Preprint of a publication in
Robotics in Education (RiE), Advances in Intelligent Systems and Computing.
The final publication is available at Springer via http://dx.doi.org/10.1007/978-3-319-62875-2_18.

TUC-Bot: A Microcontroller based Robot for Education

Sven Lange, Peter Weissig, Andreas Uhlig, and Peter Protzel

Chemnitz University of Technology, Dept. of Electrical Engineering
and Information Technology, D-09107 Chemnitz, Germany,
firstname.lastname@etit.tu-chemnitz.de

Abstract. We describe the concept of a mobile robotics course for undergraduate students from an educational point of view in terms of learning goals, experiences, and hardware design. The course as well as the hardware was continuously improved over more than a decade. Hence, we like to describe our motivation and the current structure of the course in order to share our experiences as an inspiration for similar courses.

1 Introduction

Autonomous mobile robots are becoming increasingly important in our modern world with autonomous cars being the most prominent but not the only example. Especially, industrial and service robotics are a growing market as the capabilities of the robots increase. Hence, a society without robotics will be unimaginable in the near future.

Thus, robotics should be mandatory in engineering education, particularly for students of electrical engineering, information technology, computer science or similar fields. This has motivated us to continuously improve a course at our university focusing on autonomous mobile robotics which initially started in 2004. Originally, the course was inspired by a previous idea to interest high school students in the field of electrical engineering by organizing a robotics competition called RoboKing — see (Sünderhauf et al., 2006).

For the course at our university, we adopted the competitive character of the event to motivate the students. They have to team up in groups of two or three and solve the given task of programming a mobile robot to navigate through an unknown maze. For this, they have an overall time span of two semesters with an expected work load of 240 hours. Most of the time, they have to work independently by self-organizing their team effort to come up with a final solution. Additionally, we guide them by theory lectures and pre-defined milestones, which are described in Sect. 4.1.

In the beginning of the course, suitable commercially available robots for the given task and with an educational focus did not exist, so we decided to design a custom-made mobile robot. Over time, it evolved and we currently use the third generation as described in Sect. 2.

Now that the community has grown over the last years and also because of the big interest of hobbyists, there are many commercially available products,

which may be suitable replacements for our custom-made robot. Examples are the E-Puck, Khepera, Thymio, 3pi, or TurtleBot to name a few. However, a problem with using popular commercial products is that solutions for nearly every imaginable task are publicly available and our students might be tempted to use existing solutions to solve the course problems. Hence, we decided to use our own robotics framework and enforce the pedagogical goal of understanding a previously unknown system. A more detailed description of our learning objectives follows in Sect. 4.

The idea to motivate and interest students in robotics by using competitions has been very successful, proven by the existence of various national and international robot competitions which found their way also into education. The authors of (Swenson, 2015) for example, are doing a robotics course which follows a slightly different but still comparable approach. They describe similar experiences and name problem-solving as the students' main take-away skill, which is exactly the same impression we have. An interesting modification of the typical one-final-task-per-course style is given by (Cappelleri and Vitoroulis, 2013). Their course follows a similar motivation as ours, but breaks down the idea of one final project task into several smaller ones to keep the motivation level up. We follow a similar approach but instead of varying the competition tasks completely, we use interim competitions as milestones toward the final goal which is described in Sect. 4.1. To increase the motivation, we reward the best teams from the interim competitions by giving extra points to be used as an advantage in the final competition.

Another concept, motivated by the thought, that many students feel the urge to experiment with a robot at home, is presented in (Aroca et al., 2013). They achieve a low-cost solution by using a common cellphone as the main processing unit. As an alternative to a conventional course, e-learning in combination with remote access to a real robotic systems is especially interesting for a very large number of students — e.g. in (Kulich et al., 2013), the authors present such a concept.

1.1 The Task

The general idea of the course is a robot competition with two opposing robots autonomously navigating through a previously unknown maze. Besides the exploration and navigation, there is a special challenge in locating eight so-called beacons and switching them to a team-specific state. The beacons have three states: neutral (initial state), red and green. The state toggles between red and green by pushing a button at the beacon. The first transition from neutral produces randomly either red or green. The button can be pushed by a robot by slightly colliding with it. The robots can sense the state via an IR transmitting diode, the humans see a red or green LED at the top of the beacon. The two opposing robots belong to the red or green team, respectively. Now the task of the green robot is to explore the maze, find the beacons and switch them to green. Simultaneously, the red robot tries to find and switch all beacons to red. Thus, each robot needs to revisit all beacons as often as possible and switch

Fig. 1. Two of the robots in the maze. The one in front is equipped with LEDs for external pose estimation (tracking) and in the background, aside from the second robot, a beacon can be seen.

them back to their state if necessary. This task can be greatly facilitated, if the robot is able to build a map of the maze with all the beacons and can localize itself reliably within that map. Originally, we used this idea in a national robot competition for high school students called RoboKing, which is described in (Sünderhauf et al., 2005, Sect. III–IV).

As playing field, a flat blue surface with dimensions of $2.4 \times 2.4\,\mathrm{m}$ is used. It is divided by black lines into 64 quadratic fields of 30 cm length each. The robots can use these lines for orientation and localization. The maze itself is constructed and surrounded by white walls of 15 cm height which are always aligned with the black lines. Therefore, the quadratic fields are only separated, but never subdivided. The walls can be arranged in a very flexible way which enables us to create many different mazes. Since both opposing robots should have the same conditions, they start in opposite fields of a point-symmetrical maze. Figure 2 gives an overview of a possible maze.

A single game lasts up to 7 minutes. Afterwards the teams get points for each beacon switched to their respective color. If a team had interfered with its robot during the match, maybe because it got stuck, the referees subtract penalty points for each interference. Likewise, penalties up to disqualification may be imposed by the referees, if a robot collides on purpose with its opponent.

1.2 Target Audience

The target audience for our course are third year undergraduate students. They have previous knowledge of basic math, physics, computer science, and microprocessor technology. The course in mobile robotics is the first larger project in their studies, which needs team-oriented self-organization and a large amount of applied thinking. As the courses in basic control theory and sensor technologies

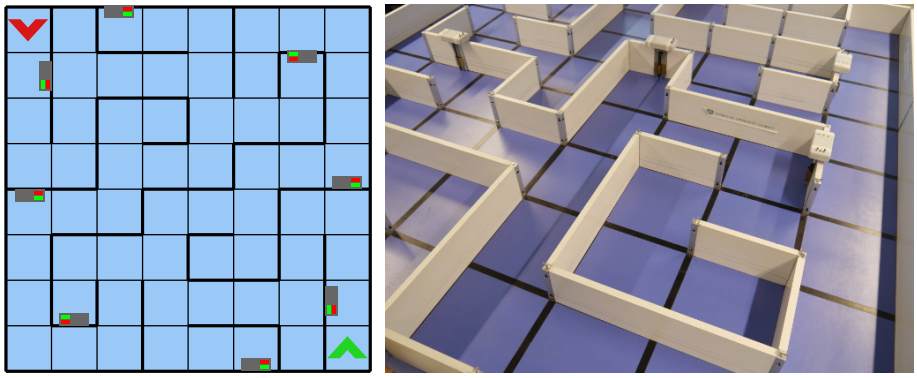

Fig. 2. Two views of different point-symmetrical mazes. *Left*: Schematic top view. The arrows represent the starting positions and directions of the opposing teams. The thick black lines symbolize the fixed walls and the black boxes mark the positions of the beacons. *Right*: Oblique view of our real maze. Easy to recognize are the blue surface, the black lines and the white walls. More difficult to spot are four white beacons in the upper part of the picture.

start at the same time as our course, we can not expect the needed prior knowledge in these subjects. Hence, we give introductory lectures to enable a smooth start into the interdisciplinary field of mobile robotics.

Besides the undergraduate students in electrical engineering, we also have high school students performing an internship as our second target audience. As they have much less time and less prior knowledge, we designed our system to cope with it and included a simpler API with more basic functions as well as additional sensors to solve a much simpler line-following task.

2 Hardware

The current version of our robot is called TUC-Bot with a chassis made of aluminum to withstand the students' harsh testing conditions. It is a differential drive robot, hence the sharp turns and dead ends within the maze can naturally be dealt with. Following this kinematics concept, we use two metal gearmotors with 6 V and a maximum current of 2.4 A each. Both motors are equipped with Hall effect sensors reading 48 counts per revolution (CPR) of the motor shaft. Consequently, by using wheels with a diameter of 60 mm, we have a resolution of about 9 counts/mm movement of the robot. Further, the configuration results in a maximum velocity of about $0.5\,\mathrm{m\,s^{-1}}$.

Besides the wheel encoders, the robot is equipped with various sensors that are especially useful for the given task as described below. Afterwards, a discussion of the user interface, the modular structure, and the maintenance requirements follow.

Distance Sensors For obstacle detection — specifically, the walls and other robots — we equipped the system with three IR-distance sensors (see Fig. 3,

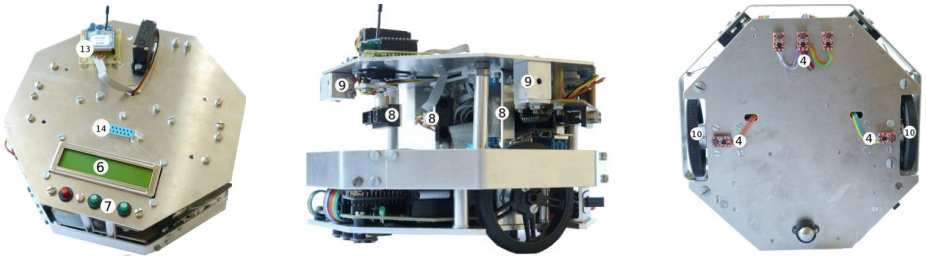

Fig. 3. Our TUC-Bot viewed from different perspectives (left: top-view; middle: left-view; right: bottom-view). 4: line sensors; 6: display; 7: buttons; 8: IR-distance sensors; 9: IR-receivers; 10: wheels; 13: radio transceiver; 14: internal connector for expansions

no. 8 and Tab. 1 for the location and exact type). They give a point measurement based on triangulation and return a range-correlated voltage with high resolution starting from about 10 cm and decreasing resolution up to about 80 cm. For ranges shorter than 10 cm, the measurement is ambiguous, hence the two sensors pointing sideways are within the robot's diameter in such a way that their lines-of-sight are crossing each other. In other words the sensor on the left side is sensing the right wall and vice versa. Additionally, both sensors are tilted slightly forward, which will improve the robot's capability to follow and adjust its position to the walls. As a special feature the front sensor is mounted on a servo motor and can therefore be turned into different directions.

Line Sensors As already described, the playing field is structured by black lines, which divide it into a grid of squares. If recognized by the robot, the borders may be used for rough position as well as orientation estimation. For this purpose, two line sensors are mounted beside each wheel (see Fig. 3, no. 4). These sensors measure the reflective properties of the ground — consequently, the black lines can be detected. Note, that three additional sensors are mounted in the front to enable the robot to do other tasks like simple line following. This is especially considered for high school students serving an internship.

Bumpers In addition to the IR-distance sensors for collision avoidance, our robot is equipped with a surrounding bumper bar for collision detection. It is slidably mounted and centered with springs. Depending on the point of collision, one or two of the four evenly distributed switches is triggered.

IR-Receivers and Beacons In the task description (Sect. 1.1) we mentioned the beacons. They are attached to the walls and emit a modulated 38 kHz infrared signal. If nearby, the robots are sensing the current state (neutral, red, green) of the beacons through a modulated infrared signal with different on-off ratios with the help of directed IR-receivers. The directionality of the receivers is realized by putting them into a pin-hole case. Three of these sensors are mounted around the robot, see Fig. 3 no. 9.

User Interface For programming and debugging of the robot, the various ways of user interaction are very important, especially for inexperienced users. This is why the robots are equipped with a display (no. 6), LED illuminated buttons (no. 7) and an IEEE 802.15.4 radio module (no. 13). While the display is intended for showing fast and simple status information, the radio module enables advanced logging and data analysis.

Modularity The third generation of our robots is intended to be used not solely for the course at hand. Hence, we divided the robot's structure into modules, where the base module includes motors, encoders, power supply, etc. as shown in Fig. 4. It has its own preprogrammed microcontroller and provides an I^2C interface for basic driving functions. This module can be extended by the additional *sensor module* or used in a direct way e.g. in combination with an embedded PC like the *RaspberryPi*. In this version of the course, we use the sensor module as the only extension. It provides additional sensors and a microcontroller to be programmed by the students.

Maintenance requirements Currently we have five robots available for the students and 7 PC workplaces, which need constant maintenance effort. At the beginning of the course, creating new user accounts and other administrative tasks is mostly automated and executed by one of two technicians. This is currently the most time consuming part. Consequently, some time is needed by a second technician to do minor repairs like tightening a loose screw or replacing a broken cable. The otherwise robust hardware is the result of the lessons learned from the previous generations of the robot, which will be described based on examples within the next section. The workload of both technicians together should not be more than 40 hours within a year. Of course, this depends a bit on the number of participants which has been fluctuating between 15 and 30 students over the years.

2.1 History

The TUC-Bot presented here is the result of a continuous process, which lasted over 10 years. Compared with its predecessor, see (Sünderhauf et al., 2006, Fig. 9), we made several improvements. As the modifications may be of general interest for similar projects, we discuss the most relevant ones following the principle: Old component, problem, solution.

Servo motors with optical encoders: Especially the encoders were prone to errors due to scratches. Additionally, the wheels' axis needed a ball bearing as a second support, which sometimes led to unwanted mechanical tensions. Our *solution* was exchanging those components with robust metal gear motors with internal Hall effect encoders.

Bumper bar: the previous construction of the bumper bar was split into a front and a rear bar. Both were directly coupled to their microswitches without further mechanical support. As a result the switches were worn out rapidly and

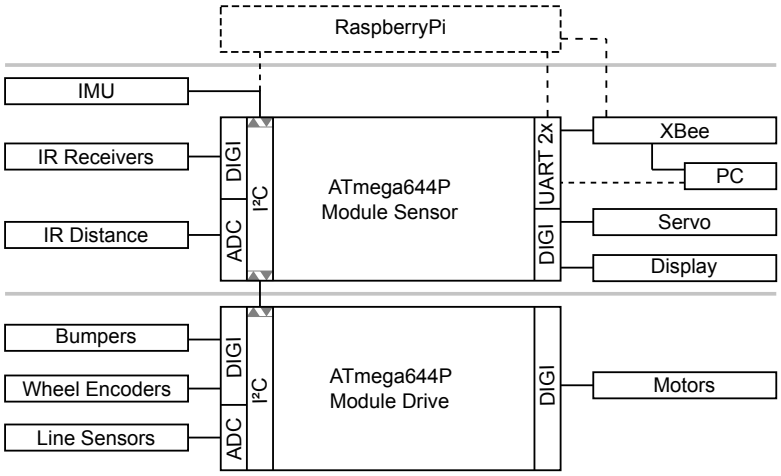

Fig. 4. The robot's architecture showing the two microcontrollers connected to the various sensor and actor components. The lower part illustrates the components of the preprogrammed base, extended by the second microcontroller with its components in the middle part. Finally, an embedded PC can be placed on top to extend the programmable module or even replace it by using the I^2C communication.

it was possible to rip the bar off during crashes. Now the bar is a closed ring-like construction hold by the chassis.

IR-distance sensor arrangement: Previously we used only two sensors mounted on servo motors, in the front instead of three sensors. This directly led to the problem that at least one sensors must be turned around all the time to detect walls either in front or side. This resulted in more complex and prolonged measurements.

Communication: The previous ways of communication were limited to a wired serial interface. Especially online debugging and logging of sensor information in this way was not reasonable. Now, with the radio module, advanced debugging features are possible.

Component	**Description**	**Costs**
2x Microcontroller	Chip45 Crumb644 (ATmega644P)	50 €
3x IR-Distance Sensor	Sharp GP2-1080K (10 cm to 80 cm)	20 €
3x IR-Receiver	VISHAY TSOP 31238	5 €
5x Line Sensor	Pololu QTR-1A Reflectance Sensor	15 €
1x Radio Transceiver	XBee (2.4 GHz; Series 1; PCB-Antenna)	30 €
1x Display	LCD (2x16 symbols; backlight)	15 €
2x Motor with Encoder	Pololu 34:1 Metal Gearmotor with 48 CPR Encoder	75 €
1x Motor Driver	Pololu Dual MC33926 Motor Driver	30 €
1x Battery Pack	6 Cells; NiMH; 7.2 V; 1900 mA h	30 €
Miscellaneous	PCBs, Mechanical Components, other	130 €

Table 1. The robot's main components result in an overall cost of about 400 €.

3 Software-Library

Since we want the students to program an autonomous mobile robot to perform some demanding high-level tasks, it would be counterproductive to let the students program the robot from scratch at the lowest level, especially since we can not expect previous knowledge in basic microcontroller programming, as stated in Sect. 1.2. Therefore, we developed a software library for our robot, which implements basic functionalities. They are provided as a documented API and include simple functions for getting sensor information or passing commands to the actuators without the need to know on which microcontroller pin they are connected to or how to use e.g. the ADC. Nevertheless, we provide this information and every student has the freedom to implement everything from scratch.

Providing the right amount of such predefined library functions is also controversial. Thus, we first defined the learning objectives and then determined the necessary functionalities to achieve our objectives. For example, implementing a speed control algorithm is part of our learning objective, so we do not have any predefined functions for controlling the speed of the robot's wheels within our library — only a function for setting the voltage.

In addition to the library and its documentation, we provide sources for relevant datasheets, literature, and other information. Furthermore, every computer workplace is preconfigured with the needed software libraries and an editor including the makefile directive to enable a seamless start for the students in an Arduino-like style.

4 Learning Objectives

In general, there are multiple learning objectives present throughout the two semesters of the course, which will be described in the following.

Omnipresent challenges in programming — not only mobile robots, but technical systems in general — are the error susceptibility, imprecision, and individual variations within hardware components like sensors and actuators. One could say that this is one of the main insights the students should take away from this practical course: Algorithms which work in theory or simulation will not necessarily work with the real hardware. Of course, this is not the only thing to learn. The following general learning objectives describe some of the desired competencies in terms of *The students will be able to*:

1. explain the components' working principles of the previously unknown microcontroller-based mobile robot.
2. analyze a complex task and split it into several sub-tasks like processing sensor data, implementing controllers, mapping, etc. while self-organizing their activities in the time available.
3. implement, modify, analyze, and create microcontroller-based algorithms in C/C++ in consideration of the limited resources of an 8-bit microcontroller.

4. record and analyze sensor data for the purpose of debugging and identification of programming errors or a deeper understanding of the sensors' characteristics and working principles.
5. explain reasons for measurement outliers and select suitable methods to cope with them.
6. explain the concept of a differential drive robot.
7. demonstrate a working system which is solving the given task of navigating through the maze and present their specific implementation details by giving a short oral presentation.
8. use tools like subversion to develop and manage source code in a small team.

4.1 Structure of the Course

We support the above stated learning objectives by additional lectures, guided practical work, software, and other material. As the time for complex topics like digital control theory is not nearly sufficient, most of our lectures are of a more practical type with theory explained on the example of the given mobile robot. A more complete theory is then given in subsequent courses, depending on the student's choice of specialization. In the following, we like to give an overview of the structural design of our course, which is the result of continuous improvement over the last years.

Workflow and Subversion We start the course with team building and proceed with a first introduction into the workflow. This includes an overview of the robot's components and their basic working principle. Additionally, all the tools and given materials are presented and programming the robot is demonstrated, so the students could start right away with programming. Surprisingly, most of the students have no prior experience with any version control system. As this is important for team work, we also give an introduction into version control, especially subversion.

Control Theory An important step in navigating the robot through the maze is the ability to drive straight. Many students without experience in mobile robots do not regard this as a problem, but this is the first issue where theory (all motors of the same type are identical) and practice (manufacturing tolerances) diverge. Providing the same voltage for each motor will lead to a curved trajectory of the robot because the individual variations within the motors result in different, non-linear characteristic curves. In consequence, wheel encoders have to be used to control the velocity of each wheel to a specific velocity. The needed control theory for implementing a cascaded velocity controller is given within one lecture. Here, cascaded means two individual velocity controllers for each wheel and an additional controller on top to control the error between the two wheel velocities.

Visualization and Debugging In a scientific approach, it is quite natural to analyze a system by recording and plotting the available sensor data. For undergraduate students, this does not seem so natural and has to be learned first.

We made the experience, that if the students do not know the exact way to record and plot the data, most of them will not bother and try to circumvent it. Hence, we saw the necessity to do an example-based step-by-step introduction regarding the communication flow and the tools to use for data visualization. We present two toolchains: first an open source solution using a simple python script for serial communication in combination with gnuplot for visualization, and second, a single tool solution by using Matlab for both — communication and visualization. We conclude our lecture with a simple practical task, where the students have to apply the just learned theory on the robot. Simultaneously, we get feedback by assisting the groups. In this way, we often see weaknesses, which optionally could be addressed in further lectures.

Guided Practical Work Parallel to our lectures, the students have to do guided practical work in the first two month of the course, before they can start with their own ideas to solve the competition's task. The guided work is composed of three individual guided task descriptions for getting to know the characteristics of the motors as well as the distance sensors and implementing a controller for the wheels. We introduced this step to familiarize the students in a guided — and therefore faster — way to the robot's properties. Of course, in a project like course, this should not be necessary at all, but our experiences showed that especially groups with low previous knowledge are appreciating this and as a result get more motivated.

Interim Evaluation Unfortunately, students tend to underestimate the work to do for a given task. With regard to this course, it leads to bad results in the final competition and weak learning outcomes because all the work is done within the last two weeks before the final competition. To counteract this behavior, we introduced sub-tasks as milestones with an interim evaluation. They produce no additional workload, because the implementations necessary for solving them are a required stepping stone for the final goal.

In the first evaluation, the robot has to move along an imaginative square of about 2 m in dimension. Mainly odometry and some logic for moving straight has to be implemented. Next, the robots have to navigate through a simplified maze in the fastest way they can. This mainly requires using the distance sensors as well as the line sensors to implement a navigation strategy. The third evaluation consists of the maze including the beacons but without an opponent. Winner is the the team who activated the most beacons within the shortest time.

As well as the final demonstration, these sub-tasks are carried out in a competition like manner. By giving points to the best three groups which can be used as advantage in the final competition, we motivate good performance.

Oral Presentations Even though the course is based on a competition, the students are required to present their findings and experiences after each evaluation task by giving a short presentation. This fosters the mutual information exchange and is an opportunity to improve their presentation skills.

5 Continuous Improvement

As mentioned before, we guide our students in their understanding of the basic hardware, especially the sensors. Based on this knowledge, they develop the control algorithms and the overall decision logic and robot control architecture. This architecture is usually either reactive or plan based. The reactive architecture is very simple as it only needs to consider the current state of the robot. Therefore, it is easy to implement and can be tested quickly. The plan based architecture is more sophisticated and complex as it additionally incorporates previous knowledge. For example, the robot creates its own map of the maze with the beacons while exploring. Afterwards, it uses this information for navigation and path planning to drive to the next beacon(s) or for self-localization. As much as we would like the students to implement more sophisticated plan based logic, most of them stop at simple reactive behaviors, because it is already sufficient for the final competition. Admittedly, these simple solutions are often more robust and actually increase the performance in the competition. Nevertheless, there are always some very motivated students who want to explore the limits of their control architecture and try out the advanced methods which we also introduce. Since we do not enforce a particular method, we always see many different approaches in the final competition, often with surprising results.

We are planning various extensions and updates of our current robot hardware and software structure. In the following, we discuss three such ideas.

Simulation and Reality Typically, the students differ in their degree of motivation as well as learning speed. Consequently, some students are overstressed and some are subchallenged. This is partly compensated by the project nature of the task and leads to very different solutions ranging from implementing only rudimentary functionality to very complex algorithms including mapping and path planning. Primarily to support the motivated and subchallenged students, we replicated the maze and the robot within the V-REP simulation. In combination with a Matlab wrapper with the same naming conventions as in our library, it is possible to test algorithms for the robot. This has the charm of circumventing possible hardware flaws and develop algorithms in a rapid-prototyping style. Additionally, we can exchange V-REP with the real robot to have a hardware-in-the-loop structure, where the robot accepts commands from Matlab and returns its sensor information.

Extended Sensor Information Depending on the quality of the sensor information available, programming the robot is more or less challenging. Hence, if the students have to spend much of their time debugging unreliable sensor information, there will be less time for higher functions like mapping and path planning. On the other hand, if the sensor readings are very accurate and reliable, the learning focus will entirely move away from processing sensor data and relating low-level tasks which are still part of the learning objective. Hence, a balance between both extremes has to be found.

Currently, we have two options for extending sensor information. We recently added an IMU, basically for the gyroscope values around the z-axis. This is a good way of giving the students reliable information regarding the robot's orientation, which is otherwise hard to get — odometry and line sensors are possible ways, but get unreliable after collisions with walls within the maze. Even if the orientation measurements are calculated by integration of the turn rate, tests showed only little drift of about 5° after 5 min movement within the maze. This could be fixed by using the maze's grid lines and the robot's line sensors. In contrast, fusing global information from a magnetic field sensor is unreliable because of the maze's metal foundation.

Besides using an IMU, an external measurement system would be another option for position and orientation estimation. Therefore, we added a camera with fisheye lens above the maze and mounted LEDs with bright illumination on top of the robots. By using the same principle as in (Lange and Protzel, 2012), we can determine the robots' poses and orientations with sufficient accuracy. This works well in controlled lighting situations as the system is sensitive to external illumination sources. On the other hand, this system with its global position and orientation information might simplify the task too much which would impair our learning objectives. For this reason, we only used it e.g. for system identification purposes but did not make it available to the students as an additional measurement system.

Higher Algorithms Due to the positive feedback from the students and the good learning outcomes, we plan to offer an advanced course with the same project-like character, but with the focus on advanced algorithms. In consequence, a simple microcontroller as the only processing source is insufficient. So we already experimented with an additional Raspberry Pi embedded PC as visualized in Fig. 4. There are three possible ways for connecting the PC: by using the TWI, or serial interface, where we can address the serial interface either directly on the robot or via the XBee module. As a proof-of-concept, a student developed a simple example of following a red ball with the robot using a camera. For educational use, this was done in three versions with ROS as a base. The actual example solution was realized in Matlab with help of the Robotics System Toolbox, in Python, and in C++ using the OpenCV library.

6 Conclusion

We described in detail our motivation and structure in teaching the basics of mobile robotics with a project-based approach. We gave an overview of our hardware and software concept in conjunction with the intended learning goals of an introductory third year undergraduate course in mobile robotics. By following an competition-like approach, we motivate the students to choose additional lectures in this field.

Even though the overview of the course's structure may be obvious to experienced tutors, it should be useful for those creating a new practical course on

mobile robots. The course benefits electrical engineering students by challenging them to overcome their software engineering deficiencies as well as computer science students who have never seen a PID control algorithm. In addition to those hard skills, the course is also a good opportunity to sharpen the soft skills in organizing their team work, giving presentations, and learning about new methods like subversion or other suitable agile methods as described e.g. in (Gerndt et al., 2014).

References

Aroca, R. V., R. B. Gomes, D. M. Tavares, A. A. S. Souza, A. M. F. Burlamaqui, G. A. P. Caurin, and L. M. G. Goncalves (2013). "Increasing Students' Interest With Low-Cost CellBots". In: *IEEE Transactions on Education* 56.1, pp. 3–8. DOI: 10.1109/TE.2012.2214782.

Cappelleri, D. J. and N. Vitoroulis (2013). "The Robotic Decathlon: Project-Based Learning Labs and Curriculum Design for an Introductory Robotics Course". In: *IEEE Transactions on Education* 56.1, pp. 73–81. DOI: 10.1109/TE.2012.2215329.

Gerndt, R., I. Schiering, and J. Lüssem (2014). "Elements of scrum in a students robotics project : a case study". In: *Journal of Automation Mobile Robotics and Intelligent Systems* 8.1, pp. 37–45.

Kulich, Miroslav, Jan Chudoba, Karel Košnar, Tomáš Krajník, Jan Faigl, and Libor Přeučil (2013). "SyRoTek—Distance Teaching of Mobile Robotics". In: *IEEE Transactions on Education* 56.1, pp. 18–23. DOI: 10.1109/TE.2012.2224867.

Lange, Sven and Peter Protzel (2012). "Cost-Efficient Mono-Camera Tracking System for a Multirotor UAV Aimed for Hardware-in-the-Loop Experiments". In: *Proc. of Intl. Multi-Conference on Systems, Signals and Devices (SSD)*. DOI: 10.1109/SSD.2012.6198047.

Sünderhauf, Niko, T. Krause, and P. Protzel (2006). "Bringing Robotics Closer to Students – A Threefold Approach". In: *Proc. of Intl. Conf. on Robotics and Automation (ICRA)*.

Sünderhauf, Niko, Thomas Krause, and Peter Protzel (2005). "RoboKing - Bringing Robotics Closer to Pupils". In: *Proc. of Intl. Conf. on Robotics and Automation (ICRA)*. DOI: 10.1109/ROBOT.2005.1570774.

Swenson, Jessica (2015). "Examining the Experiences of Upper Level College Students in 'Introduction to Robotics'". In: *Proc. of Intl. Conf. on Robotics in Education (RiE)*.