life tasks, for instance:

Robo-racer Assignment: Your robot is waiting for the start-the-race signal. When hearing it, it will move forward.

- Teach it to take its run – to increase its speed.
- It will quickly go forward. It will stop when it finds the (black) line marking the finish of the race circuit.
- After achieving the finish line, the robot will turn all around because of the joy from victory.
- Each racer will smile after turning around – there will be a smiling face on its display.



Lesson finishes by the task requiring a partial disassembling of the robot. We were inspired by [4] in its assignment – the challenge was to increase the robot's speed so that it will move faster than the programming environment allows by default:

The Thief Assignment: Try to achieve as fast motion of the robot as possible. Find at least two different solutions.
Hint: You may need to modify the construction of your robot slightly. **Notice how the power of the motor transfers to the wheels.**

Besides experimenting with the gears and program settings students should find the answers to these questions:

- What will happen if you enlarge the cog wheel connected to the motor and use smaller cog wheel connected to the wheel?
- What will happen in opposite case?
- How much load can the robot push if you use various types of cog wheels?

We motivated the students by the short movies in [4] and discussed the answers to the questions with them. They had chance to create hypothesis and test it immediately in real conditions. This task was appreciated also by two girls which showed no previous interest in mechanical issues of robotics and they were proud of themselves that they found the arguments supporting their opinions. Finding solution to this task requires lot of time (because of the need to assembling new driving mechanism for the robot) and the task should be included as the last piece of the lesson because of the need to re-build the robot construction. It also inspires future teachers to think in an interdisciplinary way – they think over the math hidden in mechanics and construction of the robotic models.

4. Lesson two: the robots communicate

As soon as the students manage the basics of programming language, one lesson is dedicated to the communication between two (or more) NXT bricks. The bricks have Bluetooth and can be programmed to send and receive various messages. Unfortunately the process of connecting two bricks has some flaws. We found out that creating a connection should be guided as much as possible as the process is nothing near intuitive and detailed guidance can significantly reduce mistakes and errors that have nothing to do with controlling the robots.

Once the bricks are successfully connected the teams shall write programs to send and receive messages. In first simple introductory assignment one brick sends a word and the other brick displays the word on its display. We used to give another introductory assignment but we realized that it is not easy if it should be done correctly (the task is to send a Morse signal and the other brick should beep the message out).

The final task is to program a car and its remote controller.

Remote-control car Assignment:

Team A: Create a program that will send control messages to the car. Use NXT buttons on your brick as a remote controller.

Team B: Create a program to receive messages and move the car according to them.

Both teams: Negotiate the message content and how to interpret them. Think about car controlling - what is the desired behavior that will be suitable for a race?

The programs contains loop, if-statement and reaction to the NXT buttons or blocks to move the motors. There are several good solutions either with variables or without using them. We encourage the students to find their own solution and we only correct their errors once they ask us. The most common solution for receiving program contains two or more nested if-statements reacting to various messages from the other NXT brick.

Students should also design the behavior of the car, once it is successfully controlled. After few takeouts they realize it is important to make the car respond in certain way to be able to navigate it around racing circuit. This task is very closely connected with actual environment and usability of their model.

This lesson culminates in a competition. We measure the time needed to finish the racing circuit and give penalty seconds for bumping into circuit borders.



Fig. 2 Testing the communication between robots on the race circuit

Realizing short contest with remote-control cars is our reaction to recent feedback from students who have missed challenges and opportunities for the competitions besides the exhibitions (organizing exhibitions is strongly encouraged in [13]). Still, some groups of the students are not interested in the possibility to compare with the others at all. We assume that teachers should provide

opportunities for exhibitions as well as for the contests in robotics classes, in order to suit the learning style of most students.

5. Our project: a LIFT as fast as possible

In the winter term of 2007 we introduced the open ended project-like activity named The Lift Model. At the end of this term the activity was rated as most popular among the students.

The assignment for the students:

Design the structure and devices of functional lift,

- build the frame tower, the cabin and some mechanism that will pull the cabin,
- program the sensors to act as lift controls (e.g. the touch sensor can be a button).

Your task is to build a functional model of a lift. You will use LEGO NXT kit, sensors and programming language. You decide how the frame will look like and how the lift will be controlled. Your model should be able to go up and down as the user needs. Note that

- the frame should be high enough,
- the frame should be steady and should not lean to the sides,
- the cabin should not tilt or spin, it should have the most stable position,
- if the cabin is too heavy, use right gears to lift it although at lesser speed.

Challenge: try to build the fastest lift possible, is your lift faster than models built by other groups?

The material for the students included also some reference photographs of various lifts, pictures of the lift that was built by us and a list of LEGO bricks we have used in our model. We couldn't bring the actual LEGO model since all kits were used by the students and there was no spare kit for the model. According to [13], for students it is very inspirational to see sample models, they have more ideas and identify the problems they are about to solve easier. As there is often a problem with material and many teachers don't have spare kits we suggest using photographs and videos instead.

Originally we expected the project to take 3 lessons (each lasts 90 minutes), but in the process we realized 4 lessons are needed to finish all the work and to present the models.



Fig. 3 A lift controlled by two touch sensors. The cabin is lifted by two motors placed on the top of frame.

Three out of four teams finished their models so they were fully functional. Two of those models used double motors at top of frame to pull the cabin. The third solution was a single motor placed at the base of tower. As we anticipated the highest tower was also the most unstable. It is noteworthy that all teams placed some LEGO figures or other decorations on their models (one of the girls even brought her own bricks from home). This indicates that the students should have the opportunity to use additional decorative materials to enhance their finished models.

This activity was open-ended but it's nature didn't leave much space for students own inventions and creativity. They also voiced this opinion in the final interview. In next courses we introduced more theme based and even more open-ended activities, though we think there are students that need more guidance and instructions during deciding what they are going to design and build. We suggest that the creative robotics principle "focus on theme" [13] is a good way to give students the basic layout of what they are going to do, but the teacher should focus on the less skilled and less creative teams and help them to find more tangible model description - this can be done via guided discussion. We strongly agree that the process of finding the problem is equally or even more important than actually solving the problem.

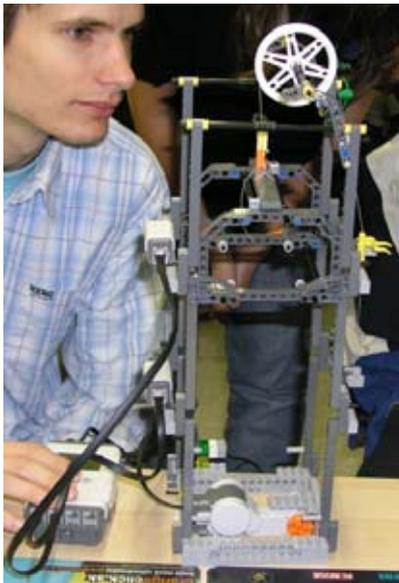


Fig. 4 A lift using light signals

At the end of this activity we let the teams to present their models. Unfortunately we were not able to evaluate the challenge for the fastest lift. This should have been done and according to the final interviews students also expected some competition like activities. We think that it's reasonable to include both types of evaluation - competition and exhibition, as they appeal to different students. In case of our seminar the need for competitions is given by specific target group (most of the course participants are computer science, mathematics and management students and males).

Conclusions

We have described design of our robotics course for pre-service teachers and computer science students. After several iterations of whole course and introducing various types of assignments we propose that at the beginning students should solve smaller close-ended tasks with

basic robotic model they do not build. This way they can learn about programming the robotic model and experience basic principles of event driven robot controlling. In later lessons it's reasonable to give students more space to create their own model and follow more constructionist lesson design with plenty of time for experiments.

Three lessons we detailed suggest that different amount of guidance and instruction is appropriate for various activities.

We have also applied some principles of creative robotics [13] and we argue their relevance for our specific target group.

To sum up, there are some relevant issues that should be considered when teaching robotics computer science students and pre-service teachers, in our course design we try to address them.

References

- [1] ALIMISIS, D., MORO, M., ARLEGUI, J., FRANGOU, S. and PAPANIKOLAOU, K.: "Robotics & Constructivism in Education: the TERECoP project". In Kalas, I. (ed.) Proceedings of EuroLogo 2007. Bratislava, 2007.
- [2] BARNES, D.: "Teaching Introductory Java through LEGO MINDSTORMS Models." In Proceedings of the 33rd SIGCSE technical symposium on Computer science education, 2002, pp. 147 - 151
- [3] CARNEGIE MELLON UNIVERSITY: Professional Development Robotics Academy Classes, available 31.5.2010 at http://www.education.rec.ri.cmu.edu/content/educators/professional_dev/index.htm
- [4] CARNEGIE MELLON ROBOTICS ACADEMY: Introduction to Robotics; Using the NXT. Electronic learning material, 2006
- [5] FLOWERS, T.R., GOSSETT, K. A.: Teaching Problem Solving, Computing, and Information Technology with Robots. In Journal of Computing Sciences in Colleges archive Volume 17 , Issue 6 , pp. 45 – 55, May 2002
- [6] HOY, P., WEBSTER, P., DIMARZIO, C., BATTERSON, T. and PERKINS, D.F.: Robotics Curriculum: Using LEGO® MINDSTORMS® Robotics kits. In Pennsylvania State University: Youth Engaged in Technology (YET) Programme, 2006.
- [7] KABÁTOVÁ, M., PEKÁROVÁ, J., Learning how to teach robotics. Paper accepted for Constructionism 2010 conference. In press.
- [8] KUMAR, A. N.: Three Years of Using Robots in an Artificial Intelligence Course – Lessons Learned. In ACM Journal on Educational Resources in Computing, Vol. 4, No. 3, Article 1, September 2004.
- [9] LAU K. W., TAN H. K., ERWIN B. T., PETROVIC P.: Creative Learning in School with LEGO Programmable Robotics Products. In Proceedings to Frontiers in Education'99, IEEE CS Press, pp. 12D4/26 - 12D4/31 vol.2, 1999.
- [10] LAWHEAD, P.B., BLAND C.G., BARNES, D.J., DUNCAN, M.E., GOLDWEBER, M., HOLLINGSWORTH, R.G., SCHEP, M.: A Road Map for Teaching Introductory Programming Using LEGO® Mindstorms Robots. In ACM SIGCSE Bulletin Volume 35, Issue 2, pp. 191 – 201, June 2003.

[11] MAUCH, E.: Using Technological Innovation to Improve the Problem-Solving Skills of Middle School Students Educators' Experiences with the LEGO Mindstorms Robotic Invention System." In The Clearing House, Vol. 74, No. 4, pp. 211-213, Mar. - Apr., 2001

[12] PAPERT, S.: The Eight Big Ideas of the Constructionist Learning Laboratory". Unpublished internal document.

[13] RUSK, N., RESNICK, M., BERG, R., and PEZALLA-GRANLUND, M.: New Pathways into Robotics: Strategies for Broadening Participation." In Journal of Science Education and Technology, vol. 17, no. 1, pp. 59-69, 2008.

[14] Sklar, E., Eguchi, A.: Learning while Teaching Robotics. In The AAAI Symposium Series: Accessible Hands-on Artificial Intelligence and Robotics Education, 2004.

The seminar webpage is available at <http://edi.fmph.uniba.sk/lego> and it contains lessons assignments (in Slovak), answers to some of them written by the students as well as the picture gallery of seminar work.

PaedDr. Martina Kabátová

Comenius University
Faculty of Mathematics, Physics and Informatics
Department of Informatics Education
Mlynská dolina
84248 Bratislava
Phone: +421 2 60295 395
E-mail: martina.kabatova@fmph.uniba.sk

PaedDr. Janka Pekárová

Comenius University
Faculty of Mathematics, Physics and Informatics
Department of Informatics Education
Mlynská dolina
84248 Bratislava
E-mail: jana.pekarova@fmph.uniba.sk

A Monocular Navigation System for RoboTour Competition

Tomáš Krajník, Jan Faigl, Vojtěch Vonásek, Hana Szücsová, Ondřej Fišer, Libor Přeučil

Abstract

In this paper, we present a mobile robot navigation system used in the RoboTour challenge. We describe the basic principles of the navigation methods and show how to combine monocular vision and odometry. We propose to use the monocular vision to determine only the robot's heading and the odometry to estimate only the traveled distance. We show that the heading estimation itself can suppress odometric cumulative errors and outline a mathematical proof of this statement. The practical result of the proof is that even simple algorithms capable to estimate just the heading can be used as a base for "record and replay" techniques. Beside the navigational principles, practical implementation of our navigation system is described. It is based on image processing algorithms for path following and landmark-based crossing traversing. An overview of experimental results is presented as well.

Keywords: visual navigation, robotic

Introduction

The RoboTour contest is aimed at outdoor autonomous robots capable of delivering a payload to a given destination. Although the competition rules have evolved over time, the basic idea is the same: an autonomous robot should traverse a given path on walkways in a park-like environment. The contest is therefore closely related to the safe mobile robot navigation in large outdoor environments. The first year of the contest has been organized in 2006 and we have participated in each year until now. During our participation, we used the contest as an evaluation scenario for our navigational algorithms as it is a great opportunity to test the reliability of the whole navigation system. At the beginning, none of our navigation algorithm could be directly used in an outdoor environment. In that time, our group at the Gerstner Laboratory has been strictly focused to indoor navigation techniques. So, from a certain point of view, we have been in a similar position to other teams participating in the first year. However, we had an advantage of sensors equipment and know-how in navigational principles.

One of the most important RoboTour rules is that a robot which leaves the pathway will be penalized. Considering the basic definition of the delivery task, we realized, that the main principle of the navigation should be based on path following. Many teams tried to solve the task by using high-precision GPS receivers. However, the GPS signal in park-like environments suffers from reflections and occlusions and therefore, the GPS precision is around thirty meters, which is insufficient to keep the robot on the narrow pathways. The GPS can be complemented by an odometry and a compass to estimate the robot position using Kalman filtering methods. Even through this sensor fusion can improve position estimation, it does not provide sufficient precision to keep the robot on the pathways. The odometric error tends to accumulate over time and therefore these methods do not perform better than sole GPS-based localization in the long term. Although the odometry is precise for travelled distance estimation, it cannot provide sufficiently good heading estimation, thus it is unsuitable for long-term robot localization.

Moreover none of the aforementioned sensors provide any information about the robot surrounding environment. A robot using these sensors is navigated by a simple control law to the desired coordinates and ignores the situation in its vicinity. The reliability of its navigation is determined solely by precision of its GPS receiver. Regarding to this issue, the aforementioned sensors should be complemented by exteroceptors like digital cameras or rangefinders providing information about the robot surrounding environment. Having an intelligent robot that will be able to recognize pathways, crossings and obstacles from its sensors, the precise position estimation is not needed at all. These evidences were main ideas in our choice of research direction in reliable robust autonomous navigation for outdoor environment, which is to build an intelligent mobile robot (possibly from cheap off-the-shelf components) that will be capable of recognizing its surrounding environment and to select the most appropriate action to fulfill its goal.

To build such an intelligent mobile robot the "only" thing is to "process" its sensory data. The most common sensors in mobile robotics are cameras and laser rangefinders. While laser rangefinders are precise and robust to changing illumination, they are prohibitively expensive to most of the teams. Digital cameras are cheaper, but image recognition in outdoor environments is not a simple task, because of the complexity of the outdoor environments. In particular, the pathways differ in color, texture, width and often are interrupted by ruptures with vegetation. The recognition is complicated by slopes, fallen leaves, dirt, shadows of surrounding trees, image blur caused by robot movement and variable illumination. However, the problem of path recognition has been successfully solved years ago [1].

Once the path following is solved, a more challenging problem of reliable and robust crossing recognition arises. Many teams have successfully achieved implementation of a reliable path recognition, but still have problems with the crossings. This was also our case in the year 2006. Our robot, equipped with the "GeNav" [2] algorithm, was able to follow the pathways with sufficient reliability, but lacked the ability to choose the right paths on large crossings.

Based on our experience gained in the first RoboTour contest, we have proposed to use different image processing methods for the path and crossing traversal. The path traversing algorithm is based on path recognition, while crossing navigation is based on visual landmark recognition.

Because our intention is to build a reliable and robust navigation system, we have decided to use map-based navigation methods, as it is known that such methods are more reliable than mapless techniques. Moreover, we have decided to not follow mistakes made in seventies in the field of Artificial Intelligence (AI), i.e. to use knowledge representation that is natural to humans, but not to machines. Instead, we considered the new AI concepts and let the robot use knowledge representation that is natural to its sensory equipment and reasoning abilities. Therefore the map, which is used by our robot, is not easily interpretable by a human at a first glance.

Our approach lead to a minimalistic monocular navigation system capable of navigation within a known environment. The robot utilizes a map just to correct its heading and measures its position by a relatively imprecise odometry. We claim, that the heading corrections efficiently suppress the cumulative errors of odometry. We formulate a particular instance of the navigation method mathematically and provide a proof of the claim. The practical implementation of the method won the RoboTour challenges in years 2008 and 2009. Beside our RoboTour participation, we examined the method in several outdoor experiments that confirm the system performance. In this paper, we present the main ideas of our methods used in the RoboTour challenges and describe practical issues that have been met during realization of the system for the RoboTour contest.

The paper is organized as follows. The principles of the proposed navigation methods are described in the next section. A mathematical model of the navigation methods is outlined and its properties are examined in Section 2. The implementation details of the methods are described in Section 3. The experimental results evaluating the system performance are outlined in Section 4. The conclusion discusses the proposed navigation method and outlines possible future improvements.

1. Principles of the Navigation Methods

This section provides an overview of the main navigation principles used in our system based on two navigation algorithms. We assume that the robot has a differential, nonholonomic drive and its movement can be described by equations

$$\begin{aligned} \dot{x} &= v \cos(\varphi) \\ \dot{y} &= v \sin(\varphi) \\ \dot{\varphi} &= \omega \end{aligned}$$

where x, y denote the robot position, φ is its heading and v and ω denote forward and steering velocities. In general, the task of a navigation algorithm is a computation of the velocities v and ω from the actual and past sensory data. To simplify the equations describing the robot movement based on sensory data, we assume the robot is capable of fast turns and the steering actuator sets the robot heading φ directly.

The first navigation algorithm is a simple path following algorithm, which is capable to keep the robot on the path. Reliable path following has been solved by most teams, but the crossing recognition remains a problem. In our opinion, the reason for it stands in the fact that most of the robots are small and their cameras are not very high above ground.

Moreover, the camera is usually firmly fixed on the robot body and therefore, the robot cannot “look around” when following a path. For a small robot, a crossing is more or less an open space, and therefore the robot cannot distinguish between crossings and wide paths, see Fig. 1 for an example of crossing. Moreover, it sparsely spots all the exiting paths. Due to these facts, we have decided to use an algorithm based on objects recognition for crossing traversal. The advantage of the algorithm is that it does not rely on environment structure, so it can also be used in areas that are not divided to paths and obstacles. On the other side, a possible disadvantage can be in the higher computational cost of object recognition and complexity of the map. Even through these algorithms seem to be different, they use the same navigational principle. Let us review the main principles of the path following and landmark-based navigation techniques.



Fig. 1 A crossing, which is difficult to survey.

The path following algorithm can be fairly simple. A robot controlled by the algorithm has to move forwards and keep itself on the path. Suppose, that the robot knows how to recognize the borders of the path in the image from its camera. This can be achieved by several ways, ranging from a simple perceptron working in the RGB space to elaborate methods combining several computer vision algorithms [1].

1.1 Path Following

If the borders are identified, the robot can compute which border is closer and steer in the other direction. Therefore, the robot adjusts its heading to keep itself in the middle of the path. Now, suppose that the robot is moving along a straight path. A 2D coordinate system $x-y$ can be defined such that the x -axis is at the path center. Then, the heading φ of a robot following the path can be expressed as

$$\varphi \approx -K_0 y$$

where K_0 is a positive constant.

1.2 Landmark-based Navigation

Suppose that the robot has a map of the environment containing descriptions and positions of salient objects. The map allows computation of objects that will be visible from a particular robot position. While the robot approaches a particular position, it retrieves objects from the map and matches them to the actual set of seen objects. Because the objects are provided with the image coordinates, the robot heading correction can be directly based on the horizontal coordinates. Assume the expected robot position and heading are zero, but its true position and heading is $(0, y, 0)$. The actually seen objects are not in the expected positions

in the image, but they are shifted to the right, for $y > 0$ and vice versa. In this case, the robot can steer in a direction that will minimize the differences in horizontal coordinates of the expected and detected objects. In other words, the robot heading φ satisfies similar constrain as in the previous case:

$$\varphi \approx -K_1 y .$$

Even though the algorithm realized the so-called visual compass, it can be used also for the path traversing.

1.3 Making Turns

Here, it should be noted that both algorithms suppose the robot heading is close to the right direction. If a robot following a path is suddenly turned by 180° , it will just continue in the wrong direction. Similarly, if a robot does not see the expected objects, the landmark-based navigation fails to correct the robot heading. Therefore, none of these algorithms are suitable to make sharp turns. In such cases, the robot has to stop and turn to the requested direction using its compass data.

1.4 Switching the Methods

Having these three methods: path following, landmark-based navigation and turn making a robot should be able to traverse paths and crossings in five steps. The robot would (1) approach a crossing by path following, then (2) traverse to the crossing center by landmarks, (3) turn towards a particular crossing exit, (4) use landmarks to approach the exit and (5) finally switch to path following to traverse to the next crossing.

The final problem stands in the mechanism of algorithms switching. Once the robot misses the crossing center, it will unlikely find the appropriate exit. The crossing is difficult to detect due to insufficient field of view and resolution of common cameras for achieving an overview of the complete crossing. Also GPS-based localization methods are not precise enough in parks. Although traditional image-based localization is more precise than the GPS, it is not robust to weather and seasonal environment changes [4].

On the contrary, the distances a robot has to travel between individual crossings are well known and can be measured by the odometry, which provides sufficient precision as it is used only locally between the crossings. Therefore, the robot can utilize odometry to decide if it has arrived at a particular crossing. The intended path of the robot may be split into several segments with a different length. A plan to traverse a simple path with one crossing can be as follows: *Follow the path by 46 meters, then go 4 meters towards the red tree, turn by 90 degrees, go 7 meters towards the blue dustbin and follow the path for 43 meters.* One might argue that the cumulative nature of the odometry error would cause this plan to fail, because both of the aforementioned navigation algorithms compensate only the lateral position error. In the next section, we explain that despite their simplicity, the algorithms can compensate the odometry error.

2. Navigation Stability

In this section, we show how the aforementioned navigational methods compensate the odometry error. At first, we introduce a model of the robot movement along the plan consisting from a sequence of path segments with various lengths. Based on the model, we explore navigation properties and outline a proof of the navigation stability. The main idea of the proof is based on a robot position uncertainty as it moves along a segment. A navigation

method is stable if the uncertainty does not diverge, which holds for our navigational methods used in the RoboTour competitions.

2.1 A Model of Robot Movement

Assume the same situation as in the previous section, i.e. the robot moves along a path lying on the x axis of the 2D coordinate system. A robot position evolution as it moves along the path can be described by relations of $y(t)$ and $x(t)$ in dependence of the controller's action values. The robot forward speed controller maintains constant speed until the robot has traversed path longer or equal to the segment length. Let us assume, that the robot steering controller sets the robot heading to drive the robot towards the segment axis, i.e. $\varphi = -k y$, where k is a positive nonzero constant. Assuming that the robot heading φ is small, we can state that $x = -v_k$ and $y = -v_k \varphi$. Solving these two differential equations with boundary conditions $x(0) = a_x$, $y(0) = a_y$, $t = s / v_k$ gives the robot position (b_x, b_y) after it completes the path:

$$\begin{aligned} b_x &= s + a_x \\ b_y &= a_y e^{-ks} . \end{aligned} \quad (1)$$

Equation (1) would hold for an error-less odometry and noiseless camera. Considering the odometry and camera noises, the equation can be rewritten to

$$\begin{pmatrix} b_x \\ b_y \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & e^{-ks} \end{pmatrix} \begin{pmatrix} a_x \\ a_y \end{pmatrix} + s \begin{pmatrix} 1 + v \\ \xi \end{pmatrix} , \quad (2)$$

where v is a random variable drawn from the Gaussian distribution with the zero mean and the variance ε and ξ is a random variable of the Gaussian distribution with the zero mean and the variance τ . A compact form of (2) is

$$\mathbf{b} = \mathbf{M}\mathbf{a} + \mathbf{s} . \quad (3)$$

To apply (3) for an arbitrarily oriented segment, the coordinate system can be rotated by the matrix \mathbf{R} and then back by the matrix \mathbf{R}^T . Thus, (3) can be rewritten as follows

$$\mathbf{b} = \mathbf{R}^T \mathbf{M} \mathbf{R} \mathbf{a} + \mathbf{R}^T \mathbf{s} = \mathbf{N} \mathbf{a} + \mathbf{R}^T \mathbf{s} . \quad (4)$$

Using (4), the robot position at the end of the segment can be computed from its starting position. However, the absolute position does not concern us, to show the navigation stability we need to describe and predict the robot position uncertainty.

2.2 Position Uncertainty

Let the robot position \mathbf{a} be a random variable drawn from a two-dimensional normal distribution with the mean $\hat{\mathbf{a}}$ and the covariance matrix \mathbf{A} . Equation (4) has only linear and absolute terms, and therefore at the segment end the position \mathbf{b} will constitute a normal distribution with a covariance matrix \mathbf{B} . Denoting $\mathbf{a} = \hat{\mathbf{a}} + \tilde{\mathbf{a}}$, where $\tilde{\mathbf{a}}$ is a random variable of a normal distribution with the zero mean and the covariance \mathbf{A} and using a similar notation for \mathbf{b} , equation (4) can be rewritten to

$$\tilde{\mathbf{b}} = \mathbf{N} \tilde{\mathbf{a}} + \mathbf{R}^T \tilde{\mathbf{s}} .$$

Assuming \mathbf{s} and $\tilde{\mathbf{a}}$ are independent and do not correlate,

$$\tilde{\mathbf{b}} \tilde{\mathbf{b}}^T = \mathbf{N} \tilde{\mathbf{a}} \tilde{\mathbf{a}}^T \mathbf{N}^T + \mathbf{R}^T \tilde{\mathbf{s}} \tilde{\mathbf{s}}^T \mathbf{R} ,$$

which rewritten in terms of covariance matrices is

$$\mathbf{B} = \mathbf{N} \mathbf{A} \mathbf{N}^T + \mathbf{R}^T \mathbf{S} \mathbf{R} = \mathbf{N} \mathbf{A} \mathbf{N}^T + \mathbf{T} , \quad (5)$$

where $T=R^T S R$. Equation (5) allows determination of the robot position uncertainty after traversing one segment. A geometrical interpretation of the algebraic terms describing the uncertainty evolution is shown in Fig. 2.

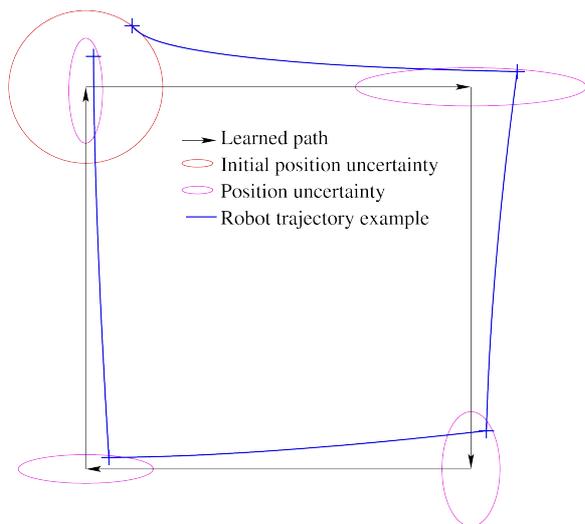


Fig. 2 Position uncertainty evolution for a simple symmetric path.

2.3 Traversing multiple segments

Let the robot path is closed and consists of n chained segments denoted by $i \in \{0, \dots, n-1\}$. A segment i is oriented in the direction α_i and its length is s_i . The robot positions at the start and end of the i^{th} segment are \mathbf{a}_i and \mathbf{b}_i . The segments are joined, so $\mathbf{b}_i = \mathbf{a}_{i+1}$ and the movement model (5) for the i^{th} traveled segment is

$$A_{i+1} = B_i = N_i A_i N_i^T + T_i .$$

The robot position uncertainty after traversing whole path consisting of the n segments will be

$$A_n = \dot{N} A_0 \dot{N}^T + \dot{T} ,$$

where

$$\dot{N} = \prod_{j=0}^{n-1} N_j ,$$

and

$$\dot{T} = \sum_{j=0}^{n-1} \left(\left(\prod_{k=n-1}^j N_k \right) N_j^{-1} T_j (N_j^T)^{-1} \left(\prod_{k=j}^{n-1} N_k^T \right) \right) .$$

If the robot traverses the entire path k -times, its position uncertainty can be computed in a recursive way by

$$A_{(k+1)n} = \dot{N} A_{kn} \dot{N}^T + \dot{T} . \quad (6)$$

Since (6) is the Lyapunov discrete equation, its limit for $k \rightarrow +\infty$ is finite, if all eigenvalues of \mathbf{N} lie within a unit circle and \mathbf{T} is symmetric. Since \mathbf{S}_k is symmetric and a product $\mathbf{X S}_k \mathbf{X}^T$ is symmetric for any \mathbf{X} and addition preserves symmetry, the matrix \mathbf{T} is always symmetric. Since the maximal eigenvalues of \mathbf{M}_i are equal to 1, the eigenvalues of \mathbf{N} cannot be greater than 1. Moreover, the eigenvalues of \mathbf{N} are equal to 1 only if all matrices \mathbf{R}_i are the same [3], i.e. all azimuths α_i are the same. This means, that if the robot travels repeatedly a path consisting of segments with different azimuths, its position uncertainty will not diverge.

3. Navigation System

Our system runs on the P3AT robot equipped with the Unibrain Fire-i601c camera, the TCM2 compass and the HP 8710p laptop. The robot camera is aimed forwards and provides color images with resolution of 1024x768 pixels and field of view approximately 60 degrees.

The system implements two navigation algorithms based on image processing that follow the principles described in Section 1. The first algorithm, called "GeNav" [2], recognizes a pathway in front of the robot and corrects the robot heading to keep it in the middle of the recognized path. Although quite simple, fast and reliable, its main disadvantage is its reliance on the environment structure. It can be used only in areas, where pathways are clearly distinguishable. The second algorithm is called "SURFNav" [3] and it is based on a salient feature recognition [5]. While robust and reliable, salient feature extraction requires a significant amount of processing power, therefore a GPU based feature extraction algorithm is used [6]. Both algorithms require a prior knowledge about the environment. While GeNav needs a pathway color, SURFNav requires a detailed map of salient objects around the path being traversed. Therefore both algorithms require a suitable map of the operational environment. The map is created by the robot that is manually driven by a human operator through the environment prior to the competition. Alternatively, a map created by another robot or publicly available maps like [7, 8] can be used.

The map consists of a set of straight line segments. Each segment is described by the initial robot orientation α , the segment length s , the landmark set L and the color table \mathbf{G} . The set L consists of salient features detected in images captured by the robot's forward looking camera. The color table \mathbf{G} is a mapping of the RGB color space to N^+ indicating the likelihood of a pixel being on the path. Once the map is created, the robot can travel autonomously within the mapped environment using the algorithms. GeNav recognizes the path in front of the robot based on the color table \mathbf{G} . SurfNav computes steering by matching the currently detected landmarks to the landmark set L . Both algorithms decide only the robot steering, and the robot forward speed is set according to odometry measurements. A simple, sonar-based obstacle avoidance routine based on the Tangent Bug method [9] can temporarily override steering and forward speeds set by both GeNav and SURFNav algorithms in cases of detected obstacles in the robot course.

3.1 The Mapping Phase

The mapping procedure is initiated by a human operator. Before the robot starts to learn the segment, it reads compass data to establish the segment azimuth α and resets its odometric counters. After that, the robot starts to move forwards while measuring the traveled distance and processing the onboard camera image.

The onboard camera image is processed by two independent algorithms. Since its lower half contains the path on which the robot moves, a small trapezoidal area at the bottom of the image is used to update the color table \mathbf{G} . The table \mathbf{G} is implemented as a three dimensional array. Each cell of \mathbf{G} represents a color in the RGB color space and contains an integer value. If a pixel of a particular color is detected in the trapezoidal area, the corresponding cell value is increased. At the end of the mapping phase, the \mathbf{G} contains a color histogram of the pathway.

The upper half of the image is processed by the SURF [5] algorithm, which provides a set of point features. Each

feature is described by its position in the image and a descriptor invariant to lighting and viewpoint changes. These features are tracked as the robot moves. If a feature is tracked long enough, it is saved in the landmark set L along with additional attributes. The attributes are the number of images, in which the landmark was detected, feature position in the image and the robot position as a distance from the current segment start, when the feature tracking was initiated and finished.

The feature tracking can be described in terms of manipulating three sets of the features: currently detected features S , tracked features T and saved features L . At first, features extracted from the current image are put to S . Then, similarity between the features in the sets T and S are computed based on the Euclidean distance of their descriptors. For each currently tracked feature, two most similar features from S are found. If these two pairs are distinguishable [5] the best matching feature is removed from S and the tracked landmark description is updated. If the two pairs are not distinguishable, the landmark is moved from the set T to the set L . After all the tracked landmarks have been updated or removed, the remaining features of S are moved to T . When the mapping is completed, each feature in the set L is described by its descriptor, position in the image and values of the robot odometric counter in moments of the first and last time the feature was tracked. The segment description consisting of the azimuth α , the length s and the set L is saved at the end of the segment and the operator can turn the robot to another direction and initiate mapping of a new segment.

3.2 Navigation Phase

In the autonomous navigation mode, the robot is supposed to traverse a sequence of mapped segments. The operator has to place the robot at the start of the first segment and indicate, which segments should be traversed by which algorithm. Then, the robot loads description of the first segment, turns in the direction of the segment azimuth and starts to move forwards. Its forward speed is set according to the travelled distance until the odometric counter indicates that the segment length has been traversed. The robot steering speed is controlled by either GeNav or SURFNav algorithms. After the segment is traversed, the robot loads description of the next segment and repeats the navigation procedure.

3.3 Pathway Recognition - GeNav

The bottom half of image is analyzed by GeNav. In particular, it recognizes a pathway in the image and determines the robot steering speed to keep it in the middle of the detected pathway. The algorithm uses the color table G to decide which pixels lies on the path and works as follows.

The GeNav algorithm starts with the bottom row of the acquired image. It searches for pixels of the path color and determines the mean value of their horizontal coordinates. After that, the algorithm searches for the path boundaries. It starts from the mean position computed in the previous step and searches for pixels with other than path color in both directions from the mean. The path center and width are then calculated out of the detected boundaries for the particular row, see Fig. 3. If the width is greater than a predefined threshold, the algorithm proceeds to a higher row. The search algorithm is completed when the current path width drops below the threshold or when it reaches the middle image row. The robot steering speed ω is then determined from the mean of the computed path centers.



Fig. 3 The image processed by both algorithms.

3.4 Landmark-based Navigation - SURFNav

SurfNav analyzes the top half of the onboard camera image. The robot uses the landmark set L and the currently detected features S to determine the robot steering speed. A set of landmarks T , which are expected to be seen at the current position, is selected from the set of the learned landmarks L . The selection is based on the robot distance from the segment start, which is measured solely by the odometry. For each landmark in the set T , the best matching feature in the set of currently detected landmarks S are found in the same way as in the mapping phase. A difference in horizontal image coordinates of the features is computed for each such tuple. Each difference is then stored in a set H . The modus of the set H is then estimated by a histogram voting method. The modus is then used to compute the robot steering speed ω . The current on-board camera image, positions of the detected and expected features, established correspondences and histogram are logged and displayed on a GUI, so the operator can detect potential error states and eventually stop the robot. A detailed description of the algorithm is given in [3].

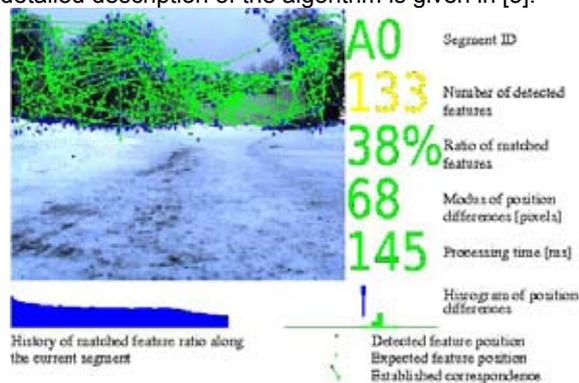


Fig. 4 Robot GUI in the navigation phase

3.5 Initial position estimation

In the RoboTour 2009 competition, the robots had to travel a circular path and the starting position of each team was different. The teams could give the robot the path description, but not the initial position. Therefore, the robot had to determine its starting position on its own.

To cope with the problem of initial position estimation, our system uses combination of GPS, compass and monocular camera-based position estimation. We assume, that the robot is positioned on the path and is headed in the direction of the current segment. In the first step, the compass and GPS are used to determine the segment on which the robot stands. In the second step, the robot uses its camera to determine its distance from the segment start as follows.

The robot creates several hypotheses (10 hypotheses per meter of the segment) about the distance, runs the SURFNav algorithm and stores the set of horizontal differences H of each of the hypotheses. Then, the robot computes the standard deviation σ_H of the each set H . The hypothesis with the lowest σ_H is then chosen and its landmark set T is retrieved. The fundamental matrix F matching the sets T and S is then established by the RANSAC algorithm. The matrix F is then factorized to obtain the orientation and position. If the computed robot orientation is close to zero, the hypothesis is accepted. Otherwise, the fundamental matrix is computed for the next best (i.e. with lowest σ_H) hypothesis.

4. Experiments

The practical verification of our navigation system has been examined in a series of outdoor experiments with a P3AT mobile robotic platform.

The first set of experiments was aimed at verification of the navigation system and measuring its precision. The experiments have proved, that the proposed method is able to cope with diverse terrain, dynamic objects, obstacles, systematic errors, variable lighting conditions and seasonal environment changes. During these experiments, the robot has autonomously traversed over 25 km with an average position error lower than 0.3 m [3].

In the second set of experiments, we have tried to build the map from the Google Street view data. The parameters of the Google Street view images were set up to resemble the image a robot would see. After that, the SURF features have been extracted from these images and a landmark map was created. The experiment showed, that a mobile robot can use this landmark map for navigation in an urban environment. However, the SURFNav algorithm has to be complemented by a collision avoidance.

Participating at the RoboTour competitions can be considered as an experiment as well. Contrary to the regular field tests, the competition is more challenging, because all system components must work flawlessly. So, the RoboTour event is not only the navigational method examination, but a test of the whole system. Our navigation method evolved during the time when we started with it in 2006. In that year, we used only the path following algorithm and we had problems with crossing recognition. A year later, our system was complete, but contained a lot of software bugs. The main milestone was made in 2008 and from that time we consider the navigational system complete, however it still required improvements in particular aspects. In years 2008 and 2009, most of the software bugs were resolved. Even through the performance has not been perfect, the robot was able to travel the required trajectories and our team has reached the first rank for both events in 2008 and 2009.

5. Conclusion

A simple navigation system based on bearing-only sensors and an odometry used in the RoboTour competitions was presented in this paper. The method navigates a robot using a map of the environment and a camera input to establish the robot heading, while the traveled distance is measured by the odometry. Using a mathematical model of the navigation method, we have shown that this kind of navigation is sufficient to keep the robot position error limited. Besides, the proposed method has been experimentally verified by a mobile robot with a monocular camera.

Even through our success in the last two RoboTour competitions, the main disadvantage of our method is in the necessity of the detailed map in advance. The robot can create the map in a guided tour, however it takes a long time, because of its low speed. The mapping of the Stromovka and Lužánky parks took two, resp. four days. Our intention is to solve this issue while preserving the reliability of the method.

Acknowledgements

The work presented in this paper has been supported by the Grant Agency of the Czech Technical University in Prague, grant No. SGS10/185, by the Ministry of Education of the Czech Republic under the projects MSM 6840770038 and 7E08006 and by the EU under the project No. 216240.

References

- [1] THORPE, C., HEBERT, M., KANADE, T., SHAFER, S.: "Vision and navigation for the Carnegie-Mellon Navlab," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 10, no. 3, p. 362 – 373, May 1988.
- [2] KOŠNAR, K., KRAJNÍK, T., PŘEUČIL, L.: "Visual Topological Mapping," in European Robotics Symposium 2008. Heidelberg: Springer, 2008, p. 333-342. ISBN 978-3-540-78315-2.
- [3] KRAJNÍK, T., FAIGL, J., VONÁSEK, V., KULICH, M., KOŠNAR, K., PŘEUČIL, L.: "Simple yet stable bearing-only navigation," Journal of Field Robotics, 2010/27, p. 511–533. doi: 10.1002/rob.20354
- [4] VALGREN, C., LILIENTHAL, A.J.: "SIFT, SURF and seasons: Long-term outdoor localization using local features," in Proceedings of the European Conference on Mobile Robots (ECMR), September 2007.
- [5] BAY, H., ESS, A., TUYTELAARS, T., GOOL, L.V.: "Speeded-up robust features (SURF)," Computer Vision and Image Understanding, Vol. 110, No. 3, p. 346-359, 2008.
- [6] CORNELIS, N., GOOL, L.V.: "Fast scale invariant feature detection and matching on programmable graphics hardware," in CVPR 2008 Workshop (June 27th), Anchorage, Alaska, June 2008.
- [7] ANGUELOV D., ET AL.: "Google Street View: Capturing the world at street level," Computer, vol. 43, no. 6, pp. 32–38, June 2010.
- [8] HAKLAY, M.M., WEBER, P.: "Openstreetmap: User-generated street maps," IEEE Pervasive Computing, vol. 7, no. 4, pp. 12–18, 2008.
- [9] CHOSET, H., ET AL.: "Principles of Robot Motion: Theory, Algorithms, and Implementations." Cambridge, MA: MIT Press, June 2005.

Ing.Tomáš Krajník, Ing.Jan Faigl, Ing.Vojtěch Vonásek, Bc.Hana Szúcsová, Bc.Ondřej Fišer, Ing.Libor Přeučil,CSc.

Czech Technical University in Prague
Faculty of Electrical Engineering, Department of Cybernetics
Technická 2
166 27 Prague 6
Email: {krajnik,xfai,gl,vonasek,szucsova,fisero,preucil}@labe.felk.cvut.cz

Teaching about humanoids in a robotics course on computer engineering studies

Martin Mellado

Abstract

In this paper, the author presents his personal experience on teaching robotics, and more specifically on teaching humanoid robotics, within the studies for the Computer Engineering degree in the Escuela Técnica Superior de Ingeniería Informática of the Universidad Politécnica de Valencia, Spain. In the paper, first of all there is an introduction to the topic and how is the situation of robotics courses within computer engineering degrees in the more significant centres in Europe and Spain. Then, the paper progresses to explain how is the situation in Valencia, where there is a Robotics Course and a Robot Laboratory Project, explaining their main characteristics, contents and progress plan. The theoretical content for the unit on humanoid robotics is explained. The paper goes in details on the available equipment and the content of the laboratory sessions, the knowledge acquired by students and the exercises they have to do in order to pass this part of the course. Main issues covered on the Robot Laboratory Project are development of walking procedures for humanoid robots, the sensor control and the robot navigation in mazes. A student contest is organized so that the different student groups can show their abilities to program specific robot tasks, such as races and going up and down stairs and ramps. Last but not least, the paper will show the conclusions on this teaching experience on robot humanoids.

Keywords: robotics; humanoid robots; teaching robotics; robotics in education

Introduction

In the last years and currently, humanoid robotics is one of the most difficult but popular topics in robotic research. Last advances on this field have produced promising results such as the Honda Asimo, the Sony Qrio and the AIST's HRP-2 and HRP-4. Every year there are more international conferences which include this topic, and specific conference for this topic, such as the IEEE-RAS International Conference on Humanoid Robots [1], which shows the late advances on this field.

In opposition to this recent interest on research and diffusion on humanoid robotics, it seems that teaching on this field is not progressing according to the repercussion of efforts done in researching. The teaching in humanoid robotics is mainly focused for post-graduate programs such as master and doctor courses and/or degrees. An example can be found in the 6th International UJI Robotics School IURS-2006 "Humanoid Robots" [2]. There are some other research courses on humanoid robotics, such as [3] in Carnegie Mellon University and [4] in University of Southern California.

Robot contests, such as RoboCup [5], are growing all around the world. During this year, the most significant robot competition at the European level has been the RoboCup Mediterranean Open, RomeCup [6], with different contests. One of the most significant one is the Football (Soccer) competition with standard platform league (then Nao robot of Aldebaran [7]) which has been won in 2010 [8] by the Spanish team Los Hídalgos of Instituto de Automática e Informática Industrial of Universidad Politécnica de Valencia in cooperation with the Universidad de Murcia.

1. The Studies on Robotics in Computer Engineering Studies

1.1 Robotics in computer engineering studies

A review of the most representative university centres (schools and faculties) around Europe imparting the degree of Computer Engineering has been done in order to know the importance of robotics teaching in the most significant centres. The centres have been selected according to the Academic Ranking of World Universities in Computer Science – 2009 [9] to select the three most significant centres in Europe and in Spain.

For Europe, none of the best three centres according to this ranking (in Oxford [10], Zurich [11] and Cambridge [12]) has any course related to robotics in the studies of Computer Science BA degree.

Related to the robotics courses in Spain, the analysis has been done considering the three most representative Spanish schools and faculties imparting the degree of Computer Engineering, which are, together with the ETSInf that will be commented next section:

- Facultat D'Informàtica de Barcelona, Universitat Politècnica de Catalunya [13]. In the new studies, there is a robotics course of 75hours including industrial robots and mobile robots [14].
- Facultad de Informática, Universidad Politécnica de Madrid [15]. There is no course on robotics in the studies for the degree in Computer Engineering. It will be

included a course on autonomous robots in the Master Program.

In centres as in Escuela Politécnica Superior, Universidad Carlos III of Madrid [16] or Escuela Politécnica Superior, Universidad de Almería [17] there are robotics courses now, but in both cases, these courses will disappear in future years for the new degrees adjusted to European Education Space. In Universitat Pompeu Fabra [18] there will be a course on robotics in the future when there is no one now.

From this study, it can be stated that, even considering that it is clear the important role of computer engineers in the development of humanoid robotics in the future, there is a lack of formation in this field.

1.2 Studies on Robotics in the ETSInf, the school of computer engineering in the Universidad Politécnica de Valencia

The School of Computer Engineering (Escuela Técnica Superior de Ingeniería Informática, ETSInf) is the result of the integration into one single institution of the Higher Technical School of Applied Computing and the Faculty of Computer Science due to the Bologna process. The ETSInf was officially created on February 27th 2009. This School inherits a long tradition of teaching and research in Computing that goes back to 1982.

The degree programmes taught at the School of Computer Science are:

- Computer Engineering (5-years programme).
- Computer Technical Engineering (3-year programmes).
- Library and Information Management (2-year programme as second stage).
- Bachelor degree in Computer Engineering (4-year programme).

The Computer Engineer undergoes extensive training in all computer-related areas. This solid training allows graduates to fit easily into different professional careers and to efficiently manage the ever-changing technological advances in the field.

In the last year of the degree course students can choose between some specialist pathways. One of them, Industrial Computing offers a complementary teaching in the application of computer engineering to the industrial systems and processes, including aspects such as real-time systems in industries, robotics, computer aided design and manufacture, automation and control of industrial processes, industrial instrumentation and computer networks, computer vision and digital image processing.

This general objective is implemented through the courses offered in the context of the pathway, which are distributed so that the student must be enrolled according to the scheme shown in Fig. 1.

From this structure it can be noted that there are two courses related to robotics, Robotics Course and Robot Laboratory Project, which will be introduced in the next section.

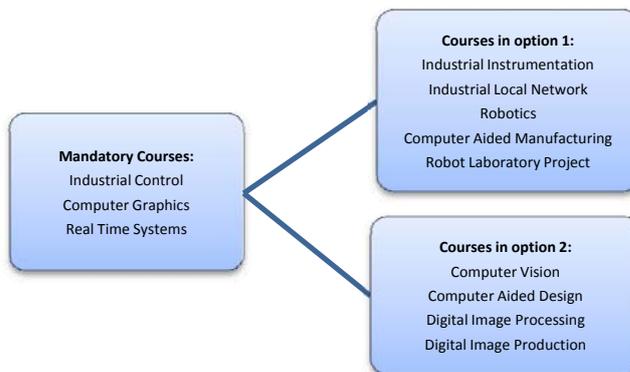


Fig. 1 Structure of the Industrial Computing pathway

2. Robotics Course and Robot Laboratory Project

2.1 Robotics course

The objectives of the robotics course are:

- To introduce the basic concepts of robotic systems in the platforms: industrial robot manipulators, mobile robots and humanoids.
- To learn the basic programming of each of the different types of robots.

The content of the course is according to the following program defined in parts and units:

- Part 1. Introduction to robotics
 - Unit 1. Basic principles of robotics
 - Unit 2. Spatial relations in robotics
- Part 2. Articulated robots
 - Unit 3. Industrial robot manipulators
 - Unit 4. Robot programming
- Part 3. Mobile robots and humanoids
 - Unit 5. Mobile robots
 - Unit 6. Humanoid robots

To fulfil the objectives, laboratory work is organized according to:

- Introduction to robotics:
 - Use of a robot simulation software
 - Spatial relations in robotics
- Industrial robot manipulators:
 - Programming simulation of robots
 - Industrial robot programming
- Mobile robots and humanoids:
 - Applications on mobile robots
 - Applications on humanoid robots

The robotics course has a total of 4.5 Spanish credits¹ meaning a total of 45 attending hours. The distribution of these hours is shown in Table 1.

Didactic Part	Attending Hours	External Hours ²
Introduction to robotics	13	5
Industrial robot manipulators	20	20
Mobile robots and humanoids	12	5
Total hours	45	30

Tab.1 Hour distribution of Robotics course

¹ Conversion: 1 ECTS = 1.25 Spanish credits; 1 Spanish credit = 0.8 ECTS

² Estimated hours dedicated by each student

The course evaluation is made using the following methods:

- Course project. It is a teaching strategy in which students develop a new and unique product by progressing with a series of tasks looking for effective use of resources.
- Online written test with open answer. Time trial, via web, in which the student builds his/her response. Students can use any material support.
- Laboratory tests. A short practical exercise that students must fill at the end of a laboratory session.

The final mark is obtained with a weight addition: 70% with the project coursework and 30% in continuous evaluation (on-line tests and laboratory tests).

All the Robotics Course teaching material is available in an OpenCourseWare website [19]. OpenCourseWare (OCW) is a web-based publication of course contents. OCW is open and available to the world and is a permanent activity. Its origin came from MIT [20] but nowadays is spreading through the world. In the case of Spain, OCW is managed by Universia [21].

2.2 Robot Laboratory Project

The objective of the Robot Laboratory Project is to develop computer projects in the field of the material studied in the Robotics Course with the integration of concepts acquired in other courses studied, mainly within the Industrial Computing pathway.

With this objective, there is only one learning unit dedicated to the development of practical computer projects in the field of robotics.

Different projects are offered to the students, as for example:

- An automation project with the use of the industrial robot and auxiliary devices (conveyor tracking, rotation table, ...).
- A project for mobile robots, in order to generate a program to solve a maze or to generate sweeping trajectories on a small room
- A project for humanoid robots, as will be explained in the next section.

The students choose one project and work on it during the semester. All hours in Robot Laboratory Project are dedicated to the practical development of the projects, with 60 attending hours and 20 external hours (estimated).

The evaluation of Robot Laboratory Project is made using a team project, developed during the semester.

2.3 Robotics teaching resources

Resources available for the laboratory sessions of the Robotics Course and for the project work on the Robot Laboratory Project are:

- Robotics Laboratory, with an industrial robot (ABB IRB 140), two mobile minirobots (Khepera-II) and 11 humanoid robots (Robonova-1).
- Software for the laboratory sessions: MS Visual Studio 2008 for C++ programming, VirtualRobot software for robot simulation and programming, EditRapid for ABB robot programming using Rapid language and RoboBasic v2.5 for Robonova-1 programming.

3. Teaching Humanoids

3.1 Teaching Theoretical Concepts on Humanoids

As it was seen in the contents of the Robotics Course, there is a unit for humanoid robots. The objectives of this unit are:

- To understand the basic characteristics of humanoid robots and their possible applications
- To learn the basic methods for humanoid motion control and its problems
- To understand the possibilities of humanoid minirobots
- To understand the development of a specific case of humanoid minirobot

The contents of this unit are:

- Introduction. This part covers the definition humanoid robots, their evolution compared to human evolution, differences between humanoid & other robots, social aspects to be considered, current problems and working fields, ...
- Applications, detailing possible service applications but also including industrial ones
- Motion control. In this part, the problem of stability in humanoid robots is introduced, and possible kinematics models are basically introduced with some examples. Walking strategies for stride execution and methods to capture human motion and their possible applications to humanoid robots are also explained in this point
- Humanoid minirobots. Within this part, it is explained the following issues: the origin of minirobots, several commercial humanoid minirobots, the RoboCup competition and an example product developed at our university, microbiro, with its hardware and software architecture and main feasibilities.

3.2 Equipment for laboratory sessions on humanoids

For laboratory sessions we have available a total of 11 Robonova-1 humanoid minirobots (Fig. 2). Hitec's Robonova-1 is a fully articulated, 12" high humanoid robot, which includes a HSR-8498HB digital servomotor in every joint.

Robonova-1 kinematics has 5 joints for each leg and 3 joints for each arm, giving a total of 16 joints moved via servos. These servos can be programmed developing users' programs in RoboBasic language with the development tool RoboBasic. Programs are downloaded into the robot controller Micom board MR-C3024 through a RS232 cable.

Servos can be modified in a range of degrees (from 10° to 190°), although some joints have a smaller range (for example, the ankle or the knee) because of physical constraints. The position of joints can be defined with program sentences, and changing angles of motors in a certain way, the robot can make several movements like walking, running, dancing, etc.



Fig. 2 Some of the 11 Robonova-1 robots used in the laboratory work.

In addition, the robot controller can manage some sensors (proximity, inclination...) and the data obtained with these sensors can be evaluated within RoboBasic programs running on the controller. For example, we can write a program with an 'if then else' sentence that depends on a variable whose value is the inclination of the robot. Depending on the position of the robot, we could execute the correct series of sentences to stand up the robot (if it is face up or face down).

Every Robonova-1 used in the course includes the following sensors:

- An infrared proximity sensor on its chest
- An infrared proximity sensor on each of its arms
- A tilt (inclination sensor) in its back
- An IR LED on its head to receive remote control orders.

Robonova-1 kits come with a remote control. Robot programs can get information from the remote control (for example, which button has been pressed) and use this data in the code as if it was another sensor. In this way, different programs can be run or robot motions and actions can be modified according to user's actions in remote control.

Robonova-1 is powered with a 5-cell NiMH rechargeable battery. In this course, for every Robonova-1 there are two batteries in order to be able to use the robot with a battery while the other one is recharging. Notice that for a full charge of the battery, it has to be plugged almost two hours, and then it gives about one hour of working time with the robot. Obviously, these times are approximated and depend on robot motions. Hence, battery save is very important, mainly considering that when battery is not fully charged, the robot can work in a wrong way.

RoboBasic is an exclusive BASIC extended programming language designed for controlling humanoid robots. With RoboBasic, commands that are needed to control a robot have been added to the general BASIC programming language. Because the grammar of RoboBasic is based on the general BASIC programming language, most of RoboBasic is similar to or the same as BASIC. In order to develop programs and download them on the robot, a development program, also called RoboBasic (v2.5) is provided with Robonova-1.

3.3 Laboratory sessions on Humanoids

The first two laboratory sessions within the Robotics Course intend to be a starting point for humanoids practical work with the use of a humanoid robot Robonova-1. There is a specific tutorial [22] prepared to introduce the robots to the students (as users and programmers) who never had a previous contact with this robot or its programming language RoboBasic. The hardware of Robonova-1 and some basic programming guides to start moving the robot are described at the tutorial.

Students are grouped in couples, so the maximum numbers of student in a session are 20 (only 10 Robonova-1 robots are used so that there is one extra for demonstrations). A very simple first exercise allows students to start working with the system, software and hardware and to try the connection of the robot to the computer and to verify their communication. The students have to program a task to control the light that is on the robot head. The students learn then how to use the RoboBasic system, including the steps to introduce a program, to compile it, to edit program errors, to download the program in the robot and to execute it.

Next step is to program movements. The students begin with the simplest way to make a sequence of movements: moving the robot manually to different positions, memorize them and play them. RoboBasic has facilities to do this and

the students soon are moving the robot to different configurations.

After verifying the previous example, in the sessions some exercises related to robot motions and postures are given for the evaluation of the laboratory sessions. Possible exercises are:

- Keep in balance on one leg.
- Step forward.
- Step lateral (right or left).
- Step backward.
- Roll 45° around robot position (right or left).

During these laboratory sessions, the students have a close contact with the robots and learn mainly the stability problems that exist to get a proper motion control on humanoid robots (Fig. 3).



Fig. 3 Two students working with the Robonova-1 miniobot.

3.4 Projects on Humanoids

After the previously explained laboratory sessions, the students are ready to develop their own project on humanoids in the Robot Laboratory Project. In the academic course 2009/10, six students of 15 have chosen to develop their project on humanoid robotics.

Every pair of students is assigned a robot through the semester so that their developments are specific for this robot. They start their project with the common goal of developing walking procedures for the humanoid robot.

Then they start controlling the sensors on the robot, first with the tilt sensor. A program must be done so that the robot control its inclination angle and move its arm with opposite angle, so that the arm keeps always in vertical status. Note that in this way, the program is using a servo motion as an output indicator of a value.

The next step is to control the three infrared reflectance sensors for distance computation. From the sensor specifications, the students must compute an approximate value of the distance from the value read for the sensor. Calibration is a critical issue in this problem.

The project is organized in a contest with the following four trials:

- A race for going up and down a stairs
- A race for going up and down ramps
- A race avoiding obstacles in a simple maze
- An open trial to demonstrate some robot programming abilities

The definition of the first three trials is shown in Fig 4. An event for the contest has been organized this year in May [23], with an attendance of more than 75 students of the

degree to watch the trials. Some pictures are shown on Fig. 5 and Fig. 6.

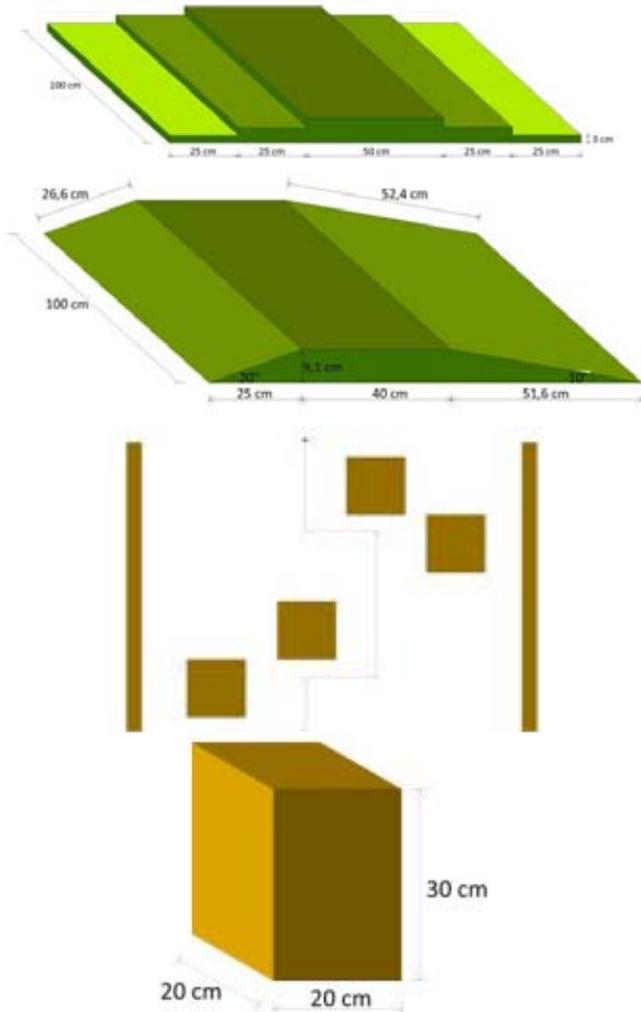


Fig. 4 The definition of the stairs, ramps and maze trials.



Fig. 5 A robot on the stair contest trial.



Fig. 6 A robot on the ramp contest trial.

Conclusions

Humanoid robotics is one of the most promising research topics in a close future for science and technology. Computer engineers have a crucial work to do in this field as robots must be programmed and controlled. Nevertheless, not many universities are including robotics in their Computer Engineering degrees. This paper shows an experience of teaching humanoid robotics in this studies, explaining in detail how is organized a course and a laboratory project.

Acknowledgements

This work has been partially supported by the Escuela Técnica Superior de Ingeniería Informática (ETSIInf) de la Universidad Politécnica de Valencia.

References

- [1] 9th IEEE-RAS International Conference on Humanoid Robots (Humanoids09), UPMC, Paris, December 7-10, 2009. www.humanoids2009.org/
- [2] Summer School on Humanoid Robots, Universitat Jaume I. Available online: www.robot.uji.es/lab/plone/events/iurs06
- [3] Course 16-264 on Humanoids, Carnegie Mellon University. Available online: www.cs.cmu.edu/~cga/humanoids-ugrad/description.html
- [4] Course on Perceiving and Controlling Humanoid Behavior, University of South California. Available online: robotics.usc.edu/~maja/teaching/cs599-hum.html
- [5] Robocup competition. www.robocup.org
- [6] Robocup Mediterranean Open. www.robocup-mediterranean-open.org
- [7] Aldebaran Robotics. www.aldebaran-robotics.com
- [8] RomeCup 2010: premiati i vincitori in Campidoglio. Available on line: https://intranet.ai2.upv.es/alfresco/webdownload.jsp?docURL=https://intranet.ai2.upv.es/alfresco/d/a/workspace/SpacesStore/fdc9f7cf-e929-45b2-a402-16fcd4af98e7/CS_RomeCup2010_vincitori.pdf
- [9] Academic Ranking of World Universities in Computer Science. Shanghai Ranking Consultancy, 2009. Available online: www.arwu.org/SubjectCS2009.jsp
- [10] Computing Laboratory, University of Oxford, www.ox.ac.uk/
- [11] Department of Computer Science, Eidgenössische Technische Hochschule, Zurich, www.ethz.ch
- [12] Computer Laboratory, University of Cambridge, www.cam.ac.uk

[13] Facultat d'Informàtica Barcelona, Universitat Politècnica de Catalunya. www.fib.upc.edu

[14] Grado de ingeniería en informática. Available on line: www.fib.upc.edu/es/grau.html

[15] Facultad de Informática, Universidad Politécnica de Madrid. www.fi.upm.es

[16] Escuela Politécnica Superior, Universidad Carlos III de Madrid. www.uc3m.es/portal/page/portal/escu_polit_sup

[17] Escuela Técnica Superior de Ingeniería, Universidad de Almería. cms.ual.es/UAL/universidad/centros/politecnica/index.htm

[18] Escola Superior Politècnica, Universitat Pompeu Fabra. www.upf.edu/esup/

[19] Asignatura OCW Robótica. Available on line: www.upv.es/ocwasi/2009/7166

[20] MITopenCourseware. ocw.mit.edu/index.htm

[21] UCW universia. ocw.universia.net/es/

[22] M. MELLADO, Programming a humanoid robot. Available in [19]. December 2009

[23] Los robots toman la ETSINF. Available online: tv.inf.upv.es

Dr. Martin Mellado

Instituto de Automática e Informática Industrial
Universidad Politécnica de Valencia
Camino de Vera s/n
46022 Valencia, Spain
martin@ai2.upv.es
<http://www.ai2.upv.es>

Robotour Solution as a Learned Behavior Based on Artificial Neural Networks

Miroslav Nadhajský and Pavel Petrovič

Abstract

Our contribution describes a mobile robot platform that has been built for the purpose of the contest Robotour – robotika.cz outdoor delivery challenge. The robot is a standard differential-drive robot with a good quality consumer market digital video camera with a lightweight, but high-performance laptop computer used as the main control board. Supplementary board is used to control motors and sensors of the robot. The robot utilizes a behavior-based architecture and its vision module that is responsible for track-following is utilizing an artificial neural network that was trained on a set of images. This is a novel solution that has not been used in Robotour contest previously, and our early experiments demonstrate promising results.

Keywords: robotour, navigation, artificial neural networks, learning robots

Introduction

Applications of robotics technology in both production and personal use are becoming possible with the development of new materials, motors, sensors and vision, ever decreasing cost of computing and memory capacity, and development of new algorithms and control strategies. Robots must be able to operate in dynamic and unpredictable environments. Therefore, one of the most important challenges to be solved reliably is robot navigation – in both indoor and outdoor environments. The robots must be able to localize themselves on a supplied map, create their own map representations of the explored environment, and they must be able to navigate their environments safely, without colliding with obstacles, or failing to follow the paths, roads, trails, and tracks. The real improvements in the technology typically occur when there is a large motivational pressure to produce a working solution. This might either be a goal to produce a final product, or alternately, with somewhat more relaxed requirements and settings, which are suitable for experimentation, and research, when the goal is to develop a robot to participate in a robotics contest.

Robotour – robotika.cz outdoor delivery challenge, organized by the Czech association robotika.cz, is an annual meeting of teams building and/or programming outdoor robots that navigate in a city park filled with trails, trees, grass, benches, statues, water ponds, bridges, and people. The task changes every year, but the main challenges are 1) be able to localize and navigate on a map supplied by the organizers, and 2) be able to follow the trails and paths without colliding with the obstacles or leaving the path without reaching the goal. See [1] for the exact rules of this year's contest.

Various solutions for the challenge were developed, however, in most cases, they did not take advantage of advanced artificial intelligence algorithms. In particular, only few different vision algorithms were developed until today, several teams shared the successful solution of [2], and many solutions rely on the use of odometry, compass, and

GPS. We would like to address this area, and prepare a solution for the contest in 2010 or 2011 that will utilize AI algorithms. The second author has participated in the competition team several times in the past, and collected some experience and motivation for a new attempt. In this article, we describe the principles our solution is based on and is currently being built. In the following sections, we describe the mechanics and the hardware, robot overall architecture, the software components, and the AI methods that we aim to use. Finally we summarize the experience with building and programming the robot up to date.

1. Mechanics

The robot is a simple robot with differential-drive kinematics with one supporting free-rolling caster wheel. The length of the sides of its square base is 45 cm; the air-inflated wheels of a diameter 15.3 cm are mounted on the outside of the base, in the front of the robot. The total weight is about 6 kg without any load. The robot provides a storage space of ca. 20 x 20 x 45 cm to carry a heavy load (approx. 5 kg), which can be placed close to the center of rotation, above the propelled wheels, so that it does not have a negative impact on maneuverability of the robot. The main control unit is a portable computer, mounted in a flat plastic frame with a foam to compensate the shocks. The lead acid 12V 9Ah rechargeable battery, being the heaviest component, is stored under the base, between the wheels, keeping the centre of gravity low. Color camera with a true optical image stabilizer and CCD image sensor is mounted using anti-shock foam on a U-shape construction frame built of aluminum profiles, together with GPS and IMU sensor, see Fig.1. The camera is inclined 10° downwards. The IMU sensor must be mounted far from any sources of electric and magnetic fields, such as motors and wires. Placing GPS high compensates also for obstacles in the surrounding terrain, which may hinder the GPS satellites signal. The robot is built from raw materials, except of the motors, wheels and consoles that hold them, which are all part of a set from Parallax. The aluminium framework allows

mounting a rain shield for the computer and the camera when necessary.

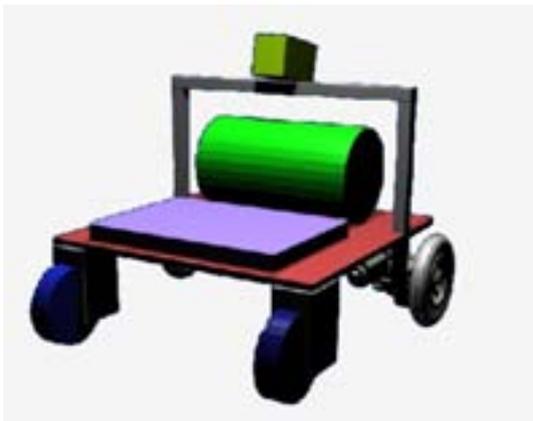


Fig.1 3D Model of the robot showing main parts. In real implementation, we have mounted only one caster wheel as it proved to be sufficient, and allowed more accurate control.



Fig.2 The resulting constructed robot from the side, front, and back. The control electronics is installed under the PC. The robot has already been tested in outdoor settings and has traveled a distance of several km.

2. Hardware Architecture

The robot is propelled by two 12V DC motors with built-in transmission, rotating at up to 150 rpm and consuming 1.5A at no load. The encoders with 36 ticks per rotation are used for speed and position feedback and are equipped with on-board microcontrollers that are directly connected to the motor drivers HB25, supplying them with the proper PWM signal to keep the requested speed. In this way, the main microcontroller board, which is the SBot control board, designed in our group originally for SBot mobile robot, is freed from the low-level motor control, and dedicates this task to both of the encoders that have an implementation of a standard P (proportional) controller and are connected using the same 1-wire serial bus. Unfortunately, we found that the original firmware for the encoders supplied by Parallax did not satisfy our needs for several reasons. Most importantly, the encoders were not designed for dynamic change of speed, but only for simple positional commands that accelerate from zero speed to a fixed predefined speed, and then decelerate after traveling the required distance. They do not allow to change the speed in the middle of such positional command. However, movements, where the speed and rotation is changed arbitrarily at any time, are required in the Robotour task, where the robot has to dynamically respond to the visual feedback when it has to align its movement with the shape of the path. Fortunately, Parallax makes the source-code for the encoders firmware available, and thus we could modify it to suit our application and support immediate smooth changes of the instant speed.

The obstacles are detected using the standard SRF-08 and Maxbotix LV EZ1 ultrasonic distance sensors that are connected to the main control board.

Outdoor robots are typically equipped with a global positioning device, i.e. GPS, and it is the case for our robot too. Information from the GPS module that is connected directly to the main computer using USB port, however, is not so reliable due to atmospheric and other occlusions, and serves only as a guidance for map localization. It is confronted with visual input and complemented by the current heading obtained from compass sensor. The compass sensor is part of the complex 9 DOF IMU sensor that includes several axes of gyroscopes, accelerometers, and magnetometers, thus compensating for various robot inclinations when traveling uphill or downhill. This is important since the simple compass sensors provide incorrect information once the robot and thus also the sensor is tilted.

Finally, for the visual input, we chose to use a standard video camera Panasonic SDR-T50, due to a very good ratio of parameters/price. The video camera is built around a CCD sensor, which has the advantage over the CMOS image sensors of taking the image instantly. Cheap CMOS cameras therefore suffer from a serious vertical distortion when the camera is moving, since the different rows of the image are scanned at different times. In addition, the camera has a built-in true optical image stabilizer, which further compensates for distortions due to the movement. Unfortunately, we found this stabilizer to be insufficient, and thus we have supported it with an anti-shock foam placed between the camera and the platform where it is tightened using flexible textile tape. The camera renders its image either as 16:9 or 4:3 image, however, it sends a wider signal down to its video output jack connector, which is further connected to a USB frame grabber card and the main computer. The main computer is a 2-core powerful PC with a GPU that can be used for the intensive image processing computation. The computer and the SBot control board are connected using a serial port or a virtual serial port over radio Bluetooth connection. In debugging and testing applications, the robot can be controlled using a wireless gamepad connected using a proprietary 2.4GHz radio link.

In general, the robot is designed in such a way that it can be used in many different applications. For instance, a stereo vision system or an arm with a gripper can be installed in the cargo hold area. Additional sensors can be easily mounted on the aluminum profiles or wooden base. Fig. 3 shows overall system architecture.

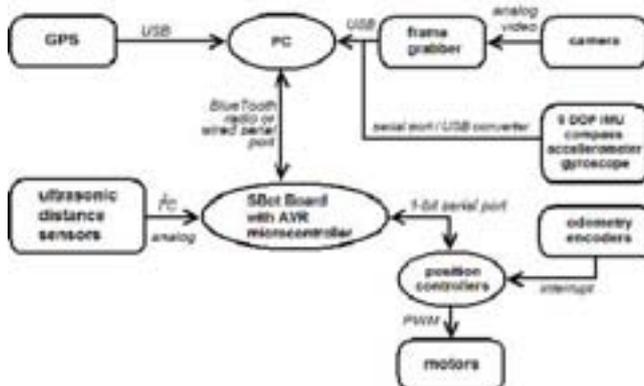


Fig.3 System hardware architecture.

3. Software Controller Architecture

The software architecture is tailored for the Robotour contest. In this year's contest, the goal for the robot is to

navigate to the target without knowing its starting location. It is only given the target coordinates and an official map of the park. It may not use other map information. The software controller is logically divided into five main components, see Fig.4.

The first component, planning, uses the map with the destination location and generates a path plan for the robot to follow. It tries to minimize the number and complexity of the crossings as these are the most critical places and candidates for navigational errors. The component outputs a sequence of locations that are to be visited by the robot. Whenever requested, the module can generate a new plan after a problematic place in the map has been reached.

The second component, localization using map, is responsible for the most accurate localization of the robot on the map. It is using the information from the compensated compass (IMU) for heading, from GPS for position estimation, and from the position encoders to estimate the distance traveled and turns made. All the information is integrated and with the help of the map and the path plan, the target distribution is determined using a probabilistic Monte-Carlo estimation. The output of the localization module is a probabilistic distribution over the expected heading in the very next correct movement, and the expected distance to the next crossing or target.

The third module, path recognition, is the most important one for the actual control of the motors, and has a priority over the localization module. It receives the image from the front camera and recognizes which parts of the image correspond to the path, and which of them correspond to other surfaces. The next section explains this procedure in more details. The output of this module is again a probabilistic distribution over the space of possible headings that can be projected to the input frame, where the headings leading to more "path" areas are more likely than those leading to less "path" area. Input from the odometry and gyroscopes helps this module to improve its estimation of the path using its previous estimations and the relative displacement of the robot.

The obstacle recognition module is responsible for detecting obstacles in the planned path of the robot and for stopping the robot in case of a possible collision early enough so that avoidance could be attempted by the coordination module. The robot is currently equipped with three ultrasonic distance sensors (front ahead, front left, front right), and thus the module reports on its output whether the path is blocked completely, or only partially, and also what is the size of the expected free buffer in front of the robot.

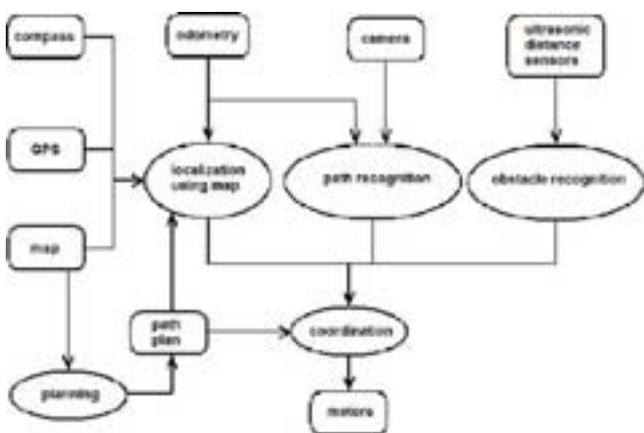


Fig.4 Overall controller architecture.

The most complex module is the coordination module. Its purpose is to take the prioritized outputs from the other three modules, and to determine the best possible angular and linear velocity for the next instant movement. When the confidence of the module is getting low, the robot slows down. If the confidence falls even lower, the robot stops, and starts rotating left or right, depending, which direction is expected to be more promising, until it finds a heading, where the module confidence is sufficiently high again. If such heading is not found, the robot attempts to return back in the reverse direction as it arrived to the problematic location, possibly moving in the reverse of the planned direction on the map. After returning back a short distance, it retries. The retries are repeated several times while gradually extending the back-up distance. If all attempts to pass the problematic location fail, the planning module is asked to generate a different path.

The controller is arranged in a behavior-based manner, individual behaviors are developed and tested independently before they are integrated in a common controller.

4. Path Recognition

Our goal was to use artificial neural networks in order to help the robot navigate and stay on the path. We obtained many images from a park with trails, and we have manually marked the regions in these images that correspond to the traversable path. This input was used to train the neural network (a standard multi-layer perceptron) to recognize the path. See figure 5 for an example of such manually classified image.



Fig.5 Manual preparation of training images.

Sending the whole image to the network as the input would obviously be infeasible. Instead, we first tried to scale the image to a lower resolution of 400x300 pixels, and divide it into 100 rectangular regions of equal sizes that covered the whole image. Each region formed an input to a neural network, and the whole region was about to be classified as "path" or "not path". However, the resulting resolution of the classified image was not satisfactory, even after a further reduction of the region size so that the image was divided into 2500 segments. Therefore, we decided to use a sliding region. For almost every pixel in the image, we define a corresponding region – it's larger neighborhood, which forms the input vector. The classification output produced by the network for each pixel in the image is then a real number from 0 to 1, estimating how much the network believes the pixel lies on the path. Two examples of images

that were not used in the training phase are shown in the Fig.6.

We used the RPROP training algorithm for multilayer perceptron, in particular the implementation that is present in the OpenCV package. The training used tens to hundreds of manually classified images from various places in a park with various path surfaces, light and shadow conditions. Since this is still an ongoing work and only preliminary results are available, we restrain from a statistical analysis of the results at this moment, and refer the reader to the page dedicated to the project with detailed results and data [5].



Fig.6 Examples of path recognition.

Once the network is trained and produces the classifications for the image frame pixels, the path recognition module enters a second phase, when it tries to evaluate all possible travel directions (headings) with respect to the chances that the robot will stay on the path. For this purpose, the module analyzes a family of triangles of the same area with the base at the bottom of the frame and the third vertex placed in the middle of the image. For each such triangle, we compute an average path likelihood. The triangle for which the path is most likely, i.e. where most pixels lay on the path, is likely to be the correct new heading. However, the module outputs a full distribution over all possible headings so that the coordination module can take advantage of this information, for instance to determine different directions at a heading, or when trying to resolve ambiguous cases. Fig.7 depicts the analyzed family of triangles. Two example pictures are further analyzed in Fig. 8, where the bars show how “likely” it is that following in the various directions is a “good” idea in order for the robot not to leave the path.

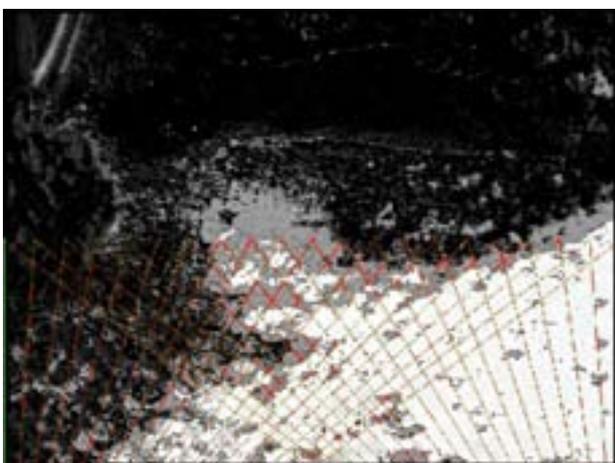


Fig.7 Triangles representing different turning projected to the image of recognized path.

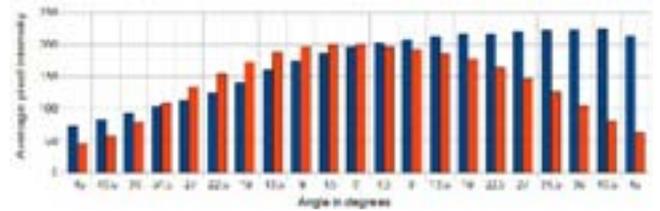
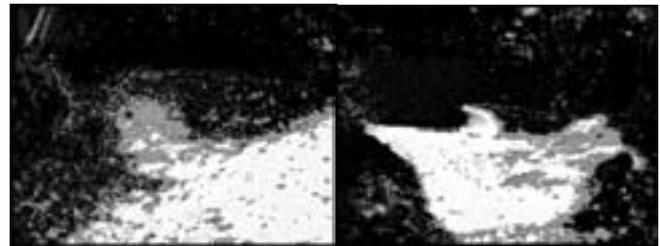


Fig.8 Two scenes after path recognition. The bars show the average pixel intensity of pixels inside of triangles for a range of different rotations for both of the resulting images (blue/dark for the left image, red/bright for the right image).

Conclusions and Future Work

We have designed and implemented a robotic hardware and software platform to be used in the Robotour contest for outdoor robots navigating in park environment. The hardware platform is implemented in a general way and most components of the software platform can be reused in other applications, the robot can be extended with stereo vision or manipulator. We have designed, implemented and tested in this context a new method for path recognition, which is based on artificial neural network that is trained on a set of static images that are similar to the environment where the robot is to be operating. We are currently working on integrating all the components of our prototype so that it could perform in its first Robotour contest this year. In the remaining 10 months of the project, we will analyze the results from our participation, and propose, implement, and verify improvements so that the robot can serve both as a competitive platform in the contest and as an educational tool in the course Algorithms for AI Robotics, which is provided at our department to students of Applied Informatics.

Acknowledgement

This work has been done with the support of the grant “Podpora kvality vzdelávania na vysokých školách” from Nadacia Tatra banky.

References

- [1] Robotour - robotika.cz outdoor delivery challenge, Rules, online: <http://robotika.cz/competitions/robotour/cs> last accessed: August 1st 2010.
- [2] K. Košnar, T. Krajník, and L. Přeučil, “Visual Topological Mapping,” in European Robotics Symposium 2008, Heidelberg: Springer, 2008, pp. 333–342.
- [3] G. Bradsky, A. Kaehler, Learning OpenCV: Computer Vision with the OpenCV Library, O'Reilly Media, Inc., 2008.
- [4] D. Gušťařík, SBot v2.0 – Educational Robot for Clubs and Classrooms, User manual to Sbot robot, 2008, online: <http://robotika.sk/>

[5] M. Nadhajský, Robotour diploma project page, 2010,
online: <http://virtuallab.kar.elf.stuba.sk/~mnadhajsky/>

Miroslav Nadhajský

Comenius University
Faculty of Mathematics, Physics and Informatics
Mlynská dolina
842 48 Bratislava, Slovakia
E-mail: miroslav.nadhajsky@st.fmph.uniba.sk

Pavel Petrovič

Comenius University
Faculty of Mathematics, Physics and Informatics
Department of Applied Informatics
Mlynská dolina
842 48 Bratislava, Slovakia
E-mail: ppetrovic@acm.org

Utilizing Lego Mindstorms as a Teaching Platform for Industrial Automation

Carolyn Oates and Alois Zoitl

Abstract

Industrial control systems are taught best using real systems. Such systems can be expensive, dangerous, and may break easily. On the other side simulations often do not react like the real system. IEC 61499 automation standard supports the current control system trend toward networks of event-driven distributed devices. Support for event driven control applications is new in IEC 61499 as are the tools supporting it. Three tutorials are presented to teach developing IEC 61499 event driven applications along with control theory basics using open source tools with the Lego™ Mindstorms hardware. This inexpensive training system can be used for teaching industrial control methods for students, as well as industrial professionals.

Keywords: automation; control systems; robotics

Introduction

Real control systems, such as an industrial robot arm, are expensive; can be dangerous [3]; and may break easily. In a simulator timings and physical modeling often do not react like the real system, teaching the students only the software. Additionally industrial automation systems are undergoing a major transition towards distributed control systems adding new development paradigms. The problem of industrial automation education has been summarized by [12] as follows:

“During the last few years the education in engineering and mainly the control engineering, has suffered multiple changes due to the fast technological development and the current demands of the field.”¹

In order to support industrial automation engineers, the IEC developed standards to define how distributed control systems should be developed. The result of this standardization activity is the IEC 61499 [3], which provides a framework for networks of event-driven distributed industrial control systems. IEC 61499 applications are built using networks of new kinds of functions blocks (FBs), supporting event as well as data connections. Support for both event driven and distributed control applications are newly supported and required for the first time industrial automation by IEC 61499. The new kinds of FBs which support distributed control applications need to be learned. Although the standard is available now for nearly five years, little tutorial information is available. As new open source based tools like the 4DIAC–Framework for Distributed Industrial Control—are becoming available the gateway hurdle for adopting the new technology is greatly reduced.

However a key open point for learning IEC 61499 based distributed control systems is the missing availability of cheap easily available training systems. Lego™ Mindstorms (LMS) offers with its building kit a flexible way of building small automation problems. Furthermore with the new system NXT it provides about the same computing performance as typical control devices used in the domain

of industrial automation. With this work we show how LMS can be used together with 4DIAC to teach IEC 61499.

LMS has a great history for teaching robotics and control programming also with block like programming languages. However none of the available tools provides languages suitable for industrial automation engineers.

Lego™ Mindstorms software (a subset of Labview) allows sequential commands. So when using LMS software to blink the LED located on the light sensor, there is typically one light sensor block for on and another for off for the same light sensor. The same sensor may be tested in different phases of an application using different blocks. Telling the motor to move occurs via multiple TurnMotor blocks. Labview has data connections, but no event connections [7].

Lejos, Java on LMS, is object oriented so there is only one instance of a physical sensor, but the method to reads a sensor can be used multiple times. Behavior programming described in the Lejos tutorial can still reference the same instance in multiple behaviors [9]. In comparison FB instances are restricted by the standard to the one physical existence.

This article is structured as follows. In Section 1 we give a short introduction to IEC 61499. The environment is described in Section 2, followed by a description on how we developed the tutorials. The developed tutorials are described in Section 4. Finally we conclude the article and describe our next planned steps.

1. Short Introduction to IEC 61499

The standard IEC 61499 defines several models—the application model, the system model, the device model, the resource model, and the Function Block (FB) model—that allow the control engineer developing distributed control applications in a graphical manner. This short introduction to IEC 61499 should serve as basis for the rest of this thesis. A full description of IEC 61499's architecture may be found directly in the standard IEC 61499-1 [6] or in a more comprehensible form in the books from Lewis [4] and Vyatkin [5].

¹ [12] || p. 3432

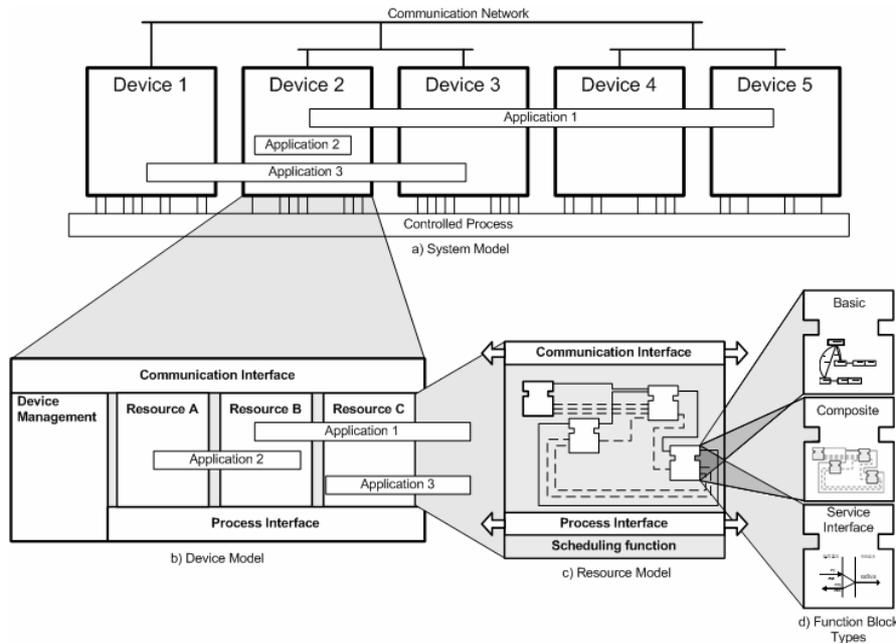


Fig.1 Overview on the main models of IEC 61499

The base model of IEC 61499 is the FB. A FB is a software component that is self contained and provides its functionality through a defined interface. This model has been adopted from the preceding standard IEC 61131-3 [11] and extended in its interface with an additional event interface. A trigger on one of the event inputs starts the execution of a FB. During the execution of the FB the input data will be processed, output data will be generated (depending on the functionality of the FB), and/or output events will be triggered. IEC 61499 defines three different FB types (schematically shown in Figure 1d):

Basic FBs (BFB) contain as main element a state machine that controls the internal execution on an input event arrival. This state machine is called Execution Control Chart (ECC) and is based on the Sequential Function Charts of IEC 61131-3. The ECC consists of three main parts: ECC-states with associated ECC-actions and ECC-transitions connecting the states. ECC-transitions are guarded by conditions. On an input event arrival the conditions of the current state's outgoing transitions are evaluated. The first true condition results in a state change. On state entry the associated actions of the state are executed. Actions consist of the execution of algorithms and/or triggering of output events. Algorithms may be programmed in any programming language. The main restriction is that algorithms can only access data inputs, data outputs, and internal variables.

Composite FBs (CFB) serve as container for FBs and may contain a whole set of FBs and their event connections and data connections. Incoming event connections and data connections are passed on to the internal FBs and vice versa for outgoing connections.

Service Interface FBs (SIFBs) provide a FB interface to functionality which is beyond the means of IEC 61499. Typical functionality encapsulated within SIFBs is the access to the control device's hardware, like the I/O interface or the communication interface. But also existing libraries that provide functions needed for the control system may be used through SIFBs. With SIFBs, this functionality can be encapsulated and the usage can be documented with so called service primitives. These service primitives allow to model event/data sequences explaining the usage of the SIFB. IEC 61499 distinguishes two general types of SIFBs. One is the requester SIFB, the other is the responder SIFB. The requester SIFB is an application

triggered FB which remains passive until an event arrives at one of its event inputs. The responder type is a resource or hardware triggered FB. That means that it can send output events resulting on actions in the resource or the hardware (e.g., interrupts).

Through interconnecting the FBs with event connections and data connections to Function Block Networks (FBNs) the control functionality can be modelled in the application model. Applications are in general modelled without any device or control infrastructure in mind. The control equipment with their communication networks used for the data exchange between the distributed controllers is specified in the system model. A second part of the system model is the so called mapping. The mapping regulates which parts of the application are located on which control device. For example in Figure 1a Application 1 is mapped to the Devices 2, 3, 4, and 5; whereas Application 2 is mapped only to Device 2.

IEC 61499 models control equipment that is capable of executing IEC 61499 applications as devices. A device consists of a communication interface, a process interface, a device management, and may contain resources (see Figure 1b). The communication interface provides communication services for the device and the application parts residing in this device. The process interface provides the services for accessing the sensors and actuators needed to control the process (e.g. read the current motor position).

A resource is a functional unit that serves as containment for applications or application parts residing in the specific device and has independent control of its operation. Within a device resources can be created, deleted, configured, etc. without interfering with other resources and their contained applications. For applications a resource has to provide an execution environment (Figure 1c). That means it has to deliver event notifications to FBs and has to allow FBs to process the incoming events corresponding to their internal structure. A resource gets access to the communication interface and process interface from the device. SIFBs are the means to provide these services to the applications.

The management functionality within a device has the main task to administrate all applications and all resources located in this device. The management also provides an external interface for engineering tools allowing engineering tools downloading and uploading applications to (from) the

device. This external interface is provided through the communication interface. Therefore the management needs an access to the communication interface (Figure 1b). At device level it provides the services to create, initialise, start, stop, kill, and delete the instances of resources and to query the attributes of resources. At resource level the same services allow the handling of FB instances and their interconnections.

2 Environment

The environment uses only open source applications. The 4DIAC-IDE is used to develop IEC 61499 standard compliant systems, applications and FB types. The standard provides portability and plug&play for controller applications. Applications are uploaded on to the Lego™ “controller” hardware [8] running the 4DIAC RunTime Environment (FORTE) under eCos operating system.

Oshows two IEC 61499 development tools, 4DIAC and FBDK. The tools generate XML files which comply with the IEC 61499 standard and can be exchanged.

As shown in Figure 2., the function blocks and applications are developed and mapped to device resources via 4DIAC-IDE. FBs are then exported to FORTE. The 4DIAC FB type export translates the FB's IEC 61499 XML representation into C++ code suitable for FORTE.

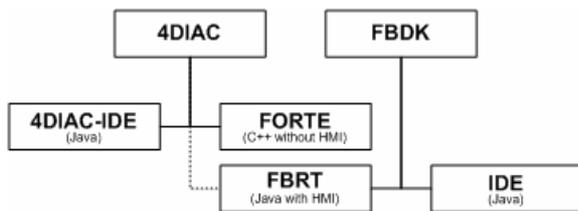


Fig.2 Development Environment

FBDK FBs are reusable, so a simulation using FBDK HMI FBs to display the output is possible. This is useful for unit testing FBs inputs and outputs by event. The “device” is a Java window.

After the FBs are developed the LMS firmware must be flashed with the eCos+FORTE using SAM-BA [1] (see Figure 3). SAM-BA is provided by Atmel, the maker of the at91sam7s (ARM7) chip in the LMS [8]. At this point FORTE is running a simple Ethernet over USB program to upload and run an application.



Fig.3. Upload to Lego™ Mindstorms NXT

eCos is an reconfigurable embedded operating system, so only the resources that exist in a device must be included. Control systems are typically embedded systems. Students who learn to work with LMS with eCos have a head start using eCos on other control devices.

3 Course Development

The tutorials assume no automation background. The tutorials build up concepts stepwise. Beginning FBs and IEC 61499 applications developed are reused and refined in following steps and tutorials. A simple example is presented and then the student must create or refine the presented example.

Research by Lego™ and MIT encourages the use of the freer explorative constructivist philosophy of education [2] by letting students explore rather than directed learning. However the problem solving cognitive philosophy is also popular for teaching control theory [12]. Teaching of basic concepts to model the problem need to be more guided. Once the student has framework to model the problem, they can be given more freedom and still communicate their work using IEC 61499 standard.

These tutorials are a mixture of cognitive and constructivism teaching philosophies. The first tutorial is guided problem solving learning, because specific control theory concepts using IEC 61499 standard are to be taught. After a basic example a related task, but slightly harder task is assigned. The second tutorial is meant to allow the student more freedom to use what they have learned. The only new concept is composite FBs. The third tutorial is a mixture and has the goal to teach the concepts of buffering and use of a bus.

Figure 4 shows a typical control loop. A line follower uses a controller to stay on the line. Calibration and software connection to hardware are also typical tasks in automation.

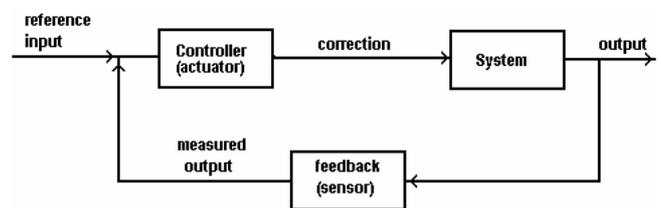


Fig.4 Feedback control loop

Tasks were chosen to teach control theory concepts as well as development of control application using the standard and tool.

In [5] three tutorials are presented for IEC 61499. The first tutorial modifies an LED application with 4 LEDs. The LEDs blink, or “chase” up or down. Turning an LED on and then blinking the LED was used as the first real test case for 3 different devices including LMS this semester. The NXT LabView Configuration VI also uses setting the light sensor’s LED as part of an example NXT software block [7].

In [5] the second tutorial used simple equation as the first full application. We tried a similar internal tutorial with 4DIAC/Forte for $x^2 + y^2$ with a network-like interface between FORTE (C++) and FBDK (Java). However there were many questions from students afterwards, especially about FORTE. There were fewer questions after developing an application to blink an LED. The blink application was first simulated with FBDK in a Java window and then applied to the actual hardware. The toggling the LED FB was reused in later applications.

We identified a set of key concepts of IEC 61499 and automation engineering for which it is important the trainee grasps in the first tutorials:

- How to represent feedback and feed forward control in IEC 61499
- IEC 61499 devices represent control hardware
- Sensors and actuators are represented by SIFB. Typically you have one instance of an SIFB per sensor or actuator.
- Error handling is performed through Boolean FB interface variables and appropriate events.
- Boolean input qualifier named QI is used for turning event processing on and off.

4 Tutorial Applications

The three tutorials developed teach the use of IEC 61499 function blocks to build three working applications. First the environment (4DIAC, [10]), hardware (LMS, [8]), and standard with a simple application are taught. A LMS FB library is provided with FBs to directly interface with the LMS hardware. This includes sensors, motor, and shutdown, plus hardware status (battery power status). A LMSUtil library will be developed during the tutorials.

Sensor FBs are associated with physical sensors or motors. Students must be careful to send and receive events to the FBs associated with exactly one physical resource. The second tutorial application uses a different kind of sensor hardware. The third tutorial application teaches timing and using buffers to send information between applications.

Tutorial 1: Line Follower

The first tutorial is a line follower application with on- and off-the-line calibration. If multiple light sensors are available the application can be expanded to use 2 or 3 light sensors.

We want to test if it more understandable to start with Basic FB (BFB) or a Service Interface FB (SIFB). A greater than BFB will be explained first. Then a two point controller (hysteresis) basic FB is assigned. So they go from a “one point” controller” to a two point controller.

For teaching how to provide an interface via a FB to the hardware the student is asked to develop a SIFB for the Lego™ light sensor. This should also help the student understand what the purpose FORTE C++ Eclipse compilation is for. The sample FB will be the touch sensor, which reads data the same as the light sensor. The light sensor ports must be initialized, which shows the direct connection to the ARM7 processor. The test application is a light blinking application utilizing the light sensor's LED. The Boolean data input QI is initialized to true. Errors indicated by the output variable QO=false are ignored for the moment.

The light blinking application also introduces the important and often needed Event FB library, which provides FBs for

manipulating the event flow as well as timed events (i.e., cyclic triggers or delayed events).

Next, the boundary between on-the-line and off-the-line for the environment and a light sensor must be found. First the light sensor must be read and connected into the calibration calculation. There is no single way to do two-color calibration, but it's important how the process knows and handles reading different colors. The top part of Figure 5 shows calibration where all samples of one color are read and all samples of the second color. All samples are averaged together.

Next the boundary between the two colors is used to turn on an LED if the light sensor is over “black” and off when it is over “white” as shown by DarkTst FB and Led4 FB in bottom part of Figure 5. Here it is emphasized that a port should only be used once.

Without error checking it is possible that a second FB for the same port/light sensor will be erroneously used. So error handling and Service Sequence diagrams explained must be explained. IEC 61499 uses + events to indicate no error and – events indicate error condition.

The application should now react when a port is allocated twice, since only one resource can be connected to a port. The INITO event combined with QO, event qualifier is split via an event switch into INIT- (QO=false) and INIT+ (QO=true). If the student previously used the same port/light sensor their design error will cause their application to no longer work.

From personal experience making this failure helps the student remember each FB represents one real physical resource.

Finally the state of the light sensor can be used to tell the motors how to move can now replace toggling an LED. This final application should also be developed stepwise. First instead of just turning the LED on and off, the 2 motors can be set. The LED toggle is left for debugging. Toggling one motor off when not over the dark line allows the application to follow the line in one direction only. This simple line follower is shown in Figure 5. When a basic version is

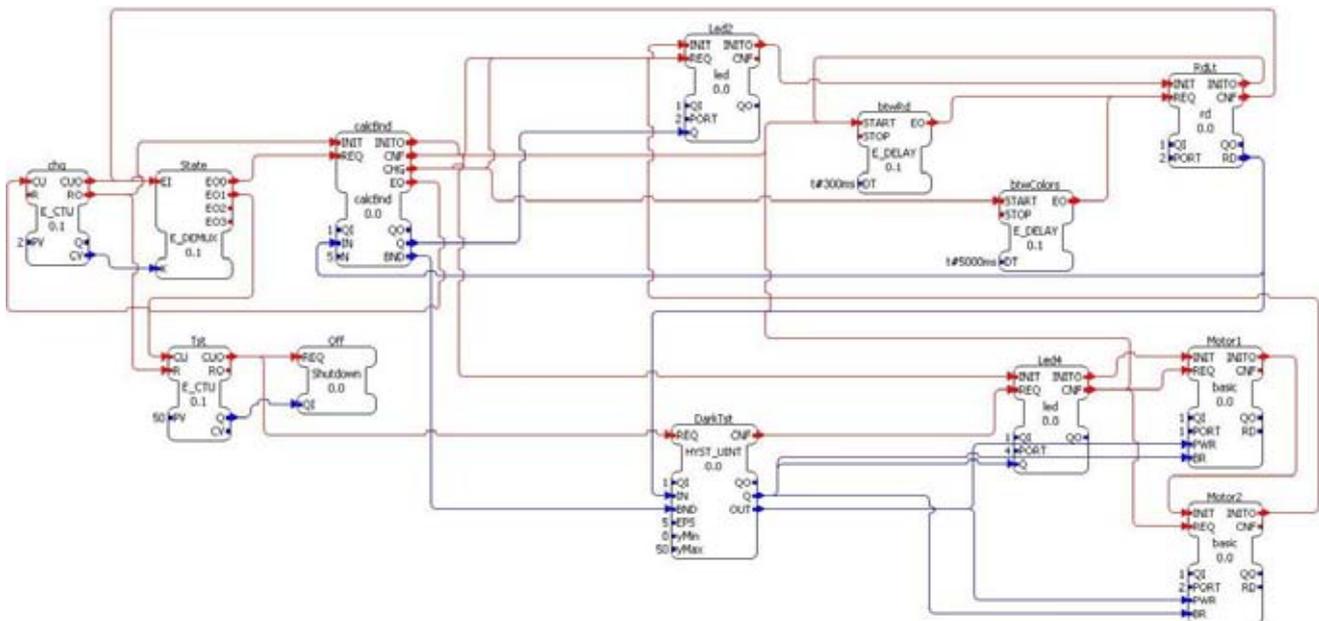


Fig.5 Simple Line Follower with light calibration

working, then the application can be expanded to a general line follower with feedback control and finding the line. Instead of just one light sensor to detect if the robot is over the line, multiple light sensors could be used. For example if three light sensors are used, the middle light sensor is over the line and other two light sensors straddle the line.

Further Tutorials

The second tutorial uses the ultrasonic sensor as part of a simple Cartesian robot to keep a certain distance from the car. A Lego™ robot must be built to move forward and backwards based the “car’s” length and up and down based on the feedback from the ultrasonic sensor. The student must develop their own control loop and Lego™ robot. Developing composite FBs is introduced to combine FBs together. Figure 6 shows how composite FBs would simplify the simple line follower by encapsulating the light calibration. The student must be careful to include error checking when needed in the composite FBs. Since only the input/ output events and variables are seen care must be given to not accidentally reuse the same port.

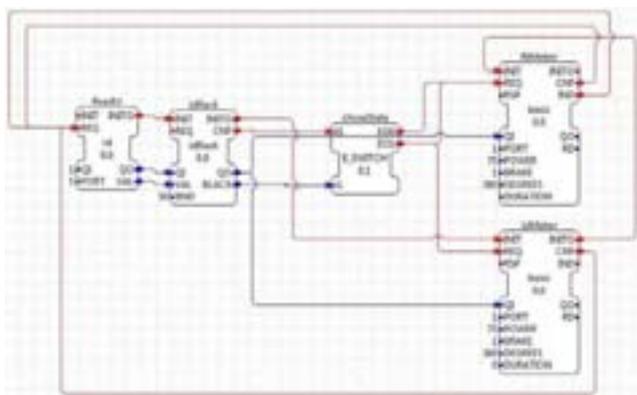


Fig.6 Simple Line Follower using composite FBs

The third tutorial uses stations to detect an object, its color, accept or reject it, and optionally deliver it. A pick-and-place robot is suggested. The application stations detect information and pass it on ahead of time via buffers, so the next station is prepared when the object arrives. This application teaches buffering data with time deadlines.

The IEC 61499 tutorial examples can build on each other if multiple LMS NXT kits are available. The car is used in car wash and the Cartesian robot as one station in the assembly line.

5 Conclusions and Future Work

Industrial automation is phased with major paradigm changes. First distributed control systems require a complete rethinking of how control applications are developed. By providing cheap and available tutorial systems control engineers can move up to the new paradigm much faster. With this work we showed how Lego™ Mindstorms NXT can be such a training platform for the new standard IEC 61499. We are developing tutorials which on the one hand utilize the LMS hardware and on the other side are representatives for typical industrial automation tasks. The final tutorial versions will appear on the 4DIAC website [10] under the development wiki.

Our next steps are to test the tutorials on different user groups in order to validate the contents and the structure of

the tutorials. The first tutorial will be tested with students doing a practice work for the institute this summer. The last two tutorials will be tested with students in the fall. We also plan to use wireless communication between Lego™ Mindstorms NXT to teach using devices and applications across device boundaries.

References

- [1] ATMEL, AT91 ISP/SAM-BA® User Guide, http://www.atmel.com/dyn/resources/prod_documents/6421B.pdf.
- [2] Mindell, D., Beland, C., Wesley, C., Clarke, D., Park, R., Trupiano, M. 2000. LEGO mindstorms, the structure of an engineering (r)evolution, 6.933J Structure of Engineering Revolutions, <http://web.mit.edu/6.933/www/Fall2000/LegoMindstorms.pdf>
- [3] S. Greengard, Making automation Work, Comm.ACM, Dec.2009, Vol.52, No.12, pp.18-19, <http://doi.acm.org/10.1145/1610252.1610261>.
- [4] R. Lewis, “Modelling Control Systems Using IEC 61499 – Applying Function Blocks to Distributed Systems”, The Institution of Electrical Engineers, London, 2001.
- [5] V. Vyatkin, IEC 61499 Function Blocks for Embedded and Distributed Control Systems Design, O3neida Publ.
- [6] IEC TC65/WG6, IEC 61499: Function Blocks, Parts 1 – 4, International Electrotechnical Commission IEC Std., Rev. 1.0, 2004/2005.
- [7] Labview, Creating Lego Mindstorms NXT Software Blocks, ftp://ftp.ni.com/evaluation/mindstorms/NXT_Creating_MINDSTORMS_Blocks.pdf
- [8] LEGO™ MINDSTORMS NXT Hardware Developer Kit, <http://mindstorms.lego.com/en-us/support/files/default.aspx>
- [9] LeJos Tutorial, <http://lejos.sourceforge.net/nxt/nxj/tutorial/index.htm>
- [10] 4DIAC – Framework for Distributed Industrial Automation and Control, www.fordiac.org
- [11] IEC TC65/WG6, IEC 61131-3: Programmable controllers – Part 3: Programming languages. International Electrotechnical Commission, Geneva, 1993
- [12] Paja, C., Scarpetta, J.. and Mejia, E. Platform for Virtual Problem-Based Learning in Control Engineering Education, p-3432-3437, IEEE EDUCON 2010, April 2010.

Dipl.Ing.Carolyn Oates / Dr. Alois Zoitl

Vienna University of Technology
Automation and Control Institute (ACIN)
Gußhausstraße 27-29 / E376
A-1040 Vienna, Austria
Phone: +43 1 58801 - 37601
Fax: +43 1 58801 - 37699
E-mail: oatesc@acm.org
zoitl@acin.tuwien.ac.at

An Open Platform for Teaching and Project Based Work at the Undergraduate and Postgraduate Level

Benjamin N. Passow, James Wheeler, Simon Coupland, and Mario A. Gongora

Abstract

Robots are a great tool for engaging and enthusing students when studying a range of topics. De Montfort University offers a wide range of courses from University access courses to Doctoral training. We use robots as tools to teach technical concepts across this wide and diverse range of learners. We have had great success using the Lego RCX and now NXT on the less demanding courses, and conversely with the MobileRobots Pioneer range for postgraduate and research projects. Although there is a distinct area in between these two where both these platforms meet our needs, neither is suitable for every aspect of our work. For this reason we have developed our own hardware and software platform to fulfil all of our needs. This paper describes the hardware platform and accompanying software and looks at two applications which made use of this system.

Our platform presents a low-cost system that enables students to learn about electronics, embedded systems, communication, bus systems, high and low level programming, robot architectures, and control algorithms, all in individual stages using the same familiar hardware and software.

Keywords: Teaching, Project Based Work, Undergraduate, Postgraduate, Robotics, Embedded Systems, Programming, Algorithms, Hardware, Software, Case Study

Introduction

Robots and control systems have become essential parts of modern industry and are increasingly used in education. Within many teaching curricula, pupils are often introduced to robots at the primary school stage, where they learn concepts such as direction, angles, measurement and sequencing. At this level, Roamers [1], Pixies [2], and BeeBots [3] are popular choices due to their simple programming interface and “friendly” appearance.

At a higher level, students may make use of their theoretical knowledge by applying these to a real world machine [4]. General computer science as well as robotics and artificial intelligence students begin to explore the mechanics of robot design, constructing their own robots and adding sensors and actuators to suit a particular challenge. In this format there is generally some form of processor unit or brain that contains the control instructions and connects to the sensors and actuators. The control software is often developed on a standard PC and then uploaded to the controller via a communications link. Common choices for this format are the Lego Mindstorms [5] RCX and NXT and the Robix Rascal [6]. This practice showed to be effective for motivating students in practical activities [7].

For teaching software processes relating to control systems, it is often desirable to employ a robot platform with standard actuators and sensors (e.g. having motion, vision, hearing, proximity detection etc) with an embedded PC as the central control processor. In this environment, students learn to write control software that uses the underlying operating system to communicate with the available sensors/actuators. Examples of such robot platforms include the MobileRobots Pioneer and Peoplebot [8].

At De Montfort University, whilst we have found the Lego Mindstorms kits and the MobileRobots equipment to offer extremely useful platforms for the various teaching courses offered, there are some concepts, such as electronic design and embedded programming, that neither platform allows us to teach in the way we would like. For this reason we have developed our own printed circuit board (PCB) with an onboard Microchip microcontroller and several I/O connections that easily interface to commonly used actuators and sensors. Since a student version of the Microchip Integrated Development Environment (including editor, compiler, debugger and programmer) is freely available, we may use this as the main environment within which students develop their embedded code. The Microchip In-Circuit Debugging tools are relatively cheap and provide a useful means for interfacing between a host PC and the robot control platform.

By using a modular approach to the design of the platform along with its accompanying electronic interfacing and software libraries, we are able to easily reconfigure the platform according to the nature of the concepts being taught. As an example, for first year students we can provide them with pre-built sensor circuitry and a software library of high-level C functions that enable them to design a simple embedded system whilst shielding them from the lower-level complexities of electronics and software. As the teaching programme progresses, the control platform can be reconfigured so that students are required to design their own electronic interfaces or write their own low-level software in order to accomplish the tasks assigned to them.

This paper describes the development of the platform and software libraries in more detail. We include two case studies highlighting how the platform has contributed to the

teaching programme at both first year Bachelor course level and also at Master and Doctoral training levels. Finally we offer a conclusion that summarises how this approach may be of benefit to other educational establishments with a robotics teaching programme.

1. The Platform

The hardware side of the platform consists of a printed circuit board with voltage regulation, a 16bit Microchip programmable interrupt controller (PIC), analogue and digital peripheral input and output pins, two RS232 serial ports, I²C bus, pulse width modulation (PWM) and motor control outputs. The PIC is programmed via a commercially available USB in-circuit debugger. This section will introduce and discuss the platform in more detail.

1.1 Hardware and its Components

The design of the board is optimised for mechatronics and control projects. It is based around a Microchip microcontroller dsPIC30F4011, which can run at up to 30 million instructions per second (MIPS), has 48kB program memory, 2kB random access memory, 1kB non-volatile EEPROM memory and 31 I/O ports. The PIC is powered by 5 VDC for the digital power supply, which is regulated by a standard analogue voltage regulator LM7805. We used the TO-92 package to maximise the power dissipation capability so that a range of battery voltages can be used, up to an online-charging lead acid battery at 14.8VDC. Since this board is intended for robotics projects, it is assumed that it will be used with batteries only, not a mains power supply; as such it has no rectifier diodes or large ripple-filtering capacitors at the input. It includes only the compact capacitors required to filter the feedback and noise from the digital clock and circuitry and the power devices that might be connected (e.g. electric motors).

This particular PIC provides three PWM-specific outputs (balanced pairs of digital outputs); two of which are connected to a dual motor driver chip L298N. This provides two full H-bridge PWM direct motor power outputs from the PCB. The H-bridge driver chip provides an interface between the digital supply voltage (typically +5VDC) and the battery voltage (typically +12VDC), which supplies power directly to the motors through the H-bridge. We have tested powering motors from 7-12 volts from different types of batteries (e.g. 7.2V or 9.6V from an array of NiMH, 7.4V from an array of Li-Po and 12V from standard sealed Lead acid), and our system has shown to be quite effective for most applications. All standard protections are included in the PCB so that the students need only connect the motors directly; there is a set of flyback fast switching inverse diodes to ground and power VCC (battery) and capacitor in parallel with the motor. The third PWM set of outputs from the PIC is available for expansions in the projects via a connector in the PCB.

Four of the PIC's signals are dedicated for driving RC-hobbyist servos (pulse position controlled position-servo mechanisms). These position-servos draw the power from the 5VDC regulated power supply to avoid problems when using batteries above 9V, which would be outside the tolerance of such devices (typically designed to work between 4.8V - 9.0V). The outputs from the PIC are connected to four 3-pin headers arranged in the standard Ground-Power-Signal configurations used by most RC-hobbyist servos.

Finally, there are two more dedicated headers, both intended for communications. One uses one of the PIC's UART pins to connect to a standard RS232 serial port. The

Quantity	Interface name and description
<i>Actuators:</i>	
2	Full H-bridge motor drivers
1	Full-balanced PWM digital output
4	Direct connections to PPM position-servos
16	Simple digital actuators via I/O ports
<i>Sensors:</i>	
<127	I ² C sensors Available to our students are: · Digital Compass · Ultrasonic ranger · Other boards
9	Analogue sensors (1Msps @ 10bit) Available to our students are: · Light dependent resistor · Inertial measurement unit (IMU)
16	Digital sensors (various)
<i>Communication:</i>	
2	UART serial ports (RS232 via converter)
1	I ² C bus (master or slave mode)
<i>Expansion:</i>	
17	Additional programmable I/O pins

Tab.1 Platform Interfaces and available equipment

pins come directly to the headers so that the digital signals from the PIC are available directly, i.e. there is no RS232 level-converter driver on the PCB. This allows connecting directly to other digital serial ports. If a standard serial port is going to be used (e.g. to connect to a computer) then an external RS232 level converter (e.g. MAX232) is required. We have various mini-PCBs with a MAX232 already mounted for use in various projects. The other communications header provides digital signal connection to the I²C port from the PIC. This is mainly used for connecting to peripherals such as ultrasonic rangers, electronic compasses or IMUs. The addressable structure of this serial bus allows multiple devices to be connected and it has proved to be very useful and versatile as there is a vast range of peripherals, sensors, etc. that are available commercially and at low cost using this protocol.

The remaining I/O pins of the PIC are connected to a general-purpose header, which the students can use to connect any other type of peripheral or device not covered by the other headers mentioned above.

This convenient and compact design provides the optimal configuration for robotic and control projects. Table 1 summarises the platform's available interfaces for the students to use.

1.2 Development Environment and Tools

The microcontroller is programmed and can be debugged using Microchip's in-circuit debugger ICD2. This device is connected via USB to the host machine running the integrated development environment called MPLab. The standard programming language that comes with this development environment is assembler. In order to program with a high level programming language, an additional cross-compiler is required. We use Microchip's C30 compiler which is freely available for research and student projects. The compiler is fully ANSI compliant and includes a set of libraries for easier device configuration and use.

2. Software Libraries

To enable students new to programming and robotics to work with the platform we have written a set of high level functions for them to use. This section details some of the software libraries that provide simple software interfaces to functionality such as timers, sensors, communication, and motor control.

2.1 Timers

At the heart of any embedded controller is a timing system, our system is no different. Our application programmable interface (API) supplies four basic functions which can be combined to give all timing functions necessary:

```
// Initialise timer device
void timePassed_init (void );

// Reset timer device
void timePassed_reset (void );

// Get elapsed time (ms) as a uint
unsigned int timePassed_ms ( unsigned char );

// Get elapsed time (s) as a uint
float timePassed_fs ( unsigned char );
```

The function `timePassed_init` sets up the timer by setting the relevant configuration bits on the PIC's timers. This function must be called before the other timing code will work. The function `timePassed_ms` returns the elapsed time in milliseconds as an integer whereas `timePassed_fs` returns the elapsed time in seconds as a floating point number. Elapsed time in both these functions is a measure of how much time (measured using processor clock cycles) has elapsed since the PIC timer was reset. The PIC timer is reset by four possible actions:

- Calling `timePassed_init()`.
- Calling `timePassed_reset()`.
- Calling `timePassed_ms(1)`.
- Calling `timePassed_fs(1)`.

Although the initialisation function must reset the timer, we also provide the explicit `timePassed_reset()` reset function. Additionally the timer may be reset when measuring the elapsed time by calling the relevant function with a parameter of 1. These functions provide a simple interface for measuring time in milliseconds and seconds.

2.2 Analogue to Digital Converter

The analogue to digital converter (ADC) provides access to readings from analogue sensors connected to our system. Our API provides four functions for controlling and accessing the sensor readings from the ADC:

```
// Initialise ADC
void myadc_init (void );

// Start the ADC reading timer
void myadc_startReadings (void );

// Stop the ADC reading timer
void myadc_stopReadings (void );

// Read data from the ADC
int sensorReading (char sensorNumber );
```

The ADC needs to be initialised, this is done by calling `myadc_init(void)`. The initialisation routine sets up a timer

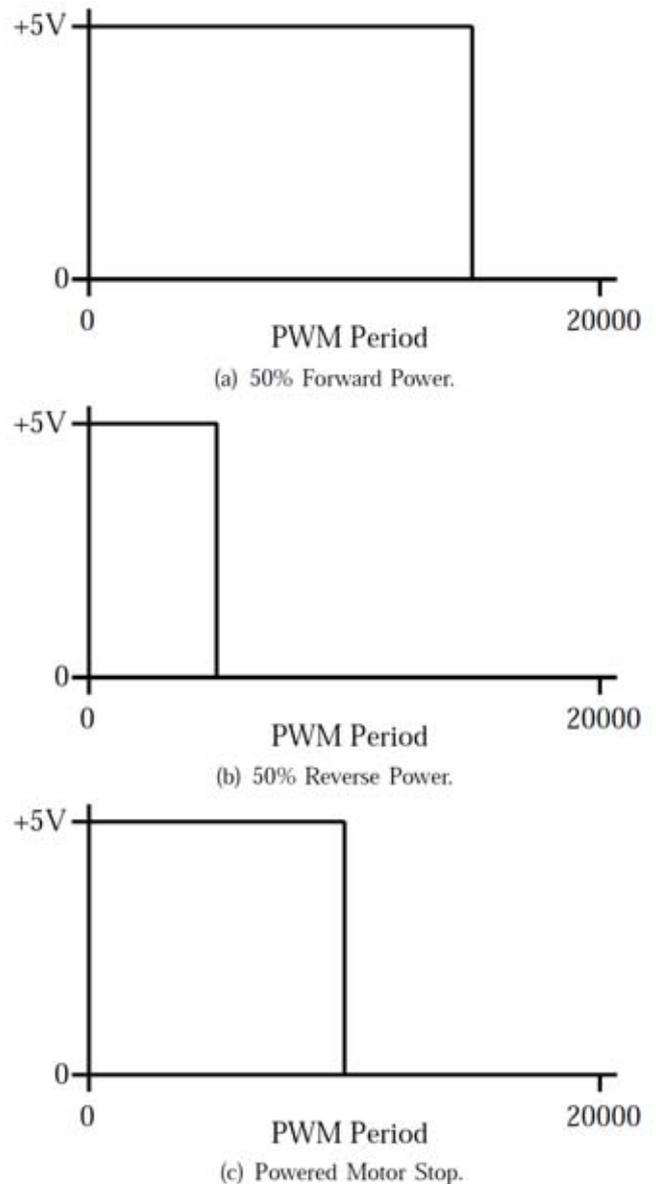


Fig.1 PWM Motor Control with an H-Bridge.

driven interrupt system which reads data off the ADC according to a timer which can be controlled through the API. The timer is started and stopped using the `myadc_startReadings(void)` and `myadc_stopReadings(void)` functions. When the timer elapses it causes an interrupt routine to run with regular frequency. The interrupt reads data from the ADC to a predefined data structure via a mean of two filter. This data can be accessed through the `sensorReading(char sensorNumber)` function. This is in effect an interrupt-driven polling system – the ADC is polled with a regular frequency as designated by a timer. It is worth noting that the polling timer causes interrupts to be raised, meaning that although the ADC-API uses a polling system this could be modified to a pure interrupt driven system fairly easily.

2.3 Motor Control

The motors are controlled using a standard pulse width modulation approach, taking into account that an H-bridge motor driver is used. Two duty cycle registers are utilised, one for each motor, with forward and reverse control. Figure 1 depicts the forward, reverse, and powered stop control of a single motor using PWM through an H-bridge motor driver. Our API provides three functions for controlling the motors:



Fig.2 KITTDASH9 on a Sumo Arena.

```
// Initialise the motor control system
void MotorControlPWM_Init (void );
// Set the motor speed of both motors
void MotorSpeed (int motorLeft ,
int motorRight );
// Turn a choice of motors off
void MotorOff(int choice );
```

The MotorControlPWM_Init() function needs to be called before motor speeds can be controlled. This function sets up the two duty cycle registers and organises the relevant pins for PWM output. The MotorSpeed(left, right) function takes integers as percentage values i.e. calling MotorSpeed(-25, 75) causes the left motor to turn in reverse with 25% power (not speed – generally power to speed is a non-linear relationship) and the right motor to turn forward with 75% power. The MotorOff(choice) function turns off one or more motors when passed one of three constants: MOTORLEFT, MOTORRIGHT or ALLSTOP. If the function is called with MOTORLEFT or MOTORRIGHT then the respective motor is stopped with a powered stop (see Figure 1(c)), if called with ALLSTOP then PWM is switched off (PWM timer base is disabled), switching off power to the motors and letting the motors drift.

3. Application Case Studies

The platform introduced in this paper has been used in a variety of projects including an inverted pendulum robot, balancing weight robot, an autonomous Dr Who Dalek, a sumo fighting robot and an autonomous helicopter. We focus on the latter two for our application case studies of the hardware and software as they are on the opposite ends of the higher education spectrum.

The first case study looks at a robot built by first year students on our Artificial Intelligence and Robotics Bachelors degree. This robot took part in the standard sumo competition at the 2009 Robot Challenge in Vienna. The second case study investigates how a compact version of the same system was used to control an autonomous helicopter for a Masters dissertation and later on in a PhD project.

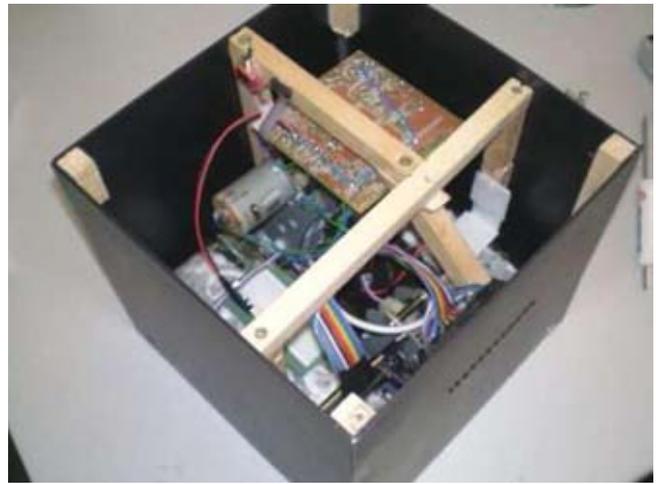


Fig.3 The Interior of KITTDASH9.

3.1 Sumo Robot – KITTDASH9

KITTDASH9 was built by a group of first year undergraduate students studying Artificial Intelligence and Robotics at De Montfort University. The students built the robot within the robot club which runs once a week and not during formal teaching time. The robot was designed and built to be entered in the standard class of the robot sumo competition at the Robot Challenge 2009. Figures 2 and 3 show the KITTDASH9 including the mounted embedded system (note that it is mounted upside down) and drive train.

The robot has four custom built light intensity sensors, one on each corner and a modified serial ball mouse to provide a basic form of odometry. The robot has no range finding or bump sensors. Locomotion is provided by two independently driven tracks fitted with a high traction rubber surface. The robot is fitted with a lighting effect system consisting of an array of red LEDs controlled by a separate PIC which is connected to the main embedded system being discussed here.

The students implemented a finite state machine control architecture, as depicted in Figure 4. Each state has a clear control objective which is implemented through a combination of the timer and motor control functions from our API. Transitions between the states are enacted by a combination of states from the light intensity sensors, given on the state transition diagram as a binary string, for example 0101. Notice the light intensity sensors give binary readings. The students achieved this by taking readings from the light intensity sensors, using the ADC part of our API, and putting them through a hard limiter to decide whether the sensor is over a white surface or a black surface – the only two surfaces the robot will encounter during a sumo battle. Each sensor has an individual threshold, allowing each sensor to be individually calibrated.

As mentioned earlier, KITTDASH9 is fitted with a modified serial mouse. Although the students did not manage to use this sensor in their control process, they did (with significant help) manage to get readings from the mouse unit. The mouse was connected directly to the second serial connection on the embedded system. As the mouse ball moves, events are generated and data giving the amount of motion in the x and y axis are sent on the serial bus. Each event consists of three 7 bit words (see Table II) and the motion reading must be decoded from these three words as given below:

$$dx = \text{word1} \& 0x03 \ll 6 + \text{word2} \& 0x3F$$

$$dy = \text{word1} \& 0x0C \ll 4 + \text{word3} \& 0x3F$$

References

- [1] "Roamer official website," May 2010, http://valiant-technology.com/uk/pages/roamer_home.php.
- [2] "Pixie official website," May 2010, <http://www.swallow.co.uk/pixie/pixie1.htm>.
- [3] "Bee bot official website," May 2010, <http://www.beebot.org.uk>.
- [4] V. Douglas, "Robots make computer science personal," *Communications of the ACM*, vol. 49, pp. 12–25, 2006.
- [5] "Lego mindstorms on wikipedia," May 2010, http://en.wikipedia.org/wiki/Lego_Mindstorms.
- [6] "Robix rascal official website," May 2010, <http://www.robix.com/>.
- [7] J. Adams, S. Turner, S. Kaczmarczyk, P. Picton, and P. Demian, "Problem solving and creativity for undergraduate engineers: findings of an action research project involving robots," in *International Conference on Engineering Education ICEE*, 2008.
- [8] "Mobilerobots research robots website," May 2010, <http://mobilerobots.com/ResearchRobots.aspx>.
- [9] P. Bourke, "Decoding data from the microsoft serial mouse," April 2003, available at <http://local.wasp.uwa.edu.au/~pbourke/dataformats/serialmouse/> or on CDROM from <http://local.wasp.uwa.edu.au/~pbourke/>.
- [10] B. N. Passow and M. A. Gongora, "Optimising a flying robot: Controller optimisation using a genetic algorithm on a real-world robot," in *Proceedings of the International Conference on Informatics in Control, Automation and Robotics, ICINCO'08*. Madeira, Portugal: INSTICC Press, May 2008, pp. 151–156.
- [11] B. N. Passow, M. A. Gongora, S. Coupland, and A. A. Hopgood, "Realtime evolution of an embedded controller for an autonomous helicopter," in *Proc. of the IEEE Intl. Congress on Evolutionary Computation (CEC'08)*, Hong Kong, June 2008, pp. 2538–2545.

Benjamin N. Passow

De Montfort University
Centre for Computational Intelligence
Gateway House
Leicester, LE1 9BH
United Kingdom
E-mail: benpassow@dmu.ac.uk

James Wheeler

De Montfort University
Centre for Computational Intelligence
Gateway House
Leicester, LE1 9BH
United Kingdom
E-mail: jw@dmu.ac.uk

Simon Coupland

De Montfort University
Centre for Computational Intelligence
Gateway House
Leicester, LE1 9BH
United Kingdom
E-mail: simonc@dmu.ac.uk

Mario A. Gongora

De Montfort University
Centre for Computational Intelligence
Gateway House
Leicester, LE1 9BH
United Kingdom
E-mail: mgongora@dmu.ac.uk

Robotika.SK Approach to Educational Robotics from Elementary Schools to Universities

Pavel Petrovič, Richard Balogh, Andrej Lúčný

Abstract

The Association Robotika.SK organizes and participates in a wide range of activities, projects, contests, events, workshops, summer schools, seminars, prepares educational materials, builds educational hardware and software platforms. This article presents our viewpoint on educational robotics, the challenges, tasks, goals, and means of achieving benefits for the learners and teachers. We summarize several years of experience we collected and provide perspectives on the future development, and some of our future plans.

Keywords: educational robotics, robotics contests, robotics summer

Introduction

In a happy society, people get the chance to work on what they believe in. We believe in robotics, and we think that we can also improve the chances of others who share our common interests. We believe that robotics technology can help humans to avoid arduous, repetitive, dangerous, and unpleasant tasks; we believe that robotics technology does and will allow us to reach beyond our current horizons, both in microscopic and macroscopic worlds, but within our environments as well. Robots may help rescue people, animals, and other living creatures in critical situations. Robotics can be applied to make our environment cleaner. Robotics technology might bring easier, cheaper, more versatile and flexible solutions to various common tasks. Moreover, we believe that robotics technology can contribute to improvement of the educational process in schools, it can provide entertainment, and for young people, it can be the reason for spending of lot of their time in a valuable and useful way. Robotics can also attract many young people to the fields of science and technology. We also honestly believe that reasonable application of robots in production process will not take the work from people and generate unemployed. On the contrary, the resources saved by cheaper production can be used to give better and more interesting work to the people, in more comfortable working conditions. We think there are not enough robots around us and large efforts are needed to bring them here. We founded the association Robotika.SK, a non-profit, non-political and non-governmental organization, and we use it as a platform for organizing cooperation of institutions of higher education, preparing seminars, talks, summer schools, competitions, initiating, coordinating and realizing various projects, supporting schools, and individuals. All activities are centered around our information website robotika.sk that always brings up-to-date news from the activities organized by us and our partners, as well as robotics news from our region, and outside. In this article, we give an overview of our past and current activities. The following sections describe our viewpoint on educational robotics, the overall structure of our activities, cooperation, individual robotics projects, student work, seminars and talks, summer schools, contests, and public presentations.

Educational Robotics

The omnipresence of technology today is a fact. However, a traditional view prevails, namely that technology is still completely dependent on us. Mobile phones, portable computers, digital assistants, intelligent security systems, automatic vending and money transfer machines, advanced technology in production - everything remains fixed at a single place where it was installed, or wherever we take it with us. Soon, however, the technology will start to move in our environments on its own. Automatic delivery, monitoring and service, personal assistants, cleaning, guiding, shopping, and many other tasks will be performed by autonomous mobile devices working on our behalf. And even those that will still be fixed, they will be able to act more autonomously and take smart decisions in dynamic environments as contrasted to being pre-programmed to a fixed sequence of operations.

Many of the tasks named above are performed by robots already today and we must get prepared for this forthcoming age. In particular, we must:

- make sure people will be able to understand the mode of operation of these devices;
- make sure people will be able to control, and even program such devices to utilize their potential;
- prepare enough skilled engineers, who will be able to create them, and provide the necessary service;
- keep building a sufficiently large community of professionals in all related areas, which are important for the progress of development of robots - material science, energy science, physics, electronics, mechanical engineering, communication and human interaction, computer science and technology.

This is why we need educational robotics today, to foster the progress and development, and to avoid stagnation and crises. Every meaningful application of the robotics technology in any form of educational process is a valid contribution. The following ideas have been tried and implemented:

- organizing robotics summer schools and summer camps
- organizing competitions with robots
- building hobby-robotics clubs, labs, and free-time centers
- teaching programming with robots
- using robotics to explain and elaborate on mathematics
- using robots in teaching physics and science
- setting up interdisciplinary student projects utilizing robots
- developing special courses with introduction to robotics
- implementing lectures about robotics into various courses
- building robotics hardware and software platforms
- using robots as educational toys from very early age
- developing art projects and presentations with robots

We believe all these ways are useful and important ways to increase the competence of the general population and specialized students, and we think there are large unfilled spaces in particularly in finding and developing new platforms with completely new features, approaches and ideas. We argue that even though it is important to support the main-stream product lines such as LEGO Mindstorms NXT, it is also important to search and support different systems. Still, only very little has been done on larger-scale parallelization, modular architectures, non-conventional kinematics, and other areas. We will continue our attempts to actively contribute to at least some of them.

Structure of our activities

We are a small group of scholars and students with some links with industry. We maintain a student and research robotics laboratory. In our institutions, we teach a few courses related to robotics, and outside of them, we try to maintain robotics clubs in primary or secondary schools. We participate in organizing various relevant activities that are initiated or organized by us or our partners. We mention both kinds for completeness. Our activities spread across several levels:

- events for general public, where everybody can register and visit, the aim is the popularization of science and technology;
- events for schools, where participants (i.e. teams) from schools can register, and participate, these events have a more specific target group and thus can be better tailored for their audience;
- events for selected students from technical universities, for instance organized in cooperation with student networks of technical universities;
- events for students in our institutions, these are local events, tailored for our students;
- activities focused on development of robotics platforms and projects that are made publicly available for those interested;
- student projects in a form of bachelor or master theses, or other types of student projects including students from secondary schools;

- publishing information, articles, and materials related to robotics technologies, methodologies, etc.

A description of individual activity types follow.

Cooperation

Our group is built on cooperation. In the very beginning it put together people from three different institutions: two universities and one private company producing sensor technologies, Microstep-MIS. However, many of our activities would be impossible without efficient cooperation with our partners, who include student organizations (BEST), non-profit associations (e.g. InnoC from Austria, Slovak Society of Electronics, Robotika.cz), foundations (e.g. Children of Slovakia Foundation), primary and elementary schools (e.g. Spojená škola Novohradská, Spojená škola sv. Františka z Assisi, ZŠ Karloveská 61), and private companies (e.g. RLX, Microstep s r.o., AVIR, Freescale Semiconductors, and other). We also cooperate with various individuals, for instance, the author of the RoboSapien Dance Machine Project, local hobby photographers who needed a robot-operated camera, or artists who are exploring new art forms utilizing technology.

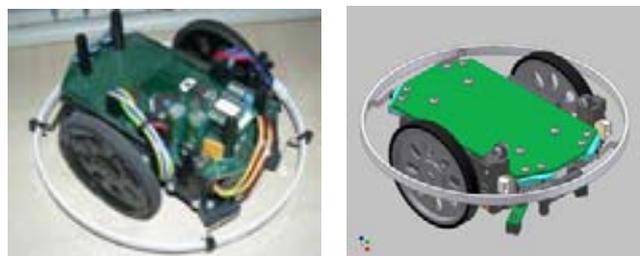


Fig.1 Sbot robot platform with Bluetooth radio communication, autonomous control, line sensors, bumpers, installable IR proximity sensors and encoders, with easily extendable circuit board. This is a figure annotation, aligned according this multi-line example

Individual Robotics Projects

Various robotics projects represent the type of activities we put our emphasis on, and when also our learning is most intensive. The projects are often developed in cooperation with students, or they are student projects. Sometimes the projects overlap with the contests, when we work as team leaders, or supporters who provide the background, equipment, and guidance. Here, we would like to name a couple of example projects:

Robotnačka v.2 - the drawing robot, controlled from LOGO language

This is a hardware platform, which can be attached to a Logo turtle, which is normally only drawing on the screen. In this way, learning programming becomes much more entertaining, and the robot can also be used for new activities, utilizing its sensors, for example, teaching geometry [1,2].

Robotnačka drawing shapes based on bitmap image

A secondary school student software project: the application received a bitmap image on its input, extracts contours from the image and generates a trajectory to be drawn by Robotnačka [3].

Remotely-Operated Robotics Laboratory

A permanent installation of robots in a laboratory that is always available on the Internet. The robots in the laboratory can be controlled the same way as locally connected Robotnačka - from Logo language, or, alternately, from a web browser, or C++, Java, or another type of application [4].

S-bot and Acrob robot platforms for education and projects

Platforms that were developed in our group for the purpose of simple robotics experiments, bachelor theses, exercises on locomotion and navigation [5, 6], see Fig. 1.

Remotely controlling WowWee family robots

A USB device for sending arbitrary IR signals that could be used to control RoboSapien and other WowWee robots [9]. We also developed a solution for controlling the robots using LEGO IR tower and directly from RCX programmable brick, see Fig. 2.



Fig.2 Controlling RoboPet from RCX using IR signals.

Student work

We use the robotics laboratory to provide the bachelor and master students with a working environment, and the required equipment. In our courses, students get hands-on experience in using robots of different types - LEGO NXT robots, BoeBot robots, Robotnacka, Acrob and Sbot robots. In these exercises, they learn basics about kinematics, signal processing, sensor types, calibration, and control. In the last two years, the following bachelor theses have been successfully completed:

- Probabilistic mapping in remotely-operated robotics laboratory (2009)
- Bayesian Robot Programming (2009)
- SBOT Sokoban (2010)
- Localization using distance sensors (2010)
- Mobile robot for category line-follower (2010)

and the following diploma theses:

- Representations in Evolutionary Design (2005)
- Visual Programming of Control System for a Colony of Robots (2007)
- Robotic laboratory experiments for secondary school physics (2010)
- Cellular Embryogenic Representations for Evolutionary Design (2010)

- Didactic materials for the topic robotics construction sets and Imagine Logo (2010)

The exact references can be found at our wiki page [7]. Currently, several other bachelor and diploma theses are in progress.

Seminars and Talks

Our group runs an internal seminar for students and researchers, but more importantly, we invite various speakers to give lectures on topics related to robotics. For instance, we organized a talk about chemical robots (Doc. Štípanek from VŠCHT Praha), and a talk about Constructionism and Robotics in Schools (Prof. Alimisis from School of Pedagogical and Technological Education in Greece).

Even though our organization is not educational by the definition, one of the best results achieved in previous years is participation in the international project Centrobot, where some joint Austrian-Slovak lectures for the students of secondary schools both from Vienna and Bratislava were organized, Fig.3. There is a big potential of increased motivation of the students from different countries to work on joined robotics projects together. This allows them not only to acquire the knowledge and skills, but also to gain a different perspective, open their minds, and compare their own performance with others.



Fig.3 A joint Slovak-Austrian lecture, Vienna, February 2010.



Fig.4 Centrobot robotics summer school 2010.

Summer schools

For several years, we have been organizing an event called "Robotic holidays", a one week intensive lab work with lectures and talks. Typically in the beginning of the summer or in September, interested students and people joined us to work on several more or less challenging robotics projects. During the last three years, together with the student organization BEST (Board of European Students of Technology) and InnoC (Austrian Association for Innovative Computer Science), we organized a summer school for

students from technical universities across Europe - twice in Bratislava and one time in Vienna. This two-week course includes lectures, workshops, excursions, and leisure activities. Fig. 4 shows a group work from our robotics summer school in 2010.

Contests

Contests are very central part of educational robotics, and they cost a lot of our time and energy. The main advantages of contests are:

- a fixed deadline - improves planning skills, makes it easier to prioritize and focus
- a clearly specified task, which was selected by experienced people in such a way to be solvable, non-trivial, and interesting
- often a standardized platform with a broad user base, which allows good access to information, saves time and efforts
- the possibility for the participants to compare their skills with their peers
- the school or club can make itself visible, this is a great motivation to produce an excellent result
- a nice possibility for building social and professional networks
- contests have a healthy competitive and sporting atmosphere, everything is subordinated to allow a perfect result of everybody

Istrobot

Istrobot is the primary contest of our association, where we are the main organizers. The tradition dates back to the year 2000 and a permanent quality growth can be observed. At the present time the contest consists of four different categories: The Pathfollower for linefollowing robots, Micromouse for maze solving robots, MiniSumo for fighting robots, see Fig. 5, and Freestyle for everything else, see Fig. 6. The contest is attracting approximately 100 robots each year and only our internal limits stopped its additional growth. The best experience from this contest is that it really fosters the development of the mobile robotics in our region. With a surprise, we find many research papers in local conferences inspired with robots solving the maze, or line-followers.

RoboTour

RoboTour is an outdoor robotics contest organized by the Czech association Robotika in Czech Republic. In the year 2010 it goes international for the first time and it takes place in Slovakia, Bratislava. Design of an autonomous intelligent vehicle appropriate for such contest inspired by the famous Grand Challenge contest is challenging for our university partners and our material support is very useful. Participants from universities and clubs in Czech Republic and Slovakia (with one exception of a foreign team) compete in autonomous outdoor robot navigation in a leisure park. Robots are allowed to use both global navigation - such as GPS, compass, accelerometers, inclinometers, etc. and local navigation such as ultrasonic distance sensors, laser range sensors, landmarks. Vision is typically the most important component, responsible for keeping the robot on the track, which is necessary to prevent an instant "game-over". This contest serves also as a good reference testing platform for various image processing problems and generates many interesting solutions of the navigation problems. Members of our association have participated in

RoboTour for about four years, and this year, our association has received an invitation to organize the contest in Bratislava.



Fig.5 MiniSumo dead-match at Istrobot 2010 contest.



Fig.6 Robotic Arm - Freestyle competition, Istrobot 2010.



Fig.7 FIRST LEGO League regional tournament in Bratislava, 2009.

FIRST LEGO League

We are actively involved in organizing regional tournament of FLL in Bratislava that is taking place for the third time this year. It is a contest for teams of 5-10 members in the age of 10-16 years. The strengths of this competition lie in the focus on creativity and team work, excellent preparation of tasks, which are solved by tens of thousands of students round the globe. It is also important that every year, a completely new challenge is to be solved, and thus it is

impossible to participate with the same robot year after year. In consequence, also excellent novice teams have a high chance of succeeding. In addition to building and programming the robot, the competition requires completing a research project and preparing a presentation. In this way, the young people get a taste of what it means to be a researcher. However, here we also see some weaknesses. In particular, the research themes are too complex to comprehend for that young people. We would like to see themes that would pose challenges appropriate to their age. For instance, many interesting small research projects in physics and chemistry at the level of elementary school can be completed to demonstrate interesting phenomena. Such experiments are genuine and achieve what they claim, answer the research question completely, and understandably. This is in contrast with typical FLL research projects that, for instance, propose to build dams, reorganize city traffic system, or find cures to diseases... That type of projects resembles somewhat the concept of "Let's pretend" society, where fridges, TV sets and CD players stop to work two weeks after the expiration period. In our local contest, we try to guide the coaches to lean towards easier projects that correspond to the knowledge level of the children. Our association not only participates in organizing the contest, but also provides equipment and staff to the participating teams. Fig. 7 shows a view from our local FLL contest.

RoboCup Junior (RCJ)

RoboCup Junior is a world-wide educational initiative targeted at young people up to age of 19 years. There is less team work focus in RCJ, individual teams are not an exception. There are also no restrictions on the material and software used as they are in FLL. Succeeding in RCJ (except, perhaps in the RoboDance category) requires several years experience, and advanced technical skills. Access to the information and guidance is a bottleneck, teams guided by students from technical universities or skilled engineers working in relevant industry have an advantage compared to the teams from schools in the countryside. Despite these shortcomings, we feel that RCJ in Slovakia contributes greatly to the interest in science and technology, it leads hundreds of young people through the experience of larger project, and it is a popular contest with good spirit. Our association supports this contest by all means. Fig. 8 shows a scene from RCJ in Slovakia. All information can be found at [8].



Fig.8 From RoboCup Junior Slovakia, February 2009.

Freescale Race Challenge (FRC)

This contest is an initiative of the Freescale Semiconductors company and its goal is to make use of the accelerometers to control the speed of the racing cars autonomously. We supported two student teams with material and advices to

actively participate in this contest, see Fig. 9. Resulted autonomous cars are very good attractor also for public presentation and were used in Istrobot contest and in Elosys trade presentation. This contest is also a very good motivational tool to study embedded systems hardware and sophisticated methods of signal processing and even learning and mapping of the toy racing car track.

Robot Challenge

Robot Challenge is the World's largest contest (from the point of view of the number of registered robots). It takes place in Vienna, and it is organized by InnoC, one of our cooperating partners. Robotika.SK always both actively cooperates and participates in the contest. We have an active exchange of participants between the Istrobot and Robot Challenge contest, which have similar categories. This exchange - best described by "a bus of Slovak participants arriving to Robot Challenge" supports Robotika.SK funding of Slovak-Austrian cooperation.



Fig.9 Freescale Race Challenge, Žilina, 2010, a team from STU Bratislava.



Fig.10 Researchers' Night, Bratislava, 2007: vision-guided Boe-Bot that follows a ping-pong ball, and the drawing robot Robotnačka.

Public Presentations

When possible, we try to present our results to the public. We participate and support presentations of our alma maters at the annual trade show EloSys in Trencin, where we occupy a booth with robots presenting their behavior for visitors. Usually the school groups are attracted and hopefully also motivated for additional studies of technical disciplines. We also participated on the Researchers Night's - a EU coordinated science popularizing project, see Fig. 10. Our presentations were also the part of various international events as the festival of cocktail robotics in Wien Roboexotica, Eurobot national contest in Prague, etc. These are important events for creating new contacts, and attracting young people to the field, which is the aim of our activities in general. Results of the students project and our own platforms make a good jobs here. Moving and operating installations are a base of successful presentation, but the human explanation is always required for public.

Conclusions

Educational robotics is a young field that springs form and connects many different areas. However, it has a place of its own, and it requires separate attention. Not only to prevent repeating the same mistakes, but also to provide a place for exchanging ideas, technologies, platforms, solutions, and a discussion.

In this article, we introduce and summarize the activities and viewpoints of the non-profit association Robotika.SK. We are proud to claim that most of our activities have raised interest in robotics, science and technology among the young people and the target audience. This claim can be supported by the number of participants and students joining our activities and projects and their positive feedback. In the future, our efforts will continue according to the challenges and possibilities we will face, keeping the cooperation and team work as our main working method.

References

- [1] ĎURINA, D., PETROVIČ, P., BALOGH, R. (2006), "Robotnačka - The Drawing Robot", Acta Mechanica Slovaca., In Acta Mechanica Slovaca., June, 2006. Vol. 10(2-A), pp. 113-116
- [2] P. PETROVIČ, "Mathematics with Robotnacka and Imagine Logo", Eurologo 2005, DrukSfera, ISBN:83-917700-8-7, pp.353-360, 2005.
- [3] L. BOŠKO, "Project Contours", Contribution to the Eucys contest, org. by Mladí vedci Slovenska, December 2007. online project repository: <http://webcvs.robotika.sk/cgi-bin/cvsweb/robotika/robotnacka/contours/>
- [4] PETROVIČ, P., LÚČNY, A., BALOGH, R., ĎURINA, D. (2006), "Remotely-Accessible Robotics Laoboratory", Acta Mechanica Slovaca., June, 2006. Vol. 10(2-A), pp. 389-194.
- [5] Robotika.SK: Sbot 2.0 – Educational Robot for Clubs and Classrooms, online: http://webcvs.robotika.sk/cgi-bin/cvsweb/~checkout~/robotika/sbot/2.0/doc/dokumentacia_en.pdf?rev=1.1;content-type=application%2Fpdf
- [6] Acrob – Arduino Compatible Robot, online: <http://ap.urpi.fei.stuba.sk/sensorwiki/index.php/Acrob>
- [7] Bachelor and Master Theses of Robotics Laboratory @ RoboWiki, online: http://virtuallab.kar.elf.stuba.sk/robowiki/index.php?title=List_of_Master_and_Bachelor_Thesis_Related_to_Robotics_Laboratory
- [8] RoboCup Junior in Slovakia, online: <http://robotika.sk/rcj/>
- [9] Controlling RoboSapien using LEGO IR Tower, online: <http://www.robotika.sk/projects/robsapien/index.php>

Pavel Petrovič

Comenius University
Faculty of Mathematics, Physics and Informatics
Department of Applied Informatics
Mlynska Dolina, 842 48 Bratislava, Slovakia
E-mail: pavel.petrovic@gmail.com

Design and Validation of a Robotic System to Interactively Teach Geometry

Lorenzo Riano and T. M. McGinnity

Abstract

Learning geometry can be significantly improved if the student interacts with shapes and their transformations. We present the design and validation of a robotic system that teaches geometry by natural interaction. The robot is able to draw arbitrary shapes in the environment by means of its own movement. It is also able to detect and track the student movements, representing them as geometrical shapes and reproducing them.

We will show four experiments where a robot is able to draw mathematical shapes, including an affine transformation, and two experiments where the robot reproduces trajectories a student previously showed it.

Keywords: geometry teaching, robot control, splines, particle filter

Introduction

In the late sixties Seymour Papert invented Turtle Graphics and LOGO [1], a programming language for geometric drawing as a tool for basic programming and geometry teaching. A real or simulated robot, shaped as a turtle, is programmed to move in a 2D space and to draw lines while moving. The main reason for its development and success is that it places the learner in an interactive environment where he/she can experiment with the geometry involved in drawing a figure, while receiving direct visual feedback from the turtle movements [2].

It is commonly assumed that the teaching of geometry should contribute to the learning of, among others, "the movement between theoretical objects and their spatial representation" [3]. Observing an abstract shape being drawn by a real robot should therefore significantly contribute to a student's understanding of geometry. Other concepts such as affine transformations can be more efficiently learnt by a student if they observe their effects on an arbitrary shape or observe the effects of varying parameters [4].

In this paper we present a robotic system that interactively teaches geometry according to the guidelines above. In particular, it describes shapes in the environment by means of its own motion. To the best of our knowledge this is the first application of a real robot to teach geometry by natural interaction.

Such interaction starts from the student, who defines a geometric shape. This can be accomplished in two ways: i) by using the mathematical description of a shape, or ii) by moving in the environment, thus describing a shape via his/her own movements. Once a shape has been chosen, the robot shows what its approximation of it looks like, using a simulated trajectory. The student can vary parameters such as the smoothness of the approximation and the speed at which the robot should travel. All of these parameters have an intuitive interpretation, and the student can immediately visualise the effect of changing them by observing how

much the proposed robot trajectory matches the original shape.

When the student is satisfied with the proposed trajectory, the robot starts moving along it. This way it is virtually "drawing" a shape in the space by moving in the environment, providing a visual feedback to the student. Once the robot stops moving the student can manipulate the shape by using affine transformations, and observe again the robot moving along it. This way the student can "see" geometry, and he/she can interact with it by manipulating figures and observing the effect on the robot's movements.

The main target audience of the proposed system are primary school pupils, who are learning the basic concepts of geometry. In this case observing the shapes being drawn by the robot's movements can improve their understanding of the subject [3]. Our system can as well be used in an undergraduate robotics course, where the students are presented with problems of path planning and trajectory following, and they can observe the results of varying several parameters on the physical robot itself.

In order to carry on the above task the robot requires several components, namely i) people detection and tracking, ii) trajectory extraction and iii) path following. The rest of the paper describes in more detail the proposed system and the experimental results that validate the system.



Fig.1 The Scitos-G5 during a demo.

1. People detection and tracking

An effective people detection and tracking system is the one that minimises both false positives and negatives, i.e. it does not mistake objects in the environment as human.

1.1 Face detection

In order to reliably detect people in the environment, we used the Viola and Jones approach to face detection [5]. This classifier is scale and light conditions invariant.

In order to measure its performance, we conducted two experiments, the first one with a person always facing the robot camera, the second one without any person in the environment. During the first experiment the robot was following the person, while during the second one it was randomly moving. Both experiment lasted around 15 minutes each, and the results are summarised in the confusion matrix shown in Table 1, left.

	Present	Not Present
Detected	98%	24%
Not Detected	2%	76%

	Present	Not Present
Detected	78%	0%
Not Detected	22%	100%

Tab.1 Confusion matrix for (left) the Viola-Jones classifier and (right) when combined with the RBFN, for a person present or not present.

These results show that, although the Viola-Jones has an outstanding true positives rate of 98%, it performs poorly in false negatives with a rate of 24%. A second drawback of the Viola-Jones approach is that a person has to directly face the robot to be detected. This is a major problem for our application as we want a person to describe a trajectory in the space, and this would not be easy if he/she has to face all the time the robot. A third drawback of the Viola-Jones approach is that it can detect people only up to about 2m away from the camera when using wide angle lens. We will describe a solution to this problem in section 1.3 with the introduction of the particle filter for tracking.

1.2 RBFN for legs detection

The Viola-Jones approach described above relies on vision to detect faces. A robot is usually equipped with several sensors, which can be combined to make a classifier stronger. In the past the laser sensor has been used to detect people using their legs, either as the sole sensor [6] together with a came [7], [8]. In most of the previous works, in order to train a laser based classifier a huge set of legs laser scans had to be manually constructed and labeled [6] or the authors created an ad-hoc classification algorithm [7]. Our approach is to use the face detection algorithm described before to train a Radial Basis Function Network (RBFN) classifier [9] for legs detection. This way the training process is completely automated.

In order to train a RBFN to classify leg patterns in laser scans, we collected the training data using the Viola-Jones face detection algorithm described above. Specifically, we had the robot running for 20 minutes with a person constantly in front of it, while recording the laser scan readings in a set C_1 . As we stated before, the detection rate of the face detection algorithm is 98%, so almost all the laser scan readings refer to legs. The camera had been calibrated so that for every pixel it is possible to calculate

the angle θ between that pixel and the camera itself. Considering that both the camera and the laser are vertically aligned, θ identifies a unique laser reading, taken at angle θ in a whole laser scan. For every face detected, its centroid is extracted and the corresponding angle θ is calculated. Every laser scan in C_1 is then clipped between angles $\theta - \pi/6$ and $\theta + \pi/6$. This way C_1 is a training set for the class 1, "legs in a laser scan", composed by 60-dimensional vectors. We then built a second set of laser scans C_0 by letting the robot randomly move in an environment with no people in it for 20 minutes. This set represents all the laser scans with no legs in it. Both C_0 and C_1 are then been used to train the RBFN.

The face detector and the legs detector have been combined to create a new people detection algorithm, that outputs 1 if both Viola-Jones and the RBFN detect a person. The new people detector confusion matrix is summarised in Table 1. It can be seen that the true positives detection rate dropped from 98% to 78%, but the number of false positives is now 0%, thus solving the problem with the Viola-Jones only approach. Once both classifiers agree that a person has been detected, the robot can switch to laser classification only. This solves the problem of a person having to face the robot camera all the time.

The only drawback of this approach is that the training set C_1 contains legs patterns which are only closer than about 2m, because this set was "created" by the Viola-Jones detector. This means that the RBFN legs classifier can detect people only up to about 2m.

1.3 Particle filter

The 2m limit described before does not allow a person to move arbitrarily in the environment, which is necessary to describe shapes. In order to solve this problem we employed a particle filter. An excellent review of this technique is in [10], while an application of it to people tracking using a laser sensor is in [11].

A particle filter requires a big number of particles to work reliably [12]. As a computational trade-off, in our application each particle does not try to detect people, but it tracks only a single laser scan. For this reason the adopted tracker relies heavily on a people detector that does not report false positives.

We tested the particle filter in several experiments using 2000 particles. In the worst case the tracker failed after 4 minutes of continuous operations, but in the average it lasted around 10m. Moreover, the particle filter is able to track a person movements up to 8 meters away from it.

2. Trajectory following

An arbitrary shape is represented by the robot as a parametric B-Spline. A motor controller that integrates feedforward and feedback signals is used to drive the robot along that shape. In the following we will give an overview of both B-Splines and the proposed controller.

2.1 B-Splines

A Basic Spline (B-Spline) is a parametric curve often used for interpolation and regression [13]. They have been widely used in robotics to approximate trajectories [14], [15], [16].

Given $m+1$ real numbers $t_0 \leq t_1 \leq \dots \leq t_m$ and $m+1$ control points $p_i, i=0, \dots, m$, a B-Spline of degree n is a parametric curve $S(t)$ composed of a linear combination of basis functions $b_{i,n}$ of degree n , as given in (1).

$$S(t) = \sum_{i=0}^m p_i b_{i,n}(t) \quad (1)$$

where $b_{i,n}(t) \neq 0$ only if $t_{i-1} \leq t < t_i$. This makes a B-spline piecewise defined, i.e. the basis functions are non-zero only in a closed interval. In order to obtain a C^2 smooth function, usually a cubic polynomial is used as a basis function $b_{i,3}$. When used to approximate a parametric curve, B-Splines are defined as in (2).

$$S(t) \equiv (S_x(t), S_y(t)), t = 0, \dots, 1 \quad (2)$$

where $S_x(t)$ and $S_y(t)$ are two B-Splines.

A B-Spline can be used to approximate a function $f(x)$ represented as a finite set of points (x_i, y_i) . In this case the sum of squares error (3) is zero.

$$\sum_{i=1}^N (y_i - S(x_i))^2 \quad (3)$$

When the data points (x_i, y_i) are noisy, the interpolation requirement is not reasonable. In this case the smoothing spline in (4) is preferred [17].

$$\sum_{i=1}^N (y_i - S(x_i))^2 + \lambda \int_{x_1}^x N[\ddot{S}(x)]^2 dx = \min \quad (4)$$

where $\lambda \leq 0$ is a *smoothing factor*. When $\lambda = 0$, we obtain the interpolation again. The higher λ , the less the spline is constrained to pass through the data points (x_i, y_i) . In the following we will use λ to generate smooth approximations of people trajectories.

B-Splines are *affine invariant*, i.e. if an affine transformation is applied to a B-spline curve, the result can be constructed from the affine images of its control points. This is very important in our application as we want the robot to show the effect of geometric transformations on shapes. Moreover, the derivatives of a B-Spline can be analytically calculated. This will come at hand for the development of the controller.

2.2 Controller

The trajectory tracking for mobile robots is characteristically a nonlinear problem. Several solutions have been proposed in the past. However, in most of them only simulation results are presented [18], [19], [20], or they require complex calculations that are not feasible in a real-time controller [21], [22]. In this work we decided to adopt a feedforward-feedback control law: the robot follows the trajectory described by the B-Spline, and at the same time tries to minimise a distance error using a PID controller.

1) *Feedforward control law*: The motion of a unicycle robot can be described by the system of differential equations in (5).

$$\begin{cases} \dot{x} = v \cos \theta \\ \dot{y} = v \sin \theta \\ \dot{\theta} = \omega \end{cases} \quad (5)$$

where x, y, θ are the robot position and orientation, and v, ω are the robot linear and angular speeds. If we consider a time interval dt sufficiently small, then we can approximate the robot movement as piecewise circular, where the radius ρ of the circle in the time interval $[t, t+dt]$ is in (6).

$$\rho = \frac{v(t)}{\omega(t)} \quad (6)$$

Any parametric curve $(x(t), y(t))$ can be approximated the same way by a set of arcs, whose curvature is in (7).

$$\kappa = \frac{|\dot{x}\ddot{y} - \dot{y}\ddot{x}|}{(\dot{x}^2 + \dot{y}^2)^{3/2}} \quad (7)$$

Considering that the curvature of a circle of radius ρ is constantly equal to the reciprocal of the radius, we can rewrite (6) as in (8).

$$\frac{v(t)}{\omega(t)} = \frac{1}{\kappa} \quad (8)$$

Moreover, the angular coefficient of the tangent to a point (x_0, y_0) of a parametric curve is given in (9), which is also the instantaneous linear speed v of a point moving along that curve.

$$v = \sqrt{\dot{x}(x_0)^2 + \dot{y}(y_0)^2} \quad (9)$$

By substituting (7) and (9) into (8), we obtain the feedforward control law for the robot angular speed in (10), which is a function of the B-Spline derivatives and the robot linear speed.

$$\omega_f = \frac{|\dot{x}\ddot{y} - \dot{y}\ddot{x}|}{\dot{x}^2 + \dot{y}^2} v \quad (10)$$

As outlined in the introduction, the user can select two parameters, namely the time approximation dt and the robot speed v . Once the user has selected them, the robot shows a simulated trajectory. This is created by using a first order Euler approximation of (5), where ω is given by (10) and dt, v are supplied by the user. This is useful to check if the robot is theoretically able to follow a given trajectory. However, as (5) does not take into consideration the robot dynamics, this method is necessary but not sufficient to guarantee a correct trajectory following.

2) *Feedback control law*: The role of the feedback controller is to correct the robot when it deviates from the desired trajectory. This deviation can be calculated by estimating the distance between the robot and the trajectory. As there is no analytical solution to this problem, we used the Newton method to find a zero of the function

Given a point (p, q) and a parametric curve $(x(t), y(t))$, the squared distance between the point and any other point on the curve is given in (11).

$$\Delta x(t)^2 + \Delta y(t)^2 \quad (11)$$

where we defined $\Delta x(t) = x(t) - p$ and $\Delta y(t) = y(t) - q$. To find the minimum of (11) we differentiate and equate to zero, as in (12)

$$2 \Delta x(t) \dot{x} + 2 \Delta y(t) \dot{y} = 0 \quad (12)$$

The Newton method is an iterative method to find the zeros of a function $f(t)$. It starts with a guess t_0 and every step it updated the candidate solution as in (13)

$$t_i \leftarrow t_{i-1} - \frac{f(t)}{f'(t)} \quad (13)$$

By applying (12) to (13) we obtain the iteration step (14) to find the minimum distance between a point (p, q) and a curve trajectory.

$$t_i \leftarrow t_{i-1} - \frac{\Delta x \dot{x} + \Delta y \dot{y}}{\dot{x}^2 + \dot{y}^2 + \Delta x \ddot{x} + \Delta y \ddot{y}} \quad (14)$$

During our experiments we found that the Newton method requires only a few iterations to converge to a solution, so it is usable in a real-time controller.

If the robot position is (x_r, y_r) and the closest point on the desired trajectory is (x_d, y_d) , then we can define the angular error as in (15).

$$e_\theta = \arctan\left(\frac{y_d - y_r}{x_d - x_r}\right) \quad (15)$$

The feedback control law is a PID with the error in (15) to steer the robot. The final robot controller is given in (16)

$$\begin{cases} v = \text{const} \\ \omega = \omega_f + \omega_b \end{cases} \quad (16)$$

where ω_r is the output of the feedforward controller (10) and ω_b is the output of the PID when supplied with the angular error (15).

3. Experimental results

All the experimental results have been produced using a Scitos-G5 robotics platform equipped with a laser range finder and a camera (Figure 1). The trajectories have been recorded using the Vicon tracking system, that allows tracking an object with an accuracy of 1mm . Every trajectory, before being replicated by the robot, has been rotated and translated so that it starts from the robot position and it is aligned with the robot x axis.

We performed two groups of experiments: in the first group the user asks the robot to follow a pure geometric shape, while in the second group the user defines a shape by moving in the environment. For every shape we show the corresponding one in the robot frame of reference, the simulated approximation (see section 2.2.1) and the real shape as produced by the robot movements. Table 2 shows the parameters and the statistics for every shape.

For every shape we calculated the approximation error as the mean absolute distance between the user-provided trajectory and the approximated one. The errors for each experiment are summarised in Table 2. Note that if we were using the mean error the results would have been close to zero, as they are equally positive and negative (cf. Figures 2, 3 and 4). However, the mean absolute error presents a clearer picture of the worst case scenarios.

	v	dt	λ	Error [m]
Cosine	0.3	0.08	0	0.17
Spiral	0.2	0.08	0	0.21
Square	0.3	0.08	0.01	0.2
Sheared square	0.3	0.08	0.01	0.19
Trajectory 1	0.4	0.1	2	0.30
Trajectory 2	0.4	0.1	2	0.28

Tab.2 Parameters and statistics for the tested trajectories..

Mathematical shapes: We tested four mathematical shapes: i) a cosine, ii) a spiral, iii) a square and iv) a sheared square. Figure 2 shows the last two shapes together with the spline approximation. The approximations of the cosine and the spiral match exactly the original data, so they are not shown here. Figure 3 shows the simulated and the real robot trajectories for all the mathematical shapes.

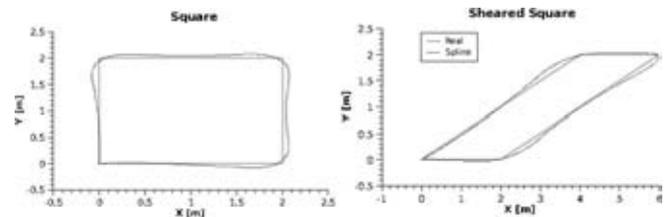


Fig.2 The square and the sheared square used during our experiments.

User produced shapes : We tested two user produced shapes. Both of them have been produced by the user moving in front of the robot while being tracked. Figure 4 top row shows the trajectories as observed by the tracker and the corresponding spline approximation. Figure 4 bottom row shows the simulated and the real robot trajectories for both the observed shapes.

Discussion: Among the mathematical shapes, both the square and the sheared square have discontinuities at the corners. For this reason the splines deviate significantly from the desired shapes when around the corners. This is due to the "smooth" nature of splines, which are not suitable to approximate a piecewise linear curve like a square. In order to avoid sharp turns at the corners, we opted for a slightly smoothed approximation with a λ parameter of 0.01 .

The average error along all of the mathematical shapes is low, as shown in Table 2. The only time the robot noticeably deviated from the desired trajectory was at the beginning of the spiral shape, as shown in Figure 3, top right. This is due to the initial high curvature of the spiral, which is not reproducible by the robot given its physical constraints. However, the feedback control law quickly corrected the error and drove the robot back on the desired trajectory. The same graph highlight the differences between the real trajectory and the simulated one, and the role physical constraints play in a robot controller.

The sheared square has been produced by shearing the square by 1m along the x axis. This shows how our proposed system can be used to interactively teach affine transformations.

The trajectories observed by the tracker are noisy and discontinuous in several points, as shown in Figure 4. Trajectory 2 benefited the most from the parameter λ , with the effect of obtaining a smooth non interpolating curve. This is reflected in the higher errors both the simulated and the real trajectories exhibit.

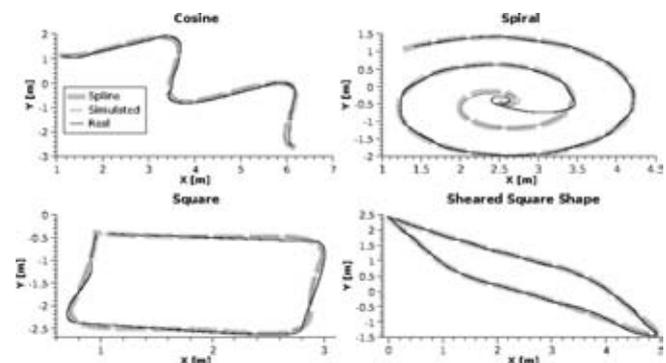


Fig.3 The mathematical shapes with the corresponding simulated and real robot trajectories. All the graphs are rotated to match the robot frame of reference.

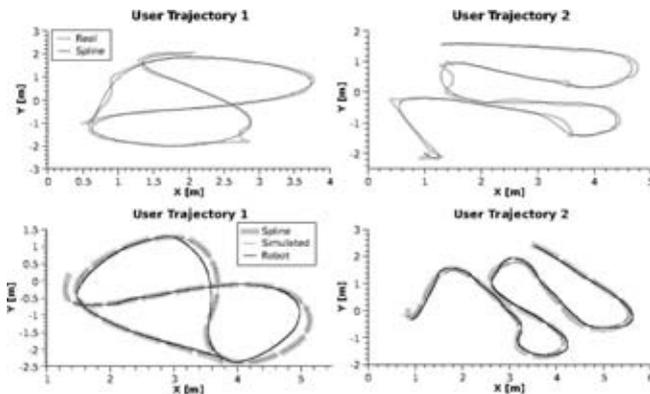


Fig.4 (Top row) The user produced trajectories observed by the tracker, and the corresponding spline approximations. (Bottom row) The simulated and real robot trajectories. All the graphs are rotated to match the robot frame of reference.

4. Conclusions and future work

In this paper we presented a robotic system to interactively teach basic geometry. The interaction with a student happens when the user selects a shape, and when the robot shows what the shape will look like when it will reproduce it, given the parameters chosen by the student. At the end of the interaction the robot describes the shape in the environment by moving along it.

The proposed system is composed of two main parts: a passive one, which detects and tracks people movements, and an active one, which plans the robot movements and drives it along a shape. We showed with several experiments that the system is reliably able to carry on both tasks.

In this paper we focused mainly on the application and on the results. In the future we will perform a survey among students and teachers to identify the main points where this system can be improved. We will also modify the controller so that it will include the robot physical constraints.

Acknowledgements

Dr. Riano is supported by the Centre of Excellence in Intelligent Systems (CoEIS) project, funded by Northern Ireland Integrated Development Fund and InvestNI.

This paper is dedicated to the memory of Prof. Ulrich Nehmzow, whose inspiration and ideas led to the development of this work.

References

[1] H. Abelson and A. A. Disessa, *Turtle geometry: The computer as a medium for exploring mathematics*. The MIT Press, 1986.

[2] T. Fujita, K. Jones, and S. Yamamoto, "Geometrical intuition and the learning and teaching of geometry," in *Topic Study Group on the teaching of geometry at the 10th International Congress on Mathematical Education*, 2004, p. 411.

[3] C. Laborde, C. Kynigos, K. Hollebrands, and R. Strasser, "Teaching and learning geometry with technology," in *Handbook of research on the psychology of mathematics education: Past, present and future*, 2006, p. 275304.

[4] K. Jones, "Issues in the teaching and learning of geometry," in *Aspects of Teaching Secondary Mathematics: perspectives on practice*, 2002, pp. 121–139.

[5] P. Viola and M. J. Jones, "Robust Real-Time face detection," *International Journal of Computer Vision*, vol. 57, no. 2, pp. 137–154, May 2004.

[6] K. O. Arras, O. M. Mozos, and W. Burgard, "Using boosted features for the detection of people in 2d range data," in *Proc. of the int. conf. on robotics & automation*, 2007.

[7] N. Bellotto and H. Hu, "Multisensor-Based human detection and tracking for mobile service robots," *IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICSPART B: CYBERNETICS*, vol. 39, no. 1, p. 167, 2009.

[8] M. Kobilarov, G. Sukhatme, J. Hyams, and P. Batavia, "People tracking and following with mobile robot using an omnidirectional camera and a laser," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2006, p. 557562.

[9] T. Poggio and F. Girosi, "Networks for approximation and learning," *Proceedings of the IEEE (special issue: Neural Networks I: Theory and Modeling)*, vol. 78, no. 9, p. 14811497, 1990. [

[10] A. Doucet, N. D. Freitas, and N. Gordon, *Sequential Monte Carlo methods in practice*. Springer Verlag, 2001.

[11] D. Schulz, W. Burgard, D. Fox, and A. B. Cremers, "People tracking with mobile robots using sample-based joint probabilistic data association filters," *The International Journal of Robotics Research*, vol. 22, no. 2, p. 99, 2003.

[12] S. Thrun, "Probabilistic algorithms in robotics," *AI Magazine*, vol. 21, no. 4, p. 93, 2000.

[13] C. de Boor, *A Practical Guide to Splines*. Springer-Verlag, 1978.

[14] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann, et al., "Stanley: The robot that won the DARPA grand challenge," *Journal of field Robotics*, vol. 23, no. 9, p. 661692, 2006. [

[15] S. Gulati and B. Kuipers, "High performance control for graceful motion of an intelligent wheelchair," in *IEEE International Conference on Robotics and Automation*, 2006, p. 39323938.

[16] H. Eren, C. C. Fung, and J. Evans, "Implementation of the spline method for mobile robot path control," in *Instrumentation and Measurement Technology Conference, 1999. IMTC/99. Proceedings of the 16th IEEE*, vol. 2, 1999, pp. 739–744 vol.2.

[17] P. H. C. Eilers and B. D. Marx, "Flexible smoothing with b-splines and penalties," *Statistical Science*, vol. 11, no. 2, pp. 89–121, 1996.

[18] E. Guechi, J. Lauber, and M. Dambrine, "On-line moving-obstacle avoidance using piecewise bezier curves with unknown obstacle trajectory," in *Control and Automation, 2008 16th Mediterranean Conference on*, 2008, pp. 505–510.

[19] J. H. Hwang, R. C. Arkin, and D. S. Kwon, "Mobile robots at your fingertip: Bezier curve on-line trajectory generation for supervisory control," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2003, p. 14441449.

[20] B. V. G., H. S. A., and J. D. S., "Auto guided vehicle control using expanded time b-splines," in *Systems, Man,*

and Cybernetics, 1994. 'Humans, Information and Technology', 1994 IEEE International Conference on, vol. 3, 1994, pp. 2786–2791 vol. 3.

[21] L. Lapierre, D. Soetanto, and A. Pascoal, "Nonsingular path following control of a unicycle in the presence of parametric modelling uncertainties," International Journal of Robust and Nonlinear Control, vol. 16, no. 10, p. 485504, 2006.

[22] C. G. Bianco, A. Piazzzi, and M. Romano, "Smooth motion generation for unicycle mobile robots via dynamic path inversion," Trans. Robot. Automat, vol. 17, p. 413422, 2001.

Dr. Lorenzo Riano

Intelligent Systems Research Centre
University of Ulster
Northland Road
BT487JL, Londonderry, UK
Phone: +44 (0)28 71375187
Fax: +44 (0)28 71375570
E-mail: l.riano@ulster.ac.uk

Prof. Martin McGinnity

Intelligent Systems Research Centre
University of Ulster
Northland Road
BT487JL, Londonderry, UK
Phone: +44 (0)28 71375616
Fax: +44 (0)28 71375570
E-mail: tm.mcginny@ulster.ac.uk

European Land Robot Trial (ELROB) Towards a Realistic Benchmark for Outdoor Robotics

Frank E. Schneider, Dennis Wildermuth, Bernd Brüggemann, and Timo Röhling

Abstract

The European Land Robotic Trial (ELROB), which was held for the fifth time in 2010, is designed to compare unmanned ground vehicles in realistic outdoor tasks. It addresses the need to create a benchmark that can reproducibly compare and evaluate different robot systems. While robot trials like the DARPA Grand Challenge or the RoboCup have proven to be adequate benchmarks to compare robots systems in specific scenarios, the ELROB provides benchmarking in a wide range of tasks, which are oriented at prospective use-cases from a large variety of applications. In this paper we describe the ELROB 2010, the rationale behind the scenario design and how the trial has been implemented. We present the benchmarking system used to evaluate the robots' performance in the different tasks and, finally, have a closer look at some exemplary results.

Keywords: robot contest; outdoor; benchmark.

Introduction

The European Land Robot Trial (ELROB) was designed to demonstrate and compare the capabilities of unmanned systems in realistic scenarios and terrains. It was founded by the European Robotics Group and is organised by the Fraunhofer Institute for Communication, Information Processing and Ergonomics (FKIE), formerly part of the Research Establishment for Applied Sciences (FGAN). The trial is held annually, alternating between a more military and a mainly civilian focus. Up to now, the so-called M-ELROB was held at the military school in Hammelburg, Germany, whereas the civilian C-ELROB is performed at changing locations throughout Europe.

One major aim of the ELROB is to get a deep insight into the field of ground robotics by testing existing solutions in practical trials. These trials are conducted with a focus on short-term realisable robot systems and are designed to assess current technology while solving real world problems. Thereby, scenarios are not limited to the abilities of today's robots, but focus on realistic missions demanded by experienced users in difficult environments.

The ELROB presents a variety of realistic user defined tasks. These tasks include, for example, security missions, convoying, or reconnaissance by day and night. Although robotic contests are widely accepted as valuable means for benchmarking real outdoor robot systems, it is generally a difficult task to compare results from different contests or to generate a reasonable ranking even within one of the quoted scenarios. Omitting all details of task design, it is still obvious that many different parameters might have an influence on the overall benchmark for a mission. Taking the convoying scenario as an example, average speed, totally driven distance, or degree of autonomy are only one possible choice from a wider range of feasible parameters. Each parameter has to be measured in a precise and reproducible manner, which often raises serious problems,

and afterwards has to be weighted in its influence on the final benchmark.

This paper will mainly address the latest ELROB, which took place from 17th until 20th of May 2010 in Hammelburg, Germany. We present the rationale behind the scenario design, the special demands of the co-organising military user, and the structure of the participants. After a detailed description of the different tasks of ELROB 2010, the remainder of the paper deals with the chosen benchmarking approach, thereby discussing the typical problems in the field of ranking systems, namely choice, measuring, and weighting of the different benchmark parameters.

1. Related Work

Generally, it is a difficult task to compare different published approaches in the field of robotics [1]. Thus, robot competitions are recognized as valuable benchmarks for real robot systems [2]. Several different competitions were held in the last years. Two of the largest and best-known competitions are the RoboCup [3] and the DARPA Grand Challenge [4], which are also recognized outside the robotics community.

While the RoboCup is currently targeted at indoor robots, the DARPA Grand Challenge aims to test and compare driverless cars. It started in 2004 with the rather simple task of following a 241 km long path, defined by several thousand UTM waypoints. Due to the difficult terrain and some teething problems, no participant was able to solve this task. In 2005, the task remained basically unchanged, and four participants successfully completed the race. In 2007, the DARPA Grand Challenge modified its goals from driving autonomously on difficult terrain to interacting with other vehicles in an urban scenario. Three teams could solve this very demanding challenge.

The ELROB is somehow comparable to the DARPA Grand Challenge in its attempt to gauge the functionality of outdoor

robots. However, as already mentioned the ELROB presents a wider choice of user defined tasks instead of only one single scenario. Different users often express completely different requirements and specifications for robot systems depending on the possible fields of application. Instead of combining demands into one large scenario, like in the DARPA Grand Challenge, it might be more meaningful to have different tasks, which correspond to the various application scenarios. The following chapters present an exemplary description of the tasks for ELROB 2010.



Fig.1 Overview of the movement oriented trials of ELROB 2010. The yellow track belongs to the approach part of the reconnaissance scenario, the purple one marks the mule scenario, the green, red, and blue tracks correspond to the different levels of the transport trial.

2. Tracks and Trials

The chosen area for ELROB 2010 lies within the training facility of the German military school in Hammelburg. Its size is of about nine square kilometres. The accessible roads have different qualities, ranging from well paved to heavy dirt roads. The environment is predominantly woody.

The different tracks on site were chosen to test specific aspects of robot deployment. Some challenges were common to all tracks; others were specific to certain scenarios. In preparation for the trials, every track was tested with respect to

- accessibility of the roads and paths,
- GPS reception, and
- radio reception between vehicle and control station.

By selecting areas with an elevation profile that does not support continuous radio communication from the control station, a certain level of autonomy was enforced. Thus, it was deliberately made difficult or even impossible to complete the missions in a purely remote-operated way.

Generally, the trials of ELROB 2010 have been divided into two major categories. On the one hand, there were scenarios with a focus on driving large distances of up to several kilometres. Due to the long distances in combination with the already mentioned hilly and woody character of the environment, it could be expected that solutions with a large degree of autonomy would perform best. Figure 1 presents an overview of the whole area. The different colours mark the different tracks and missions. In the following subsections, each scenario will be briefly described.

The second group of scenarios, on the other hand, had its focus on reconnaissance tasks. In these trials the robots had to search a given area, consisting of streets, paths, houses and grassland, for different kinds of targets, for example explosives, chemical or toxic waste, and radiation sources. Besides autonomy, other factors like manoeuvrability and a well-equipped sensor platform were of greater importance for this kind of tasks. The robots had to pass stairs and enter rooms through narrow doorways in order to reach all targets. In addition, although all targets could be seen with normal camera systems, additional hints like acoustic signals, heat or radiation sources had been installed for an easier identification. Therefore, systems with good sensor equipment had significant advantages during these trials.

Figure 2 illustrates some aspects of the reconnaissance scenarios. The leftmost picture shows a major part of the target area for these scenarios. From their starting point, the vehicles had to go there along some given, UTM-defined route. In the target area, open grassland with any kind of barricades, barriers or blockades had to be passed and different kinds of houses had to be entered and inspected. The middle picture presents an example target for the RSTA mission and the right one shows an exemplary radiation source in the NBC scenario. The details of the missions like the distances to be travelled and the exact kind of targets to be identified will be described in the following subsections.

2.1 Movement Transport Trial

Goal of this trial was to implement some kind of stable conveying in an outdoor, non-urban and off-road terrain with roads and paths ranging from asphalt streets to simple dirt roads in the forest. The convoy consisted of two vehicles of which only one was allowed to have a human driver. The second one had to be autonomous. The required path was defined by a small set of UTM waypoints, which were quite far from each other, so that the robot could not just drive straight lines between the waypoints but had to navigate along the roads and paths. The roads were part of the local testing ground for trucks, usually gravelled and led mainly through the forest. They were not marked, so mostly there was no clear distinction from the surrounding terrain. Sharp turns, dead ends and narrow passages occurred at several positions. No team member was allowed to inspect the trial area in advance.

The whole trial was divided into three levels of increasing difficulty, each consisting of a round trip with one common starting point. Looking at figure 1, one can identify the starting point at the connection of the green and blue track. The green track was the easiest one. The vehicles could use wide, well-paved roads, only with some very sharp turns to prove the robustness of the conveying algorithm. For the second level, a part of the original green route was replaced by a small dirt road in the forest; see the red track in figure 1. Level three, the blue track, was part of a special "off-road" truck testing site. Due to the very demanding character of this route and in contrast to the normally applied rules, the teams were allowed to have a look at the track in advance, in order to prevent possible harm from their vehicles. Each level was about 2.5 kilometres in length; the maximum time for completion of this trial was one hour.

2.2 Mule Transport Trial

The objective for this scenario was to let a vehicle serve as a "mule" and carry as much payload as possible between a loading and a turning point. Again, the terrain was woody and hilly with – partly very steep – roads of different quality. Instead of getting the UTM coordinates of the turning point



Fig. 2 Illustration of the reconnaissance scenario of ELROB 2010. Left part – picture of the surroundings, streets, paths, houses, and grassland. Middle part – an example target in the RSTA trail. Right part – an exemplary radiation source in the NBC scenario.

directly, the robots had to follow a human who guided the vehicle from the loading point to turning point once. To simplify the mission for the teams this leader could be one of the team members, who himself was then led by someone from the organizing personnel. Thus, the leader from the team could wear, for example, specially coloured clothes or use special gestures.

After reaching the turning point for the first time, the robot had to shuttle the payload between the two points as fast and as autonomously as possible. In figure 1, the mule track is marked in purple. It was not known to any team member in advance. The distance between loading and turning point was about two kilometres; the maximum time for completion of this trial was one hour.

2.3 Reconnaissance Trial – Approach (Day/Night)

In contrast to last years' approach, for ELROB 2010 the reconnaissance mission was split up into two independent parts. In the last years the objective was, first, to let the robot approach through unknown terrain into a designated target area and, second, search this target area for special, pre-defined targets. As already mentioned in the introduction for this chapter, the nature of these two subparts of a classical reconnaissance mission is rather different. The first part is more suitable for larger platforms with good and autonomous driving capabilities, whereas in the second part normally smaller robots with good manoeuvrability and special sensor equipment normally perform better. Consequently, nearly no participant was able to fulfil the complete trial during the former ELROB contests. Therefore, the organisers decided to separate the approach from the search in the target area.

Objective of the approach part of the reconnaissance scenario was now to reach a target point about three kilometres away. At that target point, an overview picture of a closely visible village should be taken. On its way towards the goal point, some intermediate waypoints had to be traversed. All these points were defined by their UTM positions. The yellow track in figure 1 marks one possible route for the approach, which passes all these intermediate waypoints. However, theoretically, the robots could choose their way freely. The track consisted of several narrow passages and even two dead ends, which can be identified by the small detours in figure 1. The area was completely woody and rather hilly, which notably complicated any attempt to maintain radio connection. The roads mainly consisted of forest paths with no clear distinction from the surroundings. As usual, no team member was allowed to inspect the area in advance. The maximum time for completion of this trial was one hour. The whole trial was first conducted under normal daylight conditions. The most

successful teams had the chance to repeat the identical mission during the night.

2.4 Reconnaissance Trial – Target Area (Day/Night)

The terrain for the reconnaissance missions in the target area was an urban area within a valley. The urban area consisted of small buildings and homesteads, which are spread sporadically over the grassland of the valley (see left picture of figure 1). The buildings were connected with small roads and footpaths. Barricades, barriers and other blockades occurred at several places. From their starting point at the border of the valley the robots had to move along a given, UTM-defined route into a specific target area about 300 metres away. The relevant area for inspection was defined by a set of UTM boundaries. As for all other missions, no inspection of the operational area was allowed in advance. The maximum time for the completion of a trial was one hour.

The participants could attend up to three different kinds of such reconnaissance missions, according to their specific sensor equipment. The main difference between these possible missions was the type of targets the robots had to search. In the more general "reconnaissance, surveillance and target acquisition" (RSTA) trail, targets could be suspicious persons and vehicles, weapons, barricades and blockades, but also special acoustic signals like weapon fire or agitated discussions or heat sources, for example from vehicles or fires. The middle part of figure one gives an example. Those numbered orange cones marked all targets. The letters on the small white sheet in the middle of the picture should have been readable in the images acquired by the robot. Some of the targets could be only acquired from distances of up to 500 metres.

The "nuclear, biological, and chemical" (NBC) reconnaissance scenario required special sensor equipment, because there was no distinct marker for the targets like the orange cones in RSTA. Instead, the special physical or chemical properties of the – simulated – chemical agents, toxic industrial chemicals, radiation sources or explosives had to be measured. Finally, during the "explosive ordnance reconnaissance" (EOR) mission the robot had to inspect along a pre-defined UTM route. The robot had to search for suspicious objects like possible Improvised Explosive Devices (IED), ammunition, explosives, or wires under, beside or on the road, but – of course – without touching them. For all three types of missions, imagery and exact position of each target had to be acquired and transmitted to the control station.

3. Participants

Ten teams in total participated in ELROB 2010, six teams came from European universities and four participants were from German and US industry companies. This section will shortly introduce their robots and vehicles.

The Institute of Real-Time Learning Systems of the University of Siegen took part with the robot AMOR. AMOR is a modified quad equipped with laser line scanners, PMD cameras and a stereo camera system. It uses a 3D environment model and fully featured local and global maps to drive autonomously, for example to follow a person or to pass given waypoints [5]. The Real Time Systems Group (RTS) of the University of Hannover participated with a robot called HANNA. Based on an off-the-shelf transport car, HANNA is equipped with various sensors for tele-operation, semi-autonomous operation and fully autonomous operation. The main sensors are two 3D laser range scanners used for environmental perception. In addition, multiple cameras, Differential-GPS, and inertial sensors are used for vehicle control. The Robotics Research Lab of the University of Kaiserslautern attended with their Robust Autonomous Vehicle for Off-road Navigation (RAVON). It is able to move fully autonomously, driven by a behaviour-based control system. It uses three 2D laser scanners and two custom-built stereo camera heads, as well as several additional sensors like GPS or a magnetic field sensor for localization purposes [6].

The Team MuCAR from the University of the Bundeswehr Munich (UBM) developed and operated the robot MuCAR-3. It is a modified Volkswagen Touareg, which allows computer control of steering, brake, throttle, and automatic gearbox. The team focuses on use of a Velodyne 3D laser scanner. The high definition 360 degree Laser Scanner is mounted on the roof of the vehicle. The RoboScout Team of the company BASE 10 SYSTEMS Electronics took part with the large robot GECKO. It is a four-wheel driven vehicle of about 3000 kg. Its speciality is its high manoeuvrability, because of its four separately steerable wheels. The robot can be controlled can be controlled via satellite or terrestrial communication, and can use a special small airplane as a relay station. The company Telerob presented their robot teleMAX. It is a track robot with flippers and a robotic arm. It is equipped with several cameras and is able to climb stairs. The team of Università degli Studi di Catania used a track driven vehicle that is used as an experimental research platform for volcano inspection. For autonomous navigation, the system is equipped with stereo camera-system, IMU, GPS and a SICK laser scanner.

	Transport Movement	Transport Mule	Recon. Approach
Degree of autonomy	1000	1000	1000
Total distance	100	--	100
No. of round trips	--	100	--
Total runtime	10	--	10
Delivery of digital map	1	1	1
Delivery of GPS log file	1	1	1

Tab. 1 Weights of the relevant mission parameters.

The University of Versailles used a new and self-developed robot. The team is based on a student project that used a commercial electro kid quad as chassis for the robot. While moving, the environment is perceived through a laser range

finder, sonars, infrared thermal sensors and webcams. The project addresses searching and rescuing people after natural disasters such as earthquakes. The company MacroUSA attended with a small Teleoperated UGV. The vehicle is equipped with a COFDM based vision system delivering a 360-degree view using three cameras. For navigation a GPS, IMU and a compass are included. However, the vehicle is not design to operate autonomously. The company ELP presented the PackBot, which was originally developed by IRobot. The tele-operated robot came in a basic version having on board only a camera and a manipulator.

4. Results

For the presentation of the results of ELROB 2010 and for the discussion of our benchmarking system we will consider only a subset of all the conducted trials. As already mentioned during the description of the scenarios, the trials could be divided into two categories, one that focuses on autonomously driving large distances and the other one that more concentrates on steering capabilities and specialised sensor platforms. Since from our point of view this first kind of missions is the more interesting and more important one for the robotics community, we will omit the results for those scenarios dealing purely with reconnaissance in the target area. Additionally, it can be stated that actually all vehicles in those trials acted fully tele-operated and none of them was equipped with any special sensor equipment apart from (high-resolution and sometimes heat image) cameras. A detailed examination of all omitted trials – including the missions at night – can be found at [7].

The evaluation of the remaining missions – the mule transport trial, the movement transport trail, and the approach part of the reconnaissance trial – concentrated on parameters which were clear to distinguish and easy to measure. Table 1 gives a short overview:

- **Movement Transport Mission**
Degree of autonomy, total distance driven, total runtime, delivery of a digital map and a GPS log file of the vehicle's track.
- **Mule Transport Mission**
Degree of autonomy, number of successfully completed round trips between starting and turning point, delivery of a digital map and a GPS log file.
- **Reconnaissance Mission – Approach**
Degree of autonomy, total distance driven, total runtime, delivery of a digital map and a GPS log file.

Obviously, autonomy was of overwhelming relevance to achieve a good result. In contrast to the other influencing factors, for which it is clear how they can be counted or measured, our definition of autonomy has to be explained. We used the ratio of total driving time and the so-called "manual interaction time", which starts at the moment when anyone interacts in any way directly with the vehicle or, for example, via an operation console. It ends in the moment when this interaction is over and the vehicle continues its autonomous work. The measurements for all the influencing parameters are normalized into the range [0; 1] and afterwards multiplied by the factors from table 1, leading to a team's total sum for each mission.

3.1 Movement Transport Trial

Unfortunately, the movement trial suffered from very heavy rain. Due to the weather conditions, two of the registered teams, the University of Kaiserslautern and the University of

Hannover, withdraw their participation. The University of Siegen and their robot AMOR managed the easiest first track without problems and could follow the leading car without any necessary interaction at an average speed of 5.8 km/h. However, at the rougher terrain of the second level the robot had considerable problems and often had to stop. As a result, the maximum trail time of 60 minutes ended after about 1200 metres of the second track.

The MuCAR-3 from the University of the Bundeswehr Munich performed very well and completed the first two levels of the trial at an average speed of 14 km/h and without any intervention of their safety driver. The team tried – without evaluation – even the very demanding third level and finished it with only one necessary stop. The third and last participant, the robot GECKO of the company BASE 10, also managed the easiest first track, but with some interaction, especially at sharp turns. Afterwards, the team aborted the mission because they feared damage for their vehicle due to the expected worse road conditions in the next levels.

Before looking at table 2 for the numerical results of this trial, it is important to mention, that the MuCAR team regrettably could not be evaluated because their mission setup was not compliant to the rules of ELROB 2010. For safety reasons they insisted on a human driver inside their car, who had to observe and – in case of need – control the actions of the robot. Therefore, the team started out of evaluation. As a result, official winner of this scenario was the University of Siegen, followed by the GECKO of BASE 10.

Team	Robot	Result	Rank
University of Siegen	AMOR	1011	1.
BW University of Munich	MuCAR-3	n.e.	n.e.
BASE 10 SYSTEMS	GECKO	648	2.
University of Hannover	HANNA	n.p.	n.p.
University of Kaiserslautern	RAVON	n.p.	n.p.

Tab. 2 Results of the movement transport trial.

3.2 Mule Transport Trial

The mule scenario started with the robot RAVON of the University of Kaiserslautern. Unfortunately, the team had technical problems, which forced the robot to stop after only a few metres. The second participant, HANNA of the Hannover University, successfully managed to follow its human leader and reached the turning point without larger problems. During the following autonomous shuttle mission, the robot had problems reaching the starting point again due to some technical faults. Shortly before the trial time was over, the team had to give up, only a few metres before arriving at the starting point again. However, corresponding to the benchmark parameters and since the vehicle was running autonomously most of the time, this result lead to the first rank in this scenario.

The University of Siegen and their robot AMOR reached the second place with a slightly worse performance. For a shorter totally travelled distance of about 2600 metres, the team needed longer and more frequent manual interaction with the system. The GECKO of BASE 10 SYSTEMS only drove a few hundred metres and then lost its way in the forest. Table 3 shortly presents the results of the mule

transport trial. The team of the Munich Bundeswehr University was not evaluated for the same reasons as explained in the last section. Nevertheless, it is worth mentioning that the MuCAR-3 managed to shuttle between starting and turning point several times nearly without any intervention of the safety driver.

Team	Robot	Result	Rank
University of Kaiserslautern	RAVON	206	4.
University of Hannover	HANNA	1000	1.
University of Siegen	AMOR	561	2.
BW University of Munich	MuCAR-3	n.e.	n.e.
BASE 10 SYSTEMS	GECKO	383	3.

Tab. 3 Results of the mule transport trial.

3.3 Reconnaissance Trial – Approach

The scenario design for the approach part of the reconnaissance mission required a high degree of autonomy, because the very hilly and woody terrain made a permanent radio connection between vehicle and control station nearly impossible. Nevertheless, two teams tried to run fully tele-operated by using a fibre optic cable. The first of them, the company Telerob with their small robot teleMAX, reached the first dead-end about 500 metres away from the starting point. While trying to turn it cut the cable, which meant an immediate end of the trial. The GECKO of BASE 10 Systems performed better, because after the loss of the fibre optic cable it made use of the special radio relay airplane for transmitting the signals. However, due to the nature of the benchmarking system with its special emphasis on autonomy, it is clear that a tele-operated approach could not lead to good rankings.

The University of Kaiserslautern had to withdraw the participation because of technical problems with their robot RAVON. The remaining teams, the University of Siegen with the robot AMOR and HANNA from Hannover University, both tried an autonomous approach. AMOR only moved about 600 metres and then stopped due to some problems with a very steep forest path. When trying to free the robot, it unfortunately had a complete system crash. HANNA and the team from Hannover autonomously reached a distance of nearly two kilometres, but then faced some serious orientation problems. Since the team had no means of communication between the vehicle and its control station over this far distance, it had to give up at this point. Accordingly, table 4 reflects the results of this trial. HANNA was the winner, and the University of Siegen reached a second place, although the GECKO drove faster and further, because of their autonomous approach.

It should be obvious, even through this very broad overview on the results of ELROB 2010, that the presented benchmarking system can be considered as fair only with regard to the special requisition of autonomy. Looking at the contest with a slightly different accentuation – for example on manoeuvrability, velocity or sensor equipment – immediately leads to different results. An appropriate extension and modification of the weighting parameters in table 1 might change the resulting ranks completely. Thus, it is important to keep in mind that a benchmark for robotic contests only reflects the organisers' demands and cannot reflect a robot's general suitability for outdoor robotic tasks.

Team	Robot	Result	Rank
Telerob	teleMAX	368	4.
University of Siegen	AMOR	877	2.
University of Kaiserslautern	RAVON	n.p.	n.p.
University of Hannover	HANNA	1005	1.
BASE 10 SYSTEMS	GECKO	374	3.

Tab. 4 Results of the approach part of the reconnaissance trail.

Conclusions and Future Work

The purpose of ELROB is not to get an overview over technological possibilities but to test outdoor ground robots in real world scenarios without regard to current limitations of these systems. Thus, the scenarios had to show the gap between desired and possible applications for today's robots. As could be expected, not every participant could cope with the designed missions. So the results were not unexpected and definitely not disappointing. In retrospect, two main problems could be singled out – reliable hardware, including reliable communication, and innovative autonomous software controller.

It is noticeable that while the industry generally had hardware in excellent quality available, they lacked the innovative autonomous control algorithms developed by the university teams. On the other hand, the university teams had most of their problems due to their restrained hardware budget and the required trade-off between functionality and cost. The combination from the robots used by industry and state-of-the-art control algorithms developed at universities might achieve much better results.

From the 20th to the 24th of June 2011, the sixth European Land Robot Trials will take place in Leuven, Belgium [7]. The current working title is "Robotics in Security Domains, Fire Brigades, Civil Protection, and Disaster Control". Therefore, the missions will be designed having typical scenarios of those fields of application in mind. Again the trials will be designed to present scenarios as close to real world applications as possible.

References

- [1] DEL POBIL, A. P.: "Why do we need benchmarks in robotics research?", in Proceedings of the IROS-2006 Workshop on Benchmarks in Robotics Research, Beijing, October 2006.
- [2] BEHNKE, S.: "Robot competitions - Ideal benchmarks for robotics research", in Proceedings of the IROS-2006 Workshop on Benchmarks in Robotics Research, Beijing, October 2006.
- [3] KITANO, H., ASADA, M., KUNIYOSHI, Y., NODA, I., OSAWA, E., AND MATSUBARA, H.: "RoboCup: A Challenge Problem for AI", AI Magazine 1997.
- [4] THRUN, S., MONTEMERLO, M., DAHLKAMP, H., STAVENS, D., ARON, A., DIEBEL, J., FONG, P., GALE, J., HALPENNY, M., HOFFMANN, G., LAU, K., OAKLEY, C., PALATUCCI, M., PRATT, V., STANG, P., STROHBAND, S., DUPONT, C., JENDROSSEK, L., KOELEN, C., MARKEY, C., RUMMEL, C., VAN NIEKERK, J., JENSEN, E., ALESSANDRINI, P., BRADSKI, G., DAVIES, B., ETTINGER, S., KAEHLER, A., NEFIAN, A., AND MAHONEY, P.: "Stanley: The robot that won the DARPA Grand Challenge", Journal of Field Robotics, vol. 23, no. 9, pp. 661-692, June 2006.
- [5] SEEMANN, W. AND KUHNERT, K.-D.: "Design and realisation of the highly modular and robust autonomous mobile outdoor robot AMOR", In Proceedings of the 13th IASTED International Conference on Robotics and Applications, Würzburg, Germany, August 2007.
- [6] PROETZSCH, M., BERNS, K., SCHUELE, T., AND SCHNEIDER, K.: "Formal Verification of Safety Behaviours of the Outdoor Robot RAVON", in Proceedings of the 4th International Conference on Informatics in Control, Automation and Robotics (ICINCO), pp. 157-164, Angers, France, May 2007.
- [7] European Land Robot Trials (ELROB) website at <http://www.elrob.org>

Frank E. Schneider

Research Group "Unmanned Systems"

Fraunhofer Institute for Communication, Information Processing and Ergonomics (FKIE)

Neuenahrer Str. 20, 53343 Wachtberg, Germany

frank.schneider@fkie.fraunhofer.de

On an educational approach to behavior learning for robots

Michel Tokic, Arne Usadel, Joachim Fessler, Wolfgang Ertel

Abstract

This paper introduces a system for teaching biologically-motivated robot learning in university classrooms that might be used in courses such as *Artificial Intelligence* and/or *Robotics*. For this, we present a simple hardware robot that is able to learn a forward walking policy on basis of a reinforcement signal. Students are able to conduct experiments on a PC with a software called the Teachingbox that controls the robot. This software offers the possibility to control the learning method's parameters throughout the learning process, which allows observing the effects of such parameters on a real robot. Furthermore, learning on the hardware robot is very fast since forward-walking policies are usually learned in about 30 seconds. Due to this quick learning process nearly no waiting time is caused, and in return this fact often impresses the audience and leads to the question: „How does it work?“

Keywords: higher education, robots, behavior learning, reinforcement learning

Introduction

As robots or the environment of a robot become more and more complex, the way of programming robots in the classical supervised way also becomes more difficult. As a consequence, engineers often program just a “working” behavior of a robot, but which can be far away from an “optimal” behavior, e.g. movements of a robot that maximize the forward walking velocity. One possible solution to this general problem is offered by learning behaviors from scratch—in the same way as humans or animals do—instead of manually programming the robot. In literature, learning in such way is called *trial-and-error learning* which has been first studied in the domain of psychology and animal learning [1]. Nowadays, the research domain of *reinforcement learning* (RL) [2] aims to mimic *trial-and-error learning* in a machine-learning approach based on a reward signal (or reinforcement signal) which strengthens or weakens action selections in certain situations with the goal of maximizing the cumulative reward. Since robot learning is just one possible application of RL, the knowledge about this research domain broadens an engineer's skill on behavior programming that can also be applied to other applications.

Neller et. al. said: “*Simple examples are teaching treasures. Finding a concise, effective illustration is like finding a precious gem. When such an example is fun and intriguing, it is educational gold.*” [3]. At the University of Applied Sciences Ravensburg-Weingarten, we were looking for such kind of illustration that enables to teach RL within a narrow time-slot of about four lessons of an *Artificial Intelligence* introductory course. Within that given time-slot, we introduce the value-iteration and Q-learning algorithms on discrete state and action spaces and explain the exploration/exploitation problem. In order to explain to the students the field of RL, we found a crawling robot with a simple two-DOF arm as sketched in Figure 1 appropriate that has been proposed by Kimura et al. [4] and built in hardware by Tokic [5]. Furthermore, the reason why we also favor a robot instead of a theoretical problem is due to the

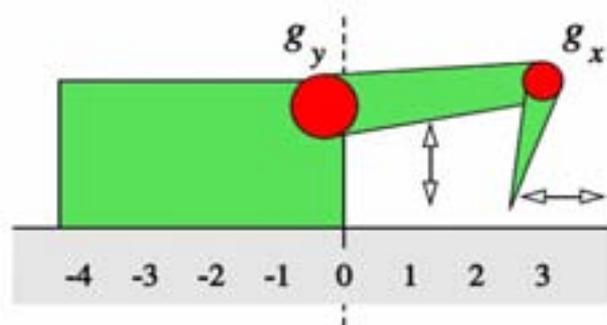


Fig 1: A model of the crawling robot with its two joints g_x and g_y .

fact that robots seem to be encouraging motivators for students as also recently reported by Kay [6].

In case of discrete positions and small movement angles of the joints, the state space of the robot arm can be approximated by a grid world. In order to move forward, the robot has to repeatedly perform a cycle of moves as shown in Figure 2 or in the sequence shown in Figure 3. The task for the learning algorithms is to find a policy (which might be such a cycle) that maximizes the cumulative reward. For this, the reward is the speed of the robot, i.e. the distance that the body of the robot moves forward per time step. Consequently, a move forward gives positive reward whereas any backward move yields negative reward.

In the following we describe the robots architecture and elaborate on simple experiments that students can conduct with a software called the “Teachingbox” [7], which is an open-source framework written in Java. By using this software tool, students are (1) able to send action commands to the robot in order to observe the robot's behavior and (2) are also able to learn policies on the basis of observed rewards from the robot's environment.

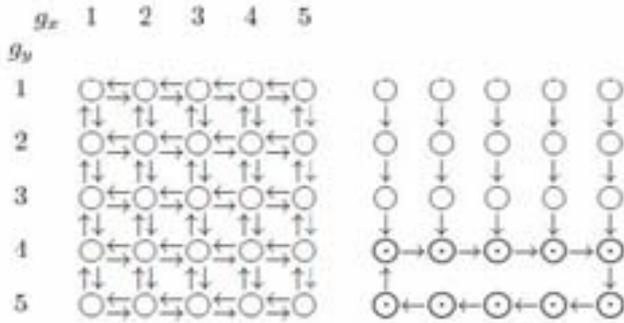


Fig 2: The 5x5 grid-world model (left) and a cyclic walking policy (right). States within the cycle are bold marked.

FOUR STEPS OF A SIMPLE CYCLIC FORWARD-WALKING POLICY.

robot	time	state		reward	action
	t	g_y	g_x	x	a_t
	0	up	left	0	right
	1	up	right	0	down
	2	down	right	0	left
	3	down	left	1	up

Fig 3: Four steps of a simple cyclic forward-walking policy.

1. Hardware Robot

A prototype of the robot that we use in our “Laboratory on Artificial Intelligence” is depicted in Figure 4. Basically, this robot is controlled by an ATmega32 microcontroller board that is mounted on top of the robot. This board controls the joints of the robot which are driven by Dynamixel AX-12 actuators. These servos communicate with a half-duplex asynchronous packet-protocol on TTL-level with up to 1,000,000 bps. The maximum holding torque is about 1.17 Nm.

The speed of the robot is measured by an optical incremental encoder that is connected via a non-slip belt transmission to a (rigid) wheel axle. The controller board also comes up with outlets for the servos, an outlet for the encoder and a DIP switch for setting up several parameters. For instance, one of these parameters inverts the encoder signal and results the robot to learn a backward-moving strategy instead of moving forward.

On top of the controller board, there also exists a RF04 ER400TRS serial transceiver module, which is used for communication with the Teachingbox software on the PC side. This module is directly attached to the ATmega’s serial port and operates by a speed of 19,200 baud. On the PC side we use a RF04 USB telemetry module for communicating with the controller board over a standard RS232 COM port.



Fig 4: The crawling robot we use in our laboratory tutorials.

2. Reinforcement Learning

We consider the reinforcement learning framework [2] where an agent interacts with a Markovian decision process (MDP). At each discrete time step, $t \in \{0, 1, 2, \dots\}$, the agent is in a certain state $s_t \in S$ —for example, the angular position of the robot’s joints. After the selection of an action, $a_t \in A(s_t)$, the agent receives a reward signal, $r_{t+1} \in \mathbb{R}$, from the environment and passes into the successor state s_{t+1} . The decision which action is selected in a certain state is characterized by a policy, $\pi(a) = a$, that could also be stochastic: $\pi(a|s) = Pr(a_t = a | s_t = s)$. A policy that maximizes the cumulative reward over time is denoted as π^* .

In practice, there exist several approaches by which a policy for the robot can be learned. For this, we recently proposed using the value-iteration algorithm [8] with an online model-learning of the environment in parallel which was derived from the *Dynamic Programming* approach. In order to save additional memory required by the model-learning task, we now propose using a different learning algorithm within this paper that belongs to the family of *Temporal-Difference Learning* methods.

In order to learn an optimal policy π^* for the robot, we use Watkins’ Q-learning algorithm [9] as depicted in Algorithm 1. This algorithm basically works by assigning a numerical value to each state-action pair (s, a) , where each state-action value, $Q(s, a) \in \mathcal{Q}$, is an estimate of the expected cumulative reward, R_t , for following the current policy by starting in state s and taking action a :

$$\begin{aligned}
 Q_\pi(s, a) &= E_\pi \{ R_t | s_t = s, a_t = a \} \\
 &= E_\pi \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s, a_t = a \right\}
 \end{aligned} \tag{1}$$

where $0 < \gamma < 1$ denotes a discounting factor that specifies the influence of rewards received more far in the future. Furthermore, the parameter $0 < \alpha < 1$ specifies a learning rate that determines how much the value-function estimate is being adapted w.r.t. to the current *temporal-difference error*:

$$\delta = r + \gamma \max_{b \in A(s')} Q(s', b) - Q(s, a) \tag{2}$$

The affect of both algorithm parameters on the learning process is explored by the students during the conduction of experiments as described in Section 4.

Algorithm 1 Q-LEARNING ON ROBOT WITH ϵ -GREEDY

```

1: Initialize  $Q$  arbitrarily, e.g.  $Q(s, a) = 0$  for all  $s, a$ 
2: Initialize start state arbitrarily, e.g.  $s \leftarrow (g_x = 1, g_y = 1)$ 
3: loop
4:    $\zeta \leftarrow \text{rand}(\{0, 1\})$ 
5:   if  $\zeta < \epsilon$  then
6:      $a \leftarrow \text{random action from } \mathcal{A}(s)$ 
7:   else
8:      $a \leftarrow \text{argmax}_{b \in \mathcal{A}(s)} Q(s, b)$ 
9:   end if
10:  select action  $a$ 
11:  observe reward  $r$  and successor state  $s'$ 
12:   $a^* \leftarrow \text{argmax}_{b \in \mathcal{A}(s')} Q(s', b)$ 
13:   $\delta \leftarrow r + \gamma Q(s', a^*) - Q(s, a)$ 
14:   $Q(s, a) \leftarrow Q(s, a) + \alpha \delta$ 
15:   $s \leftarrow s'$ 
16: end loop

```

The robot's action selection policy, which is based on the Q-function learned throughout the interaction with the environment, works as follows. Since the robot is faced with an unknown environment after switching it on, a tradeoff between exploration (long-term optimization) and exploitation (short-term optimization) has to be done [2, 10]. A very simple and commonly used technique for this is ϵ -Greedy exploration [9], where at each time step the agent selects an action at random with probability $0 \leq \epsilon \leq 1$ (exploration). With probability $1 - \epsilon$ (exploitation) the agent selects an action that is greedy with respect to the current value-function estimates:

$$\pi(s) = \begin{cases} \text{random action from } \mathcal{A}(s) & \text{if } \xi \leq \epsilon \\ \text{argmax}_{a \in \mathcal{A}(s)} Q(s, a) & \text{else} \end{cases} \quad (3)$$

where $0 \leq \xi \leq 1$ is a uniform random number drawn at each time step. If there is more than one action having the highest estimated value in the current state, then a random action of this set of best actions is chosen.

In order to speed-up learning, a commonly used approach is to reduce the exploration rate ϵ over time. In this case ϵ is set to a high value at the beginning of the learning process which is decreased by a constant fraction at each time step. This results that the agent is more explorative at the beginning of the learning process, when the environment knowledge is unknown, as later the agent becomes pure exploitative. The final outcome of the learning algorithm where the robot interacted some time with the real world is shown in Figure 5.

3. The Teachingbox

When students should learn to understand the behavior of an algorithm, it is didactic supportive to perform experiments with a simple demonstrator. With such it should be possible to easily play with, e.g. in terms of algorithms parameter variations which enable to observe the affect of such parameters on the learning progress. Furthermore, such a demonstrator should also be usable without much effort in order that students focus only on relevant things.

Our recently presented software framework, the Teachingbox (TB) [7], aims at providing a rich library of implemented algorithms for robot learning in a universal

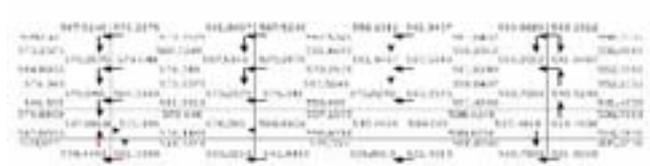


Fig 5: The Q-values and (greedy) policy learned by Q-learning from a real-world interaction of the walking robot (learned with $\gamma=0.99$). The corresponding rewards are shown in Figure 6.

robot learning framework. Hereby, the main purpose of this open-source Java framework is to support the development of autonomous agents with learning capabilities. The TB comes up with algorithms for RL, Learning-by-Demonstration, the possibility of manually programming policies and a build-in grid-world editor for modeling simple two-dimensional grid worlds. In particular, the RL-part of the TB currently consists of implementations of the most popular learning algorithms such as value-iteration, Q-learning and SARSA-learning with the support for Softmax action selection and ϵ -Greedy policies [2]. Furthermore, the TB also supports eligibility traces as well as gradient-descent learning of value functions, e.g. by CMACs or radial basis function networks. In order to visualize the learned behavior of an agent, the TB also provides a plotting library for value functions and learned policies.

In our "Laboratory on Artificial Intelligence" students conduct experiments with the TB and the crawling robot by writing simple Java programs. A typical program code that demonstrates the usage of the TB with learning on the hardware robot is depicted in Algorithm 2. At first, a Q-function with tabular approximation is instantiated. Then, the environment (the hardware robot) and the policy to be used are specified. Finally, the Q-learning algorithm (learner) is configured, attached to the agent and a new experiment is started for 1 episode with 300 time steps.

Immediately after the experiment is started, the TB's grid-world editor appears to the user that visualizes the current state of the robot and the rewards observed from the environment in real time, (Figure 6). Furthermore, also a policy window appears (the upper window of Figure 6) in which the user is able to configure the exploration/exploitation policy to be used by the agent. Additionally to the standard policies such as ϵ -greedy and Softmax, the user can also control the robot "by-hand" when selecting the "Human-Trainer" policy. With this, the robot's actions are controlled by the cursor keys (up, down, left, right) whereby it's also possible to select the "Greedy" action with respect to the currently learned Q-function.

It is easy to adapt the Java code for the use with other environments. For example, if learning should be based in an arbitrary $m \times n$ grid-world environment modeled by the user, then only line 2 of Algorithm 2 needs to be adapted to:

```

GridworldEnvironment env =
  new GridworldEnvironment( m, n );

```

which simply replaces the agent's environment and also demonstrates the flexibility of the Teachingbox which is based on the use of Java Interfaces. This approach standardizes methods of policies, environments and learners with the goal of being interoperable with each other. For example, each environment in the TB implements an `Environment` interface that standardizes important methods such as:

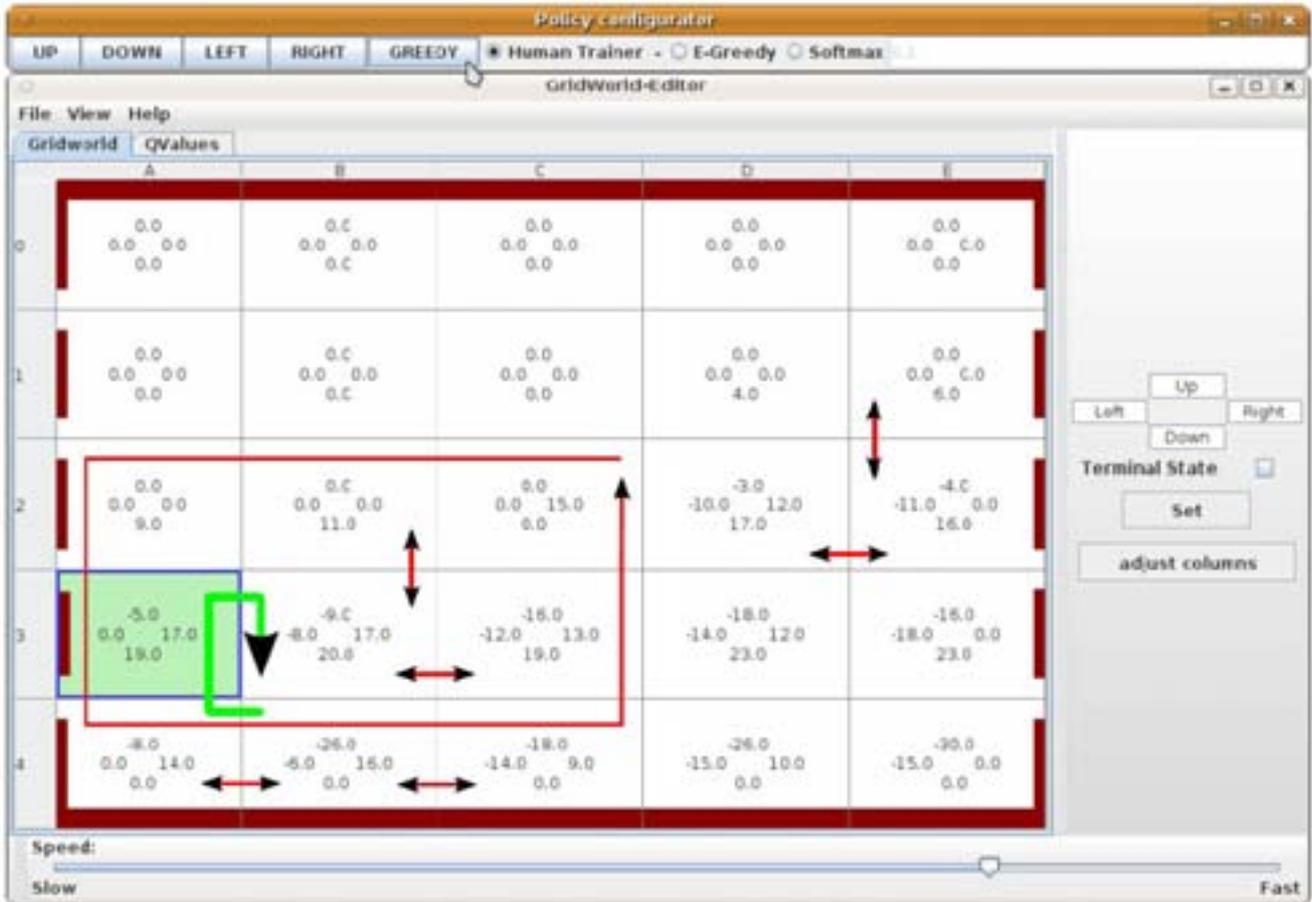


Fig 6: This figure shows the grid-world editor of the Teachingbox where on top of the window the user is able to configure the policy. Numbers in cells indicate the reward $r(s, a)$ observed from the environment. The cycle **A3→B3→B4→A4** indicates the optimal cycle having an average reward of $r = (17+20-6-8)/4 = 5.75$ per action. All other marked cycles indicate examples of sub-optimal cycles that have a lower average reward/action compared to the optimal cycle. The cell having the surrounded border (A3) indicates the current state of the robot.

- double doAction(Action)
- State getState()
- boolean isTerminalState()

where State and Action are double vectors in order to be compatible with each component of the Teachingbox.

Another example for the use of interfaces are policies that have to implement a Policy interface in order to standardize the methods:

- Action getAction(State)
- Action getBestAction(State)

4. Experiments with the robot and the Teachingbox

In the laboratory tutorial on RL, the first task for the students is to model the rewards of a simulation of the crawling robot by using the TB's grid-world editor. After the modeling of an environment, the policy must be learned by a learning algorithm such as Q-learning, Sarsa or value iteration. During this process, students have to conduct several experiments with variations of the learning algorithm parameters α and γ as well as with the policy parameter ϵ . These experiments lead to the observation that, for example, the discounting-factor γ has an important influence on the quality of learned policies. For example, if γ is chosen very small, then the agent is more near-sighted and takes

not rewards into account received from actions more far in the future, and which often results in learning sub-optimal policies. In comparison, large settings of γ make the agent more far-sighted, but in turn to this, the speed of learning can also be slowly at the beginning of learning since the convergence of the Q-function requires more transition experiences.

While learning the Q-function, the TB can memorize the function on the computer hard-disk, which enables reusing it in other experiments. After the successful learning of the Q-function, students have to evaluate the learned policy from the simulated environment on the real hardware robot. For this, the GridWorldEnvironment in the Java code has to be replaced by the CrawlerEnvironment. Furthermore, the exploration/exploitation policy for the robot has to be a pure greedy policy ($\epsilon=0$), which results that in a given state the action with the highest Q-value is selected. It is important that throughout this experiment learning is disabled in the Java-code, i.e. no Learner is attached to the Experiment. This results that only the policy of the simulated environment runs on the hardware robot. The idea behind this approach is to enable observing how well the environment has been modeled by the students and how the resulting policy looks like by observing the robots behavior. Therefore, after the selection of an action, the robot transmits the actual state as well as the reward for the most recent selected action to the Teachingbox that visualizes these values in the grid-world editor.

Algorithm 2 SIMPLE Q-LEARNING JAVA EXPERIMENT

```
1: // initialize new Q-Function with  $Q(s,a)=0$  by default
   TabularQFunction Q = new HashQFunction(0);
2: // establish serial robot link (baudrate, port)
   CrawlerEnvironment env =
     new CrawlerEnvironment(19200, "/dev/ttyUSB0");
3: // setup policy configurator
   PolicyConfigurator pi =
     new PolicyConfigurator ( Q,
       CrawlerEnvironment.ACTION_SET);
4: // create agent
   Agent agent = new Agent(pi);
5: // setup experiment with 300 time steps
   Experiment experiment =
     new Experiment(agent, env, 1, 300);
6: // setup Q-function learner
   TabularQLearner learner = new TabularQLearner(Q);
   learner.setAlpha(0.3);
   learner.setGamma(0.9);
7: // attach learner to agent
   agent.addObserver(learner);
8: // start experiment
   experiment.run();
```

Next, students conduct experiments with the environment of the real hardware robot. In this experiment the `Learner` has to be attached to the `Experiment` again, which enables learning from the robot's environment instead of learning from the simulated environment. Again, each state and reward received from the robot is visualized in the grid-world editor.

Throughout the experiment with the hardware robot, the students task is to vary the learning rate α of the Q-learning algorithm that determines how fast the learner adapts the Q-function with respect to the current TD-error δ . The understanding of this parameter is especially important, because the robot interacts in a non-deterministic environment with a noisy reinforcement signal due to the sensor and which also varies due to irregularities of the robot's surface. On the one hand, a large setting of the learning rate causes a fast adaption to environmental changes, for example, when the hardware robot walks from tar to grass. But on the other hand, large settings of α can also be problematic because sensor noise as part of the reinforcement signal may cause the robot to leave a learned "optimal" cycle. In contrast, when the learning rate is relative small, learning of policies takes more time due to the slow adaption of the (more accurate) Q-function.

The last experiment evolves the understanding for the need of balancing exploration and exploitation, which is also conducted on the real hardware robot and where students vary the policy parameter ϵ of the ϵ -greedy method (policy configurator on top of Figure 6). As a result, students will find out that without any exploration, i.e. $\epsilon=0$, the agent is very likely to stick in sub-optimal cycles of the state space (due to local minima of the Q-function) that in sum yield to a relative small cumulative reward than compared with the optimal cycle. Such a sub-optimal behavior is observable as the forward walking velocity of the robot that might be significantly slower as in comparison to the optimal cycle. The reason for this behavior on the hardware robot is often due to the fact that the robot hasn't walked through every

state transition of the state space and thus actually doesn't know about other (better) cycles. If such behavior occurs throughout the learning process, one can simply solve this misbehavior in the TB by increasing the exploration parameter ϵ in order that the robot also tries other actions which are not greedy with respect to the Q-function. Furthermore, one may also use the "Human-Trainer" policy and guide the robot into parts of the state space which haven't been explored yet.

Finally, after the experiments with Q-learning, students can perform the same kind of experiment with other learning algorithms, for example, with value iteration or Sarsa and then compare the speed of learning.

5. Educational Experiences & Conclusions

In order to get an overview whether or not the robot is a good demonstrator for reinforcement learning, we asked our students to participate in a pilot survey. Until the deadline of this paper, the online questionnaire was filled out by 5 of 10 students which represent 50% of the participants of our latest AI course. Because of this low return of responses, the quality of the following results indicates just a rough direction and must be seen as preliminary.

The results of our questions reveal that the students are of the opinion that the robot is a good object of study in general and it seems that the understanding of how reinforcement learning works and how learning method parameters affect the robot's speed (policy quality) got conveyed. Despite of being a good demonstrator, the students also remarked that the robot's hardware is still not comprehensive enough. From our point of view, this answer is not surprising since we cover reinforcement learning in about four lessons (each 1.5 hours), where we're limited in teaching just a small scope of the overall research field, i.e. we just explain discrete state and action spaces and do not consider the continuum. Furthermore, the problem of delayed rewards that exists in real world applications (e.g. the outcome of board games) is not given by the robot example. Anyway, interested students may write their own environments within the Teachingbox, which is possible due to the public availability of the source code on SourceForge. Alternatively, students may also play with standard use-cases such as the mountain-car or inverse-pendulum problem, which are already implemented in the Teachingbox.

The results also indicate that the students sustainably enhanced their skills on differentiating between the two main algorithms we convey: value-iteration and Q-learning. In turn, we couldn't obtain the same good result on the importance of the exploration/exploitation problem. Nevertheless, the students' feedback on both the algorithms and the exploration/exploitation problem was still positive and so we achieved our main goal: To present reinforcement learning in a lively, interesting manner and to support the students learning success. Finally, the students were enthusiastic to see that a behavior which has been learned in a simulation could be transferred to a real robot.

From our teachers' point of view, the robot demonstrator is a versatile instrument which greatly enhanced our lessons on reinforcement learning in order to present behavior learning for robots to the students. With the Teachingbox, we have a reliable software tool that also supports more complex hardware demonstrators that eventually will be constructed in the future. Furthermore, we also provide the robot's construction plans, printed circuit board diagrams and

videos on our website¹ and thus enable other interested institutions to rebuild the robot by themselves.

Acknowledgements

The authors like to thank Markus Schneider, Richard Cubek, Tobias Fromm, Tobias Bystricky, Haitham Bou Ammar and Andy Stumpe from University of Applied Sciences Ravensburg-Weingarten which were involved by the co-development of the Teachingbox. The authors also thank Günther Palm, Mohamed Oubatti and Friedhelm Schwenker from Ulm University for the scientific discussions on the crawling robot and also acknowledge Philipp Ertle from University of Applied Sciences Ravensburg-Weingarten for proof reading and discussing a first draft of the paper.

References

- [1] THORNDIKE, E. L.: Animal Intelligence. The Macmillan company, New York.
- [2] SUTTON, R. S., BARTO, A. G.: Reinforcement Learning: An Introduction, Cambridge, MA: MIT Press, 1998.
- [3] NELLER, T. W., PRESSER, C. G. M., RUSSEL, I., AND MARKOV, Z., Pedagogical possibilities for the dice game pig, Journal of Computing Sciences in Colleges, vol. 21, no. 6, pp. 149–161, 2006.
- [4] KIMURA, H., MIYAZAKI, K., AND KOBAYASHI, S., Reinforcement learning in POMDPs with function approximation, In: Proceedings of the 14th International Conference on Machine Learning (ICML'97). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1997, pp. 152–160.
- [5] TOKIC, M., Entwicklung eines lernenden lauroboters, Diploma thesis, University of Applied Sciences Ravensburg-Weingarten, Weingarten, Germany, 2006.
- [6] KAY, J., Robots as recruitment tools in computer science: The new frontier or simply bait and switch?, In: Educational Robotics and Beyond: Design and Evaluation. AAAI Press, Tech. Rep., 2010, pp. 26–30.

¹ <http://ailab.hs-weingarten.de/>

- [7] ERTEL, W., SCHNEIDER, M., CUBEK, R., AND TOKIC, M., The Teaching-Box: A universal robot learning framework, In: Proceedings of the 14th International Conference on Advanced Robotics (ICAR'09), 2009, pp. 1–6, <http://www.teachingbox.org>.
- [8] TOKIC, M., ERTEL, W., AND FESSLER, J., The crawler, a class room demonstrator for reinforcement learning, In: H. C. Lane and H. W. Guesgen, (Eds), Proceedings of the 22nd International Florida Artificial Intelligence Research Society Conference (FLAIRS'09), Sanibel Island, Florida, USA: AAAI Press, 2009, pp. 160–165.
- [9] WATKINS, C., Learning from delayed rewards, Ph.D. dissertation, University of Cambridge, England, 1989.
- [10] THRUN, S. B., Efficient exploration in reinforcement learning, Carnegie Mellon University, Tech. Rep., 1992.

Michel Tokic, M.Sc. (Corresponding author)

University of Ulm
Faculty of Engineering and Computer Science
Institute of Neural Information Processing
James-Frank-Ring
89069 Ulm, Germany
Phone: +49-751-501-9755
Fax: +49-751-501-59755
E-mail: michel.tokic@uni-ulm.de

Arne Usadel, M.Sc. Dipl.-Ing. Joachim Fessler Prof. Dr. Wolfgang Ertel

University of Applied Sciences Ravensburg-Weingarten
Faculty of Electrical Engineering and Computer Science
Doggenriedstrasse 38
88250 Weingarten, Germany
Phone: +49-751-501-9746
E-Mail: <firstname>.<lastname>@hs-weingarten.de

Some didactic aspects of teaching robotics

Anton Vitko, Ladislav Jurišica, Andrej Babinec, František Duchoň, Marian Klúčik

Abstract

The contemporary robotics is an excellent tool for teaching science and engineering and an attractive topic for students of all ages. Problems of robotics are fundamentally about the couple “sense – act”, the two parallel activities bounded by the robot’s dynamics. It is just the robot’s dynamics that makes the relation “sense – act” difficult to control and calls for advanced approaches. The first part surveys the authors’ experience as teachers and researchers in the field of robotics at the electrical and mechanical engineering faculties. The second part points out some issues of robot control and navigation as we teach them at the university level.

Keywords: didactic aspects; mobile robotics; stationary robotics

Introduction

There is much more to robotics education than just teaching about robots. Robots are finding their way into the classroom so as to help teaching science, math, mechanics, teamwork and even management skills. Many enterprises rely on *off-the-job training* (formal learning) without considering its suitability for the learning tasks at hand. *On-the-job training* (informal learning) has a substantial advantage: it is more close to the problems to be solved. On the other hand *on-the-job training* is often unplanned and therefore mostly ineffective. For this reason, bridging formal and informal learning, theory and practice, the abstract and concrete in robotics is the best way to convince the students at all grade levels that the robotic subjects are interesting and useful. The educators have found that teaching with and about robots provide a new and exciting way to interest and motivate their students.

At the Institute of Control and Industrial informatics in Bratislava was established the Office of Robotics Education as a way to help educators, students and parents with interests in robotic. We hope this webpage will serve as a helpful launching point.

From the perspective of teaching robotics may be useful to look at the relation between robotics and mechatronics. Some time ago we found on the Internet a scheme of mechatronic system. It revealed that the same is also true for a robotic system. The robotic system (robot) is also a purposeful connection of mechanical and electrical systems (electromechanical system) equipped with actuators through which the system acquires moving abilities. Its motion is controlled in real time by a digital controller which acts on the electromechanical system through a set of D/A and A/D converters. Judging by the scheme its author probably supposed that the control program together with control data (e.g. desired motion trajectory) is loaded into the computer memory at the beginning of the working task. This may be the case of a grinding, milling or other numerically controlled manufacturing machines, which repeatedly do the same operations. Except for some force, torque or temperature sensors such simple “mechatronic” system does need any feedback from its (possibly changing) environment. Thus the scheme represented at most a classical “low level” controlled system without learning.

Complexity of current fixed or mobile robots goes much further. They are required to do tasks which go far beyond the capabilities of the classical industrial manipulators. Letting alone the sophisticated nonlinear robust and adaptive control, the primary requirement laid on modern “mechatronic” systems, which the contemporary robot undoubtedly belongs to, is ability to grasp a “mental image” of both its own state and the state its environment. Having this image (context) in mind the robot should improve its knowledge through learning from interactions with the environment.

From what has been said follows that the subjects of robotic cover a wide range of sophisticated problems requiring the university study. Therefore in what follows some problems of teaching the robotics at the university level will be briefly mentioned.

1. Some experience with teaching robot modeling and control

To understand moving operation of a robot, students must be familiar with the robot kinematics, in particular the homogenous transformations, Denavit –Hartenberg parameters, problems related to the direct and inverse kinematic, manipulator’s Jacobian matrix and the like. These problems are relatively easy grasped by all students regardless their previous education. Mastering the problems of robot kinematics creates a basic prerequisite for understanding issues of robot dynamic.

Contrary to the robot kinematics the robot dynamics is much more difficult to teach. Primary reason is that the students of electrical engineering are not sufficiently good in mechanics. So as to teach them the notions and mathematical means like the Euler- Lagrange equations, inertia matrix, expressions of kinetic and potential energy etc., the lecturer is forced to remind the basic principles of the mechanics. After doing this he/she can continue with explanation of the robot dynamics. The undergraduates of mechanical engineering are facing the opposite difficulties. They need some preliminary introduction to more sophisticated control issues.

After understanding the robot kinematics and dynamics the subject of robotics becomes much more interesting and attracts a great deal of the students' attention. The essential knowledge the students must comprehend consists in understanding the theoretic reasons why the robot manipulator (except for special configurations, e.g. SCARA robot) being controlled by linear PID controllers cannot reach an acceptable tracking performance. This knowledge is a stepping stone for presentation the philosophy of autonomous control, namely that of named as the computed torques methods.

Grasping the problems of multivariable control is again a rather demanding task. Here the most difficult is to explain principles of co-called inverse dynamics and subsequent synthesis of a robust and/or adaptive controller.

In relation to the design of a robust controller the special attention is given to the robot control based on the theory of variable structure systems (VSS). Though rather difficult, due to step by step explanations the students understand the basic principles of VSS control relatively easily and become fascinated with the possibilities the VSS control offers. In the end the strength of the VSS control is demonstrated by some results obtained with the VSS control of a flexible joint robot. They are acquainted with undesirable effects of the joint flexibility.

The syllabus ends by brief presentation of hybrid position-force control and control of mechanical impedance. It can be concluded that the subject provides students with a good overlook over the field of advanced control industrial robots.

2. Teaching mobile robotics – intelligent navigation and data

2.1 Intelligent navigation

The robot navigation is another aspect of teaching robotics. An autonomously operating mobile robot must respond to instantaneous incentives coming from its own "body" and surrounding environment. To this end the robot needs to handle a wide range of unexpected events, detect and distinguish between normal and faulty states, classify them and finally, if the fault cannot be compensated by a nominal control it should switch to an appropriate fault-tolerating regime. To manage these tasks, the robot functionality must be organized into an appropriate architecture, i.e. a set of organizing principles and core components that create a system basis.

The control community is familiar with the notion „intelligent control“, denoting abilities which the conventional control system cannot attain. Leaving alone various meanings of the “intelligent” system, some basic features characterizing an intelligent system will be mentioned here. To mention a few, they are decision making, adaptation to new and uncertain media, self-organizing, planning, and the more. [1, 2] Intelligent systems should not be restricted to those that are based on a particular constituents of so-called soft computing techniques (fuzzy logic, neural networks, genetic algorithms and probabilistic reasoning), as it is frequently done. Soft computing techniques should be considered as mere building blocks or even "bricks" used for building up a "large house" of an intelligent system. What makes a system intelligent is just a synergic use of the these techniques, which in time and space invoke, optimize and fuse elementary behaviors into an overall system behavior. For instance, fuzzy inference is a computing framework based on the fuzzy reasoning. But the fuzzy system is not able to learn and must be combined with neural networks which add the learning ability. To this end, the fuzzy rule-set

is commonly arranged into a special neural architecture like ANFIS or NEFCON with Takagi-Sugeno-Kang and Mamdani inference respectively. [3] Intelligence of such system springs from successive generalization of information chunks (granules), namely singular, crisp, and fuzzy granular information pieces. [4, 5] Due to the information granularization a system becomes robust with respect to imprecision, uncertainties, and partial truths. Thus, the system's intelligence comes from its architecture i.e. from an inner organization of the system elements and functionalities. To demonstrate this, the *subsumption architecture* (developed in 1986 by Brooks [6, 7]) and used in the synthesis of navigation algorithms of a mobile robot developed at the authors' workplace will be briefly described.

The subsumption architecture was inspired by the behavior of living creatures and heralded a fundamentally new approach to achieving more intelligent robots. The robot behaviour is typically broken down into a set of simpler behaviours which are loosely co-ordinated towards a final goal. Behaviours having higher priority are subsumed under those with lower priorities (running at the background), thus a layered structure is developed. Contrary to the classical hierarchical architecture, in which a particular behaviour assumes control if a given set of logical conditions is fulfilled, the behaviours which are organized into subsumption architecture can appear concurrently and asynchronously and with different intensities. For example, if a robot is navigated in an unknown environment cluttered with obstacles, it is natural to assign the highest priority to the obstacle-avoidance behaviour, and lower priorities to the behaviours which are to be initialised e.g. if the robot finds itself trapped in a deadlock. Using such priority management, the robot finding itself in a deadlock inhibits all obstacle-avoidance related behaviours. Instead, the behaviour being typical for escaping from the deadlock assumes control. In other words, the obstacle avoidance behaviour is normally “subsumed” by the deadlock-resolving behaviour. If the robot finds itself in a deadlock (e.g. in a partly closed space), the obstacle-avoidance behaviour is to some extent suppressed by the deadlock-resolving behaviour. Similarly, a striving-towards-a-goal behaviour subsumes both of them and therefore it possesses the lowest priority. An example of subsumption architecture that was used in the navigation of our experimental robot [8] is depicted in Fig 1. One reason why the highest priority is assigned to the obstacle-avoidance behaviour is that one can reasonably expect that the robot will encounter an obstacle when moving in a terrain. The deadlock-resolving behaviour (lower priority) subsumes the previous one because it is less probable that the robot will be trapped in a deadlock. These two behaviours are subsumed by the goal-striving

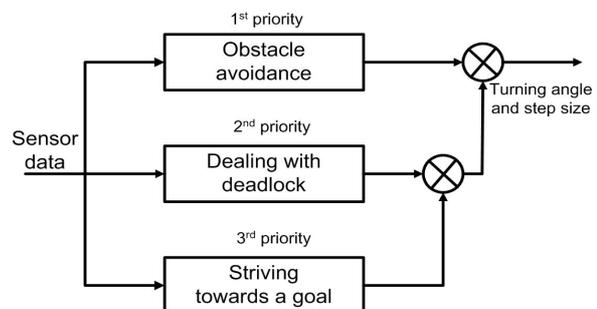


Fig.1 Subsumption architecture

behaviour (the lowest priority), because the probability that an obstacle-free landscape will appear in front of the robot is relatively low. If it happens, the goal-striving behaviour would inhibit or even suppressed both of them. The

subsumption architecture is a kind of behaviour-based architectures [9].

Great advantage of the described architecture is that if implemented through fuzzy IF-THEN rules, the transition between behaviours is very smooth. In case that a transition is controlled exclusively by current sensor information, the system is called *reactive*. The reactive systems typify a majority of the autonomous robots operating in distant and unknown environments, like sea beds, battlefields, areas hit by disasters etc. The robot navigation based on the subsumption architecture is has find great popularity among students.

2.1 Data fusion

When teaching robot navigation the issues of data fusion cannot be avoided, because an autonomously navigated robot is a particular realization of an intelligent system. In this view the teaching the data fusion naturally precedes issues of robot navigation. The thing is that the robot functionality relies on numerous disparate sensors through which it grasps a consistent image of what is going on. An underlying idea of the sensor integration rests on a synergic use of the overlapping information delivered by the sensors of different types. An aim is to obtain aggregated information that would be more complex then that of received from a single sensor. The aggregated (or blended) information is beneficial at least from aspects of noise reduction and novelty extraction, which makes the data patterns hidden in raw signals more obvious.

It is stressed that a single sensor cannot provide a required amount of information. For instance, the ultrasonic range sensor used for identification of an obstacle is uncertain about the exact location of the obstacle to which the distance is measured. This is because of the wide angle of the ultrasound wave cone. Therefore there is a need to install an additional sensor, let us a laser one, which adds additional information about the obstacle direction. Another reason that necessitates the fusion, stems from the fact that mobile robot operates in changing environment; therefore the fusion must take place not only in space but also in time. Besides, the use of a set of (distributed) sensors of different modalities allows fusion of high-level information (e.g. statements) and even to grasp a context. This is to some extent, tantamount to mimicking human-like reasoning. For instance, the fact of finding a personal mine implies higher likelihood of finding other mines or even a whole battlefield (i.e. context).

The number of sensors needed for robot navigation and fault detection is relatively large. Examples include the GPS sensors, proximity sensors, odometers, accelerometers, gyroscopes, inclinometers, velocity, temperature, light and darkness sensors and many others. In order to know "what to fuse", multimodal information must be fused into a common format, and what is very important, the uncertainty of sensed and fused signals must be taken into account.

Special attention is devoted to the hierarchical structure of the data fusion. It is explained that at the lowest level is performed the fusion of single signals or pixels. Features (mean value, variance, kurtosis, covariance, power spectrum etc.) are fused at the second level. As to the signals are of random nature, the fusion is usually based on the Bayesian statistics with Kalman filter [10, 11] as a typical representative. The aim of so called *complementary* fusion is to obtain not only accurate but also more complete information. For instance, images from two cameras looking in different directions are fused to obtain a more complex image. Another possibility is that more sensors sense the same quantity, e.g. sonar and laser range sensors. In this

case the sensors "compete" in a sense, therefore one can speak about *competitive fusion*. The third kind is *cooperative fusion*, where one sensor relies on the others, (e.g. the battery state can be observed by simultaneous measuring the electric current and time).

The situation is illustrated in Fig. 2. As seen, at low levels run cooperative and competitive fusion while at high levels runs complementary fusion. Results of high level fusion are statements (declarations) about instantaneous state of the robot, saying for instance that "in the azimuthal angle "α" at the distance "d" is seen a small pond" or "the battery is discharged to 50% of its initial capacity". In general, at the lower level runs the signal fusion and at higher level runs the symbolic fusion. While a typical means for signal fusion is Kalman filtering, a typical means used at higher levels is either Dempster –Shafer theory of evidence [12-14] or fuzzy logic [15].

The students must become aware that results of the fusion process (at all levels) are not only estimated values (numeric or symbolic) but also corresponding *certainty values*. In case of Kalman filter the result is an estimate of the mean value and by way of the certainty value is used

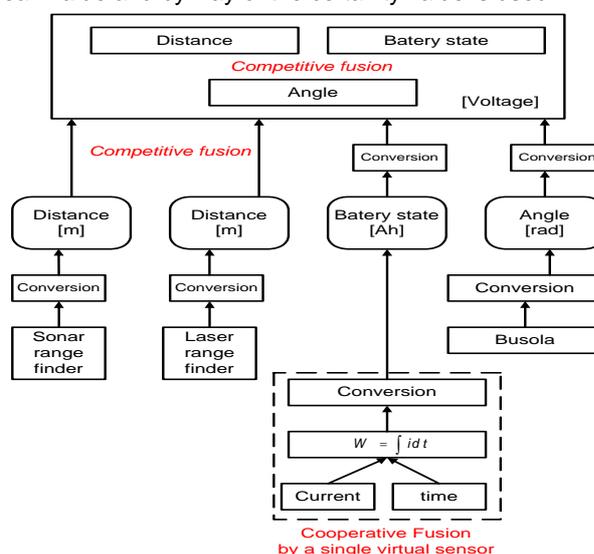


Fig.2 Types and hierarchy of data fusion

signal variance. Contrary to this, in case of Dempster-Shafer evidence theory the output is a symbolic value, supplemented by its *belief value*. Finally, in the case of fuzzy fusion, the output is the consequent of the fuzzy rule, supplemented by corresponding *degree of fulfillment* (firing strength). In the end of semester some means of data fusion are explained.

1) Example of low level fusion

Let us suppose that the random signal x with normal distribution is directly measured by two different sensors S_1 and S_2 . The estimates are x_1, x_2 , and their certainty values are given by the standard deviations σ_1 and σ_2 . An optimal estimate X is then obtained by fusing the measurements in accordance with the rule

$$X = x_2 + \left[\frac{\sigma_2^2}{\sigma_1^2 + \sigma_2^2} \right] (x_1 - x_2) \quad (1)$$

Variance σ^2 of the fused estimate X is given by the expression

$$\frac{1}{\sigma^2} = \frac{1}{\sigma_1^2} + \frac{1}{\sigma_2^2} \quad (2)$$

The fusion process can continue repeatedly in such a way that the estimate X is considered as if it would be a new

reading of the sensor S_1 , that is x_1 . The sensor S_2 performs the next measurement with the reading x_2 , which is again fused with x_1 . In this way the variance σ^2 gradually decreases while the preciseness of the estimate X is gradually improved.

2) Example of high level fusion

The high levels are occupied with more sophisticated procedures of notion identification, i.e. "what was observed" and "what it means to have observed that". The higher level is a domain for application of *possibilistic approaches*, which can directly handle symbolic quantities, e.g. propositions. Every proposition is accompanied by its certainty value (score), which expresses how certain the sensor is about its estimation of the measurand. Examples of fused propositions:

$Z_{i,e}$ = there is a cube "i" in the robot's environment "e"

$Z_{i,c}$ = object "i" belongs to cluster "c"

$Z_{d,\alpha}$ = at angle "α" there as an obstacle at the distance "d"

Higher-level fusion is based either on *Bayesian statistics* (not mentioned here) or possibilistic means, like Dempster-Shafer evidence theory and fuzzy set theory but even a short recapitulation goes beyond this paper.

Conclusions

Some didactic issues with a brief indication of syllabuses of industrial and mobile robotics taught at the university level were presented. Both the syllabuses and teaching experience as described here cannot be generalized. The teacher can appropriately modify them so as to reach the best educational results. The practical laboratory activities were not described. In general, the computer simulations of robot dynamics, control and navigation are supplemented with experimental measurements and control of both industrial manipulators and mobile robots.

The students' understanding of the robotic problems presented during the lectures and within laboratory activities can be considered as very acceptable. The graduates can easily join the robotic or other related companies.

Acknowledgements

Support of the grant VEGA No. 1/0690/09 is duly appreciated.

References

- [1] Fu K.S., Learning Control Systems and Intelligent Control Systems: An Intersection of Artificial Intelligence and Automatic Control. IEEE Trans. AC. Vol. 12, No 2, pp 17-21, 1971.

- [2] Meystel A., Messina E., The Challenge of Intelligent Systems. Proc. of IEEE Intern. Symp. On Intelligent Systems (ISIC 2000), Rio Patras, Greece, 2000.
- [3] Nauck D.; Klawonn F., Kruse R., Foundations of Neuro-Fuzzy Systems., 1th edition, Chichester, 1997.
- [4] Jang J.S.R., Sun E.C.T., Mituzami E., Neuro-fuzzy and Soft Computing. 1st edition, Prentice Hall, Inc. Upper Saddle River, 1997
- [5] Zadeh L.A., A New Direction in AI. Toward a Computational Theory of Perception. AI Magazine, pp 73-84, Spring 2001.
- [6] Brooks R.A., Achieving Artificial Intelligence Through Building Robots. AI Memo 899, MIT, Art.Intelligence Lab., 1986.
- [7] Brooks R.A., A Robot that Walks: Emergent Behaviours from a Carefully Evolved Network. Proc. IEEE Conf.on Robotics and Automation, pp 692-696, 1989.
- [8] Vitko A., Markusek J., Jurišica L., Unified Simulation Environment for Learning Navigation of a Robot Operating in Unknown Terrain. Proc.of CLAWAR, Karlsruhe, Germany, pp 435-441, 2001.
- [9] Mataric M. J., Behaviour-Based Robotics as a Tool for Synthesis of Artificial Behaviour and Analysis of Natural Behavior. Trends in Cognitive Sciences, Vol.1, No.3, pp 82-87, 1998.
- [10] Saffiotti A., The Use of Fuzzy Logic in Autonomous Robot Navigation: a catalogue raisonné Technica Report TR/IRIDIA/, Univ.Libre de Bruxelles, 97-6,1997.
- [11] Maybeck, P.S., Applied Optimal Estimation-Kalman Filter Design and Implementation. Notes of a continuing education course offered by the Air Force Institute of Technology, Wright- Patterson AFB,Ohio, since Dec.1974.
- [12] www.wins.uva.nl/~arnoud/education/OOAS/fwi/Cap9.pdf
- [13] Wu, H. et al., Sensor Fusion Using Dempster-Shafer Theory. IEEE Instrumentation and Measurement Technology Conf., Anchorage, AK, USA, pp 21-23, May 2002.
- [14] Klein.L.A., Sensor Technologies and Data Requirement for ITS. 2nd edition, Artech House, 2001.
- [15] Mendel J.M.,T.,Fuzzy Logic System for Engineering: A Tutorial. Proc. of the IEEE, Vol.83, No.3, pp 345-377, 1995.

doc. Ing. Anton Vitko, PhD.

prof. Ing. Ladislav Jurišica, PhD.

Ing. Andrej Babinec

Ing. František Duchoň, PhD.

Ing. Marian Klúčik

Faculty of Electrical Engineering and Information Technology
Institute of Control and Industrial Informatics
Ilkovičova 3
812 19 Bratislava
anton.vitko@stuba.sk, ladislav.jurisica@stuba.sk,
andrej.babinec@stuba.sk, frantisek.duchon@stuba.sk,
marian.klucik@stuba.sk

Building robots as a tool to motivate students into an engineering education

Francis Wyffels, Michiel Hermans, Benjamin Schrauwen

Abstract

Today, robots have become an integral part of our society: children have robot pets, mobile robots are mowing our lawn and robot arms are assembling cars. Since people are clearly fascinated by these mechanical slaves, we were wondering: why not use robots as a tool to teach more abstract concepts in a practical way. Recently, a new course was added to the first year of the Bachelor's in the engineering program at Ghent University. In this course, first year students have the opportunity to get more hands-on experience through several projects. In this article we focus on one of these, titled *How to build your own intelligent robot*. This work covers our approach for the practical sessions. Additionally, we elaborate on the low-cost robot platform that was built specially for this course, and which can be used easily by other schools or universities. Two years after the introduction of the robot project, we find that students not only like the sessions, but are very motivated to solve problems which would be otherwise considered too abstract and tedious.

Keywords: robotics, undergraduate education, low-cost robot platform

Introduction

Nowadays, robots are slowly finding their way from industrial settings to households, clinics and schools. Robotic pets, such as the Sony Aibo [1], are commercially available, robots such as Roomba are cleaning our houses and the first prototypes of social pet robots, such as the huggable robot Probo [2], for robot-assisted therapy are built. Similar to this evolution, robots are finding their way to the classroom [3,4,5] although often still hindered by economic constraints and some less successful stories. In [6], authors conclude that robots did not have any positive influence on student learning. However, other studies [7] show that robots can motivate students to actively do things that are not required for the course.

Recently, a new course was added to the first Bachelor's year of the engineering program at Ghent University. After an introduction of nine lectures which cover mainly written and oral presentation techniques, students have the opportunity to get more hands-on experience through several projects. Approximately 400 students have to pick their favorite subject from a list of 19 different projects such as constructing a small but precise catapult, design of a fish ladder and design of an intelligent robot. In what follows we focus on the project entitled *How to build your own intelligent robot*¹. This assignment is organized in several sessions during which students try to solve different, relatively small problems, each focusing on a particular problem in mobile robotics.

This work covers our approach for the practical sessions. In the following section we give a description of the course. Next, we elaborate on the low-cost robot platform that was

built especially for this course and can be used easily by other schools or universities. After that, we describe the content of the hands-on sessions and the feedback we got from anonymous polls taken by the students.

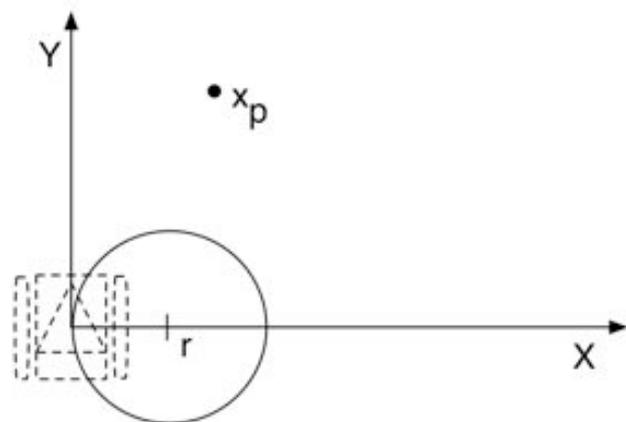


Fig.1 Graphical depiction of the main geometry problem the students have to solve during the first milestone. Throughout the course, students look for a solution such that their robot can drive as close as possible to a certain goal x_p . Students start with a two-wheeled mobile robot which is able to drive straight forward or turn with a certain radius r . Next, sensors are added and thus feedback is used to reach the goal. At the end, the morphology of the robot is changed such that the robot can cross obstacles and difficult terrain.

1. How to build your own robot

The main goal of the course is threefold. First, we show that secondary school math can be applied to a real engineering application. Next we try to give the basics of several

¹ Additional material such as pictures and videos can be found on our website:
<http://reslab.elis.ugent.be/studentcourses>

practical skills that are useful in robotics. Finally, the students have to improve their communication skills, specifically working in a team and presenting their results in oral and written form. In order to meet our main goal, we organize eight sessions which center around one practical problem: *programming a mobile robot such that it can reach a predetermined end goal in space* illustrated by Fig.1. The students try to solve this problem step by step, to break up to problem three milestones were set: *pétanque*, *golf* and *hiking*.

- In the first milestone, *pétanque*, basic concepts of mobile robot kinematics and open loop control are introduced. The students need to solve the geometry of the robot trajectory and perform measurements of the robot speed and movement.
- The second milestone, *golf*, introduces light sensors and closed loop control of the mobile robot. Here, the destination of the robot is indicated by a bright light and a white mark on the floor. Students need to program an algorithm that lets their robot drive towards the light.
- Finally, for the third milestone, *hiking*, students have to rebuild the robot in order to allow its basic morphology to cross obstacles and rough terrain.

To complete a milestone, students have to do calculations and measurements such that they can implement a solution. In order to improve their communication skills they have to defend their solution with an oral presentation and to write down a report.

The milestones are divided over eight hands-on sessions which are organized weekly. At the start of the course, students form five teams of four students which are graded both as a whole and per individual. The course counts for six credit units which indicates that an average student spends approximately 180 hours on the course, including classroom lectures. Evaluation is done throughout the semester by means of graded reports, graded oral presentations and evaluation of the given solution and collaboration. At the end of the semester, each group of students has to produce a final written report and a final presentation.

2. Low-cost robot platform

For the course, we searched a robot platform that is cheap, robust, easy to repair and flexible:

- The robot must be cheap in order to make it possible to provide enough robots such that students can work in small groups.
- The robot should be built robustly and should be easy to repair. When a large number of people are working with a device things can break or wear out easily. The robot platform should be sturdy enough to work under demanding conditions, and if something breaks, it should be repairable without too much work.
- The robot hardware should be flexible such that the platform can be adapted to different circumstances and different tasks. It should be possible to add or remove different types of sensors, without taking the robot apart.

We found this combination of properties in a platform we build of Lego™ NXT bricks and the Dwengo-board. A photograph of the robot platform can be seen in Fig.2. We designed the robot ourselves with as few pieces as possible. The Dwengo-board is a microcontroller platform with a PIC18F4550 and a wide range of onboard devices which can be used directly to build a robot without the need

for designing additional electronics. It comes with a display, motor driver, a USB- and serial port and an expansion connector where sensors can be plugged in easily². The microcontroller can be programmed in C using Microchip MPLAB IDE. The C-compiler is freely available for educational purposes.

The power supply of our robot platform is provided by six (rechargeable) AA batteries which can power the robot for the duration of at least one lesson. The total cost of the robot platform is estimated at 120 euro and is determined mainly by the microcontroller platform and the two Lego™ NXT motors.

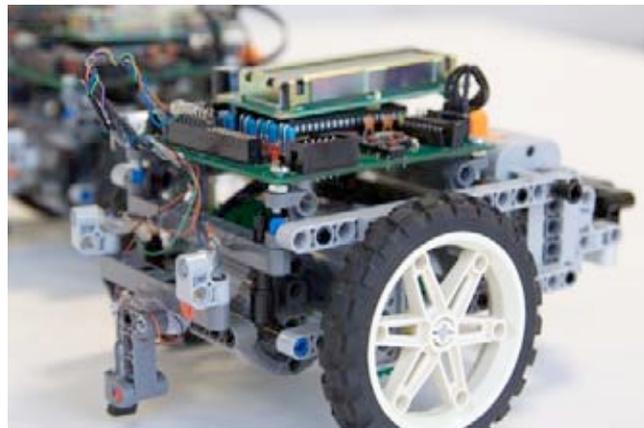


Fig.2 Robot platform used in the course. The construction is built by Lego™ NXT bricks. The core of the robot is formed by the Dwengo-board which contains a PIC microcontroller. Through the expansion connector the robot can be equipped with multiple sensors. In this photograph, two light sensors and one ground sensor are visible.

3. Hands-on sessions

The core of the course are the eight weekly held hands-on sessions. In order to meet the course goals we choose to apply a combination (not necessarily all) of following teaching methods in one session:

- Homework: searching a solution for a problem through homework by investigation of existing literature and using creativity. Often, a session ends with an open question for which they have to seek an answer at home.
- Presentations: usually, the homework included preparation of a presentation in which they formulate their ideas, solutions for the posed problems.
- Brainstorm moments: the student presentations were followed by classical brainstorm sessions during which students try to extract the best elements from each presentation in order to come to a solution.
- Theoretical introduction: during each sessions the main concepts and workflows are introduced by means of a theoretical introduction. We choose to keep these introductions as brief as possible and they never last longer than one hour in order to get maximal attention. Additionally, we interact (questioning, polls, ...) as much as possible to keep them attentive.
- Hands-on work: by applying several methods and doing measurements themselves, students get the most

² The full specifications of the microcontroller platform can be found on <http://www.dwengo.org>

experience in how to bring theory into practice. Therefore, the main bulk of the time slot of the lessons was dedicated to this.

- Competition: at the completion of each milestone, competitions are held such that students are able to compare their results with other groups. They are assured that the result of the competition doesn't directly influence their grades.

As stated before, three milestones are divided over eight hands-on sessions. In order to reach the first milestone, during three hands-on sessions students have to find a solution to program a robot so that their robot can reach a certain goal (x,y) on a flat surface. At this stage, the robot has two wheels and no sensors. Additionally, students have to assume that the robot is limited to drive straight forward over a distance D or taking a turn with certain fixed radius r over an angle θ . Therefore they have to find the angle θ and distance D in function of the goal on (x,y) . Some additional problems, such as finding the shortest possible path, have to be solved. Next, students have to measure the properties (speed, possible deviation when driving forward,...) of their robot and estimate the angle and distance so that the robot reaches a given point as close as possible. Since most of the students have no programming experience yet, programming is done through a graphical interface we provided in which they can specify how long the robot has to follow a certain path. The workflow of this milestone is comparable with the game *pétanque* for which a ball has to be thrown so that it lands as close as possible to the object ball. Such as the open loop control of the robot, during the flight, one can not interfere with the ball.

In the second milestone we introduce the concept of feedback. Two light sensors and one ground sensor³ were added to the robots, while the destination was marked by a light source and a white sign. During four sessions students have to measure the properties of the sensors, program the robot using a state chart, and finally program their robot using the programming language C (using some helpful libraries and starting from a template such that not much knowledge of C is necessary). Again, they have to find the optimal (quickest) way to get their robot to the destination. One can compare this with the game *golf* for which it is possible to correct (by multiple strokes), give feedback, in order to get the ball into the hole.

Finally, the third milestone is devoted to finding a solution to drive a robot, cf. *hiking*, over difficult terrain. Until now, they refined the intelligence and the senses of their robot. However, this doesn't enable it to drive over obstacles or irregular surfaces. Therefore each team has to design a new robot that is inherently able to do so by how it is constructed. A good example of this, using a limited amount of pieces, is depicted in Fig.3. The idea is to introduce the concept of morphological computation [8], i.e. using morphology rather than a "brain" (microprocessor), to solve locomotion problems.

For every milestone at least one presentation and one report has to be completed. After the first presentation and report a lecture is held during which distinctly good and bad things are pointed out. Additionally, for every report we gave some remarks to each group individually.

³ The ground sensor is distance sensor, but is used to measure the reflectance of the underlying surface. This allows the robot to detect it has reached its end goal, which is marked with a white spot



Fig.3 A minimalistic robot design which illustrates the concept of morphological intelligence: without being programmed to do so, the robot is able to go across obstacles.

4. Feedback from students

Since the first introduction of the course two years ago, we have had one official evaluation (organized by the faculty) and two informal evaluations (organized by the specific lecturers of *How to build your own intelligent robot*). The overall conclusion is that students highly appreciate our project and our enthusiastic approach. In the official evaluation, students gave the project a score of 87%. Additionally, 90% of the students agreed with the proposition that the course increased their interest for engineering while the other 10% had no strong feelings about this question and thus didn't agree nor disagree.

On top of the official evaluation, we wrote an informal questionnaire, which probes for a more detailed opinion of the course. Again, we learned that students were charmed by the content and our approach. In order to find out whether students found the course useful, we posed the following three propositions with which they could respectively strongly agree, agree, stand neutral, disagree or strongly disagree:

1. I have the feeling I had to use my creativity during the course
2. I have the feeling I learned from the hands-on sessions
3. I have the feeling I learned from the given theory

On all three questions, students replied positive, as can be concluded from the graph in Fig.4. We believe that a combination of factors form the basis of our success. By implementing a robot platform in our course, students can immediately see the consequences of their thoughts and actions. For the same reason, hands-on sessions where practical engineering skills can be fully expressed form the core of the project. Apart from this, we believe that motivation and enthusiasm of the lecturers also play a role in the positive reception of the course. This enthusiasm is transferred to the students and motivates them to solve the posed problems.

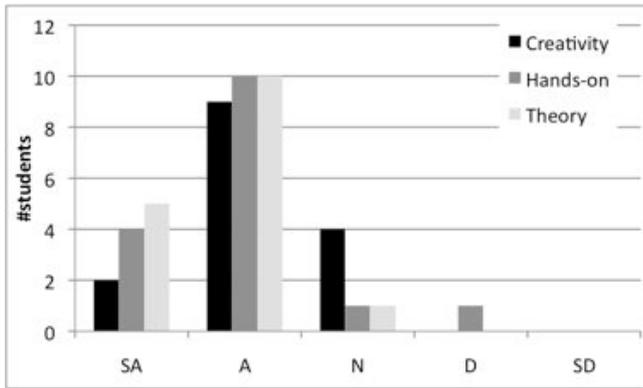


Fig.4 Student feedback concerning three questions: how much creativity they felt they used, whether they learned from the hands-on experience, and whether they learned from the theory or not. Students can strongly agree (SA), agree (A), being neutral (N), disagree (D) or strongly disagree (SD).

We also wanted to know whether the students found the course helpful in order to improve their communication skills. The following three propositions were posed:

1. I learned how to work efficiently in a team
2. I learned how to write a good report
3. I learned how to give a good presentation

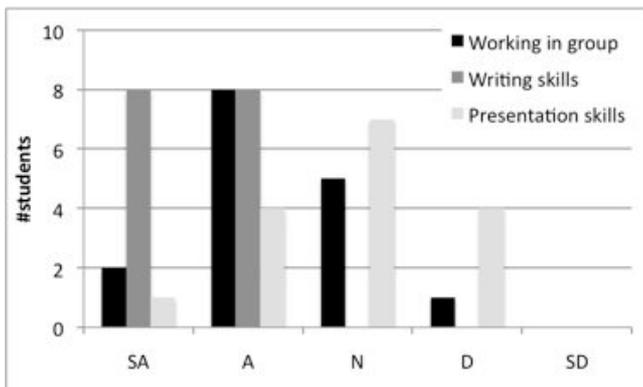


Fig.5 Student feedback concerning the success of the course in increasing their teamwork skills, writing skills and presentation skills. Students can strongly agree (SA), agree (A), being neutral (N), disagree (D) or strongly disagree (SD).

From the results presented in Fig.5 we learn that students are able to work in group, even if the members are not acquainted with each other from start, and that we succeeded in teaching them how to write a good report. However, some students believe that their presentation skills did not increase by our course. We believe we can overcome this problem in the future by giving more detailed and individual feedback of their presentation.

Conclusions

In this work, we gave an overview of the course *how to build your own robot* which is held in the first year of the Bachelor's in the engineering program at Ghent University. In this course, students use high school mathematics to solve problems in the domain of mobile robotics. Additionally, student communication skills are increased by

working in a team, writing reports and giving presentations to their peers. From student polls, we learned that students not only gain useful new skills but are also motivated to solve problems in the domain of engineering which could be otherwise found abstract and tedious.

Acknowledgements

The authors would like to thank Professor Jan Van Campenhout for the fruitful discussions related to this course.

References

- [1] FUJITA, M.: Aibo: Toward the era of digital creatures, The international Journal of Robotics Research, vol. 20, pp. 781-794, 2001
- [2] SALDIEN, J., GORIS, K., YILMAZYILDIZ, S., VERHELST, W., LEFEBER, D.: On the design of the huggable robot probot, Journal of Physical Agents, vol. 2, pp. 3-12, 2008
- [3] COOPER, M., KEATING, D., HARWIN, W., DAUTENHAHN, K.: Robots in the classroom - tools for accessible education, Proceedings of the 5th European Conference for the Advancement of Assistive Technology, 1999
- [4] GOLDWEBER, M., CONGDON, C., FAGIN, B., HWANG, D., KLASSNER, F.: The use of robots in the undergraduate curriculum: experience reports, Proceedings of the 32th Technical Symposium on Computer Science Education, 2001
- [5] CHALLINGER, J.: Efficient use of robots in the undergraduate curriculum, Proceedings of the 36th Technical Symposium on Computer Science Education, 2005
- [6] FAGIN, B. S., MERKLE, L.: Quantitative analysis of the effects of robots on introductory computer science education, Journal on Educational Resources in Computing, vol. 2, 2002
- [7] KAY, J. S.: Robots in the classroom... and the dorm room, Journal of Computing Sciences in Colleges, vol. 25, pp. 128-133, 2010
- [8] PFEIFER, R., BONGARD, J.: How the body shapes the way we think, a new view of intelligence, MIT press, 2006

ir. Francis Wyffels

ir. Michiel Hermans

Prof. dr. ir. Benjamin Schrauwen

Ghent University
 Faculty of Engineering
 Electronics and Information Systems Department
 Sint-Pietersnieuwstraat 41
 B-9000 Ghent, Belgium
 Phone: +32 9 264 95 26
 Fax: +32 9 264 35 94
 E-mail: Francis.wyffels@elis.ugent.be,
 Michiel.Hermans@elis.ugent.be,
 Benjamin.Schrauwen@elis.ugent.be

A programming platform for Arduino on Physical Etoys

Gonzalo Zabala, Ricardo Moran, Sebastián Blanco

Abstract:

The objective of this paper is to introduce a new module of Physical Etoys which aims to persuade kids to do different electronic projects with an Arduino Board in an enjoyable and powerful way.

Keywords: educational robotics, Etoys, technology education, Arduino

Introduction

In the last fifteen years, technology education has been essentially based on digital technology, leaving aside the use of real material. Although we have excellent simulators of the physical world, working with real material allows the development of cognitive structures that the digital world does not offer. Moreover, these didactic resources encourage highly participative group dynamics that have not been yet reached by the existing computers at schools.

Unfortunately, in our view, there are two major difficulties for the presence of these resources in the classroom. The physical technology is expensive and suffers from constant wear. On the other hand, teachers are not accustomed to work in a dynamic classroom with a methodology of participative groups, and have fears about the use of specific technological equipment.

Physical Etoys is a development that aims to overcome these difficulties. It facilitates the interaction between inexperienced users and real material such as open hardware devices or popular toys by providing a powerful and intuitive visual programming system in order to explore and learn science in an enjoyable way.

1. Reasons for the development of the project

In the following sections we explain the reasons of why we decided to develop this type of project.

1.1 Fluency in the use of technology

First of all, during the last fifty years, technology has taken an important place in our lives at the point that we cannot conceive a life without the integral use of it. It is for this reason that different analysts of the current school, as David Perkins [8] among others, consider that the presence of technology in classrooms is essential and a change of perspective that includes the student in its environment and educational process is needed. That is to say, the student is no longer only the student: it is him plus his technological resources. It no longer matters where the knowledge is but how you access it. The problem is that, in spite of the exponential decrease of the costs of these resources, we are still in front of a considerable digital divide among those included and those excluded from the system. This gap is given by the significant use of technology rather than the

access to it. The more disadvantaged social classes are away from the metaphors that current technologies propose. It is for that reason that the use of physical resources allows to leave this framework and opens conceptual and learning new opportunities. In summary, the children of all social classes grow up playing with real material, and these games involve learning deep technological concepts. If we keep this profile in the formal learning of technology, we will be able to reach a bigger number of students.

1.2 Technology with real material

Besides from a social greater reach, the physical material allows us to develop not only as excuse intellectual activities but also sensory, which diminishes the problems of the passage from the concrete thought to the abstract one. In the physical experimentation, the student takes the error as a factor of his learning. Moreover, it allows him to operate and control a group of continuous variables that no computer simulator provides. It is the real world the one that defines the results reached by the boy's experiences.

However, the solving problems abilities that this material requires help the students develop a systemic, structured, and logical thought, starting from physical problems instead of premises or abstract situations.

Linda Williams [10] proposes to realize activities with real material that generates processes not only in the left hemisphere of the brain (highly developed by the daily activities of the school) but also in the right hemisphere, which will allow to integrate components with a simultaneous and parallel process in order to exercise spatial and visual intelligences.

1.3 Cross-curricular thematic without gender differences

Technology is present in all the activities of our lives. It does not belong to a particular science, discipline or workspace. Therefore, it is fundamental that our students integrate the use of technology in all their subjects, and not simply in those where its presence seems "more natural". This implies that we should leave the traditional framework of technology education in where we develop devices with an end in itself, like the robot that follows lines. We should carry out significant projects for each child such as modeling devices that men use in their daily life. These projects will then serve as excuse for analyzing and pursuing diverse topics of the curricula. It is habitual that the technological activities of this type attract more boys than girls, for cultural diverse reasons that escape to this article. If we are able to propose the design of daily-life devices (for example, a table to

create ceramic vessels, a microwave, a washing-machine, the dancer of a music's box, a turnstile) we will encourage the cultural diversity that we have inside our classrooms.

1.4 Motivation for learning

Diverse studies demonstrate the motivational impact that generates the use of these materials in the students, which are not accustomed to highly participative activities. The possibility to build significant devices of real utility and the instant feedback that offers the test and error, creates in the student a deep interest not only in the construction but also in the contents linked to the carried out activity. That is, the use of these materials, even in less technological subjects, makes the curricular content more interesting to the student.

1.5 Teamwork

Working with these physical tools cannot be done properly without cooperation. Teamwork is necessary, beyond the economic limits in the purchase of equipment. It is important to organize this teamwork with differentiated roles so that each participant has a specific work in the activity. Each of these roles enables the student to develop a skill set. Therefore, these activities make us possible to introduce learning about teamwork and roles, conflict resolution, respect for differences and the need to listen to all members of the team. Each participant has its own standpoint that enriches the work of the team. The proposed roles are related to the organization of working materials, the construction process, the communication with the teacher and the others teams, the development of written reports, and other activities.

1.6 Using low cost hardware

The main cost of these projects is not the software, but the hardware platform used. For this reason, it has been decided to make a programming platform that supports open or low-cost hardware and popular robotics kits which are being used in many schools across the world. Therefore, schools without much funds can still purchase low-cost materials (such as Arduino boards) and schools that already have some robotic kits can improve their use by programming them in Physical Etoys.

2. Technological characteristics of this project

2.1 Cross-platform

One of the goals defined at the start of this project was the possibility to work on both Linux and Windows. In addition, the works developed in it should be cross-platform too in order to improve sharing between students. During the development, Sugar support was requested by end users. Sugar is the operating system of the XO, the computers of the One Laptop Per Child project. Nowadays it runs on 90% in the three systems.

2.2 Extensible

The experiences lived in the education technology community suggested that the development should not only be open but also easily extensible. The hardware proposals for the teaching of technology emerge every day so it is desirable a flexible tool that can be adapted by any developer to the technology of his preference. This is the reason of why an easily extensible framework was developed.

2.3 Why we use Etoys?

Etoys, the new educational version of Squeak, is an education tool to create multimedia and interactive projects. It has a long tradition of open development, because it was made by the Smalltalk team: Alan Kay, Dan Ingalls and other researchers. Furthermore, their educational criteria have been defined by great educational thinkers, such as Jerome Bruner and Seymour Papert [7]. Etoys is a highly effective tool for teaching math, science and arts, in a context of play and experimentation. Moreover, it is cross-platform and has become the most important software on the OLPC laptops because it came integrated with Sugar from the outset. A large academic community is behind its development. Examples include MIT, Viewpoints Research Institute and University of Illinois.

For these reasons, Etoys was chosen to be the base platform for the development of Physical Etoys.

3. Physical Etoys: Overview

Physical Etoys (<http://tecnodacta.com.ar/gira>) is a visual programming tool based on Etoys that connects the virtual world of computers with the real world in which we live in. With Physical Etoys it is possible to program real world objects (such as robots) to perform interesting tasks, or sense the world and use that information to control virtual objects (such as drawings on the screen).

It does not require any programming skills, and its consistency across the entire system makes it easy to accomplish some reasonably complex tasks that would be almost impossible in a different one.

Etoys is a wonderful software that helps children explore their own creativity in fun and educational ways, but it lacks communication with the outside world. Physical Etoys extends Etoys in order to overcome this necessity by providing an interface to work with real objects keeping the same educational philosophy as Etoys.

In outline, Physical Etoys is divided into a set of independent modules. Each module is responsible for controlling one robotic kit. Even though these modules can work independently from each other, the connection between them produces the most interesting results.

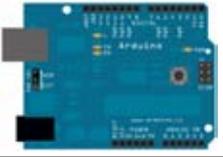
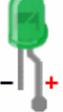
4. What is Arduino?

Arduino is an open hardware platform based on a simple microcontroller board with digital and analog I/O pins. Due to its open philosophy, every teacher can access to different designs and build his own board (it is also possible to buy a prebuilt board). In addition, there is a great variety of examples of Arduino and a very collaborative community that is fond of helping people.

These characteristics are suitable for people who want to start using physical technology. Although the Arduino's official software is intuitive, it is still a low-level language like C and it looks cryptic for the average user.

5. Using Arduino with Physical Etoys

All Physical Etoys modules are composed by a few objects that try to resemble the real objects of their respective kit. The Arduino module is not an exception. You can see in the table below some of the Physical Etoys' objects and their correspondence in reality.

Name	Virtual object	Real object
Arduino board		
Buzzer		
Led/Pwm Led		
Photoresistor		
Potentiometer		
Pushbutton		
Servo		
Switch		
Thermistor		
Tilt switch		

All the pictures of electronic devices were taken from Fritzing, an open source software that allows users to document and share their hardware prototypes. Fritzing is widely used in the Arduino community to document examples and tutorials.

The “Arduino board” is the main object of the Arduino kit. It contains pins on which other electronic devices can be attached using wires. All these interactions between real objects have been represented in Physical Etoys as you can see in the picture below.

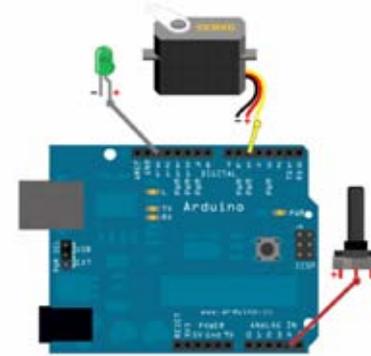


Fig.1 Components attached on a virtual Arduino

Every object has its own set of properties and commands that are accessible using the same interface. For instance, the “Led” object has an “is on” boolean property, the “Servo” has a “degrees” property and the “Photoresistor” has a “light value” property.

This interface is also shared with all the graphical objects in Physical Etoys. Texts, sliders, pictures, buttons and every user interface widget that composes Physical Etoys is accessible and programmable in the exact same way (although they contain a different set of properties and commands). This extreme consistency across the entire system makes it really easy to use and explore.

6. Educational examples

This section will describe a few exercises that can be implemented in a classroom.

6.1 Building a greenhouse

It is possible to build a miniature model of a greenhouse by using a servomotor and a thermistor. The motor will be used as a fan that keeps the greenhouse cool and the thermistor will sense the temperature of the air and it will activate the motor when its value exceeds a certain number.

The picture below shows a simple implementation of the greenhouse project. The script “controlTemperature” at the top of the picture is the responsible of the behaviour described above.

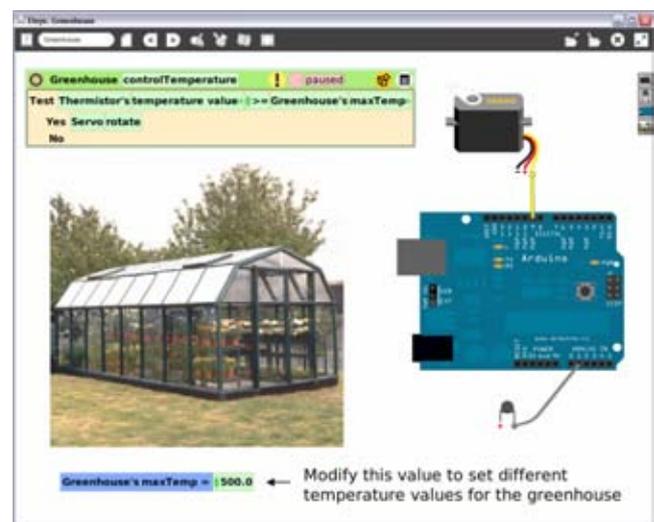


Fig.2 Greenhouse project implementation

6.2 Building a traffic light

This exercise is a little more complicated. It uses three leds of different colors to represent a traffic light. Each led is turned on/off depending on the color behind a little "Stick" that moves across three different backgrounds: red, yellow and green.

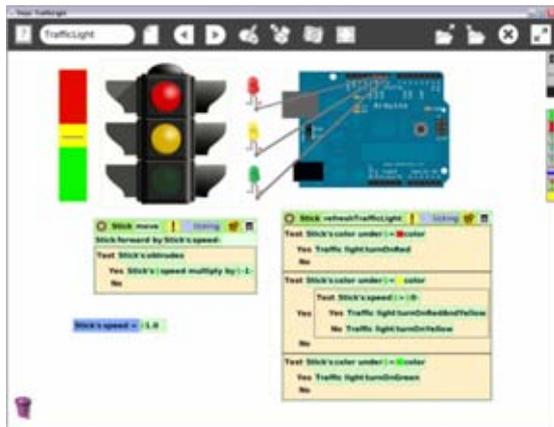


Fig.3 Traffic light project implementation

These examples show two essential aspects of Physical Etoys programming. On the one hand, it shows how abstract information such as the traffic light state and its behaviour become palpable. On the other hand, it shows how the information of the world, such as the temperature of the air, can be conceptualized as numbers which can be used in any arithmetic or logical operation.

7. Other hardware platforms supported by Physical Etoys

The other modules composing Physical Etoys are listed below:

- Nintendo Wiimote: The famous Nintendo Wii's Joystick which detects the gesture of a hand, enabling the user to make scripts with a non-conventional way of communication with the computer.
- Parallel port: A type of interface for connecting various peripherals to the computer.
- Lego Mindstorms Nxt: A programmable robotics kit released by Lego. It allows the user to build almost anything without any knowledge of electronics. Considering that a lot of schools around the world already utilize the Lego Nxt to teach robotics, using Physical Etoys to program it is ideal for children that are just starting on the subject.
- RoboSapien V2, Roboquad and I-Sobot: These robots can be controlled by using an infrared transmitter. They are prefabricated and although their capabilities are limited, they are very attractive to the general public.

8. Physical Etoys in the world

Different educative communities have shown interest in using Physical Etoys on their own classes and workshops after the publication of its modules:

The SqueakNxt module, responsible of controlling Lego Mindstorms Nxt robots, has been used by an educative organization called Planète Science which took place in a workshop of Introduction to Robotics given at the Japan Expo Paris in France 2009. This non-profit organization intends to spread the science on the youth by organizing multiple activities including workshops at festivals and

national contests such as the Final Eurobot, the French Robotics Cup and the First Lego League of France among others. During the Japan Expo Paris 2009 they used the SqueakNxt module to do different projects including:

- A drawing robot (similar to Logo).
- A robot that reacts to the environmental noise (its arms moved when somebody shouted).
- A robot that navigates through the exposition avoiding people.
- A robot that navigates through the exposition in order to lift plastic glasses using its clamps.

Planète Sciences has also shown interest in the Arduino Project, which has also been included in a software pack called SqueakBot, similar to Physical Etoys. Educational robotics has become mandatory in the French official curricula so there was a special class oriented to teachers in the region of Toulouse about the basic concepts of electronics and programming with Physical Etoys.

In Colombia a company called HYPER Neurotek which develops and integrates new technologies with education (preferably open-source projects) has shown interest in using Arduino to teach children how to use microcontrollers for building robots with an OLPC laptop.

In Spain, Citilab, an institute for the formation and the spreading of the ICT in Barcelona, decided to use SqueakNxt and Arduino for its Introduction to Robotics talks.

Finally, in Brazil, a consulting company called O3 Tecnologia that works in the area of educational technology used the Parallel Port project with Physical Etoys in robotic classes in high school.

Conclusions and future work

The recognition that Physical Etoys has received in this short time has filled us with pride. This invites us to new challenges. The first one is to fully support the use of all hardware platforms on the three operating systems. Some users have also requested to add Mac Os to them. The next challenge is to include the microphone and camera of the netbooks as sensors to our project. In the case of the camera, we must think how to provide students with an easy programming mode, removing the complexity that the image processing has. And finally, we will propose a simple physical structure with motors and sensors, allowing to locate the netbook on it for using as an autonomous robot. Physical Etoys has a long way to go. We invite you to do this together.

References

- [1] Ahlgren, D. J, Verner I.: "An international view of robotics as an educational medium", International Conference on Engineering Education (ICEE'2002).
- [2] Alimisis D., Moro M., Arlegui J., Pina A., Frangou S., Papanikolaou K.: "Robotics & Constructivism in Education: the TERECoP project", EuroLogo, 40:19-24, 2007.
- [3] Bers M., Ponte I., Juelich K., Viera A., Schenker J.: "Teachers as designers: Integrating robotics in early childhood education", Information Technology in childhood education 123: 145, 2002.
- [4] Druin A.: "Robots for kids : exploring new technologies for learning", San Francisco: Morgan Kaufmann, 2000.
- [5] Kafai Y. B, Resnick M.: "Constructionism in practice: Designing, thinking, and learning in a digital world", Lawrence Erlbaum, 1996.

- [6] Odorico A. H.: "La robótica desde una perspectiva pedagógica", Revista de Informática Educativa y Medios Audiovisuales 2, no. 5: 33–48, 2005.
- [7] Papert S.: "Mindstorms: children, computers, and powerful ideas", 2^o ed, New York: Basic Books, 1993.
- [8] Perkins D.: "La escuela inteligente. Del adiestramiento de la memoria a la educación de la mente", Gedisa Editorial S A, 2003.
- [9] Sánchez E.: "La robótica pedagógica". Educación, universidad y sociedad: el vínculo crítico: 117, 2004.
- [10] Williams L.: "Aprender con todo el cerebro", Barcelona: Martínez Roca, 1986.
- [11] Zabala G.: "Desarrollo en un entorno educativo de objetos para el control de una interfaz de domótica", Anales de WICC, 2007.
- Lic. Gonzalo Zabala
Ricardo Morán
Sebastián Blanco
Centro de Altos Estudios en Tecnología Informática
Universidad Abierta Interamericana
Buenos Aires, Argentina
Av. Montes de Oca 745
(C1270AAH) Ciudad Autónoma de Buenos Aires
(+54 11) 4301-5323
gonzalo.zabala@vaneduc.edu.ar
richi.moran@gmail.com
sebastiangabrielblanco@gmail.com

- 2001** AT&P journal PLUS 1: Adaptívne a nelineárne riadenie systémov (tlačená verzia)
Adaptive and nonlinear control systems (printed version)
- AT&P journal PLUS 2: Robotika, mechatronika, diskrétné výrobné systémy (tlačená verzia)
Robotics, mechatronics, discrete manufacturing systems (printed version)
- 2002** AT&P journal PLUS 3: Robustné systémy riadenia (tlačená verzia)
Robust control systems (printed version)
- 2003** AT&P journal PLUS 4: Samonastavujúce sa systémy v riadení procesov (tlačená verzia)
Selftuning systems in process control (printed version)
- 2004** AT&P journal PLUS 5: Robotické systémy (elektronická – CD verzia)
Robotics systems (electronic – CD version)
- 2005** AT&P journal PLUS 6: Mechatronika (elektronická – CD verzia)
Mechatronics (electronic – CD version)
- AT&P journal PLUS 7: Umelá inteligencia v praxi (elektronická – CD verzia)
Artificial intelligence in Practise (electronic – CD version)
- 2006** AT&P journal PLUS 1: Mechatronické systémy (elektronická – CD verzia)
Mechatronic systems (electronic – CD version)
- AT&P journal PLUS 2: Inteligentné meracie systémy (elektronická – CD verzia)
Intelligent measurement systems (electronic – CD version)
- 2007** AT&P journal PLUS 1: MMaMS'2007 (elektronická – CD verzia)
MMaMS'2007 (electronic – CD version)
- AT&P journal PLUS 2: Riadenie procesov (elektronická – CD verzia)
Process Control (electronic – CD version)
- 2008** AT&P journal PLUS 1: Mobilné robotické systémy (elektronická – CD verzia)
Mobile robotic systems (electronic – CD version)
- AT&P journal PLUS 2: Riadenie v energetike (elektronická – CD verzia)
Control of Power Systems (electronic – CD version)
- 2009** AT&P journal PLUS 1: Inteligentné pohybové systémy (elektronická – CD verzia)
Intelligent motion control systems (electronic – CD version)
- AT&P journal PLUS 2: Riadenie procesov (elektronická – CD verzia)
Process control (electronic – CD version)
- 2010** AT&P journal PLUS 1: Systémy automatického riadenia (elektronická – CD verzia)
Systems of automatic control (electronic – CD version)