High-Level Control System for a Path-Planning Teaching Aid

Pavel Škrabánek and Filip Majerík

University of Pardubice, Faculty of Electrical Engineering and Informatics,
Studentská 95, 532 10 Pardubice, Czech Republic
pavel.skrabanek@upce.cz

Abstract. The paper brings a description of a high-level control system which is a part of a teaching aid aimed at practicing path-planning methods. The teaching aid uses a proven concept of a mobile robot operating within a maze. The high-level control system ensures path-planning, data collection, data processing and data distribution. This contribution covers topics related to the development of a software part of the high-level control system. Specifically, software requirements, software design, and software testing are detailed in the text.

Keywords: software requirements, software design, software testing, path-planning, autonomous mobile robot, teaching aid

1 Introduction

Incorporation of practical work in the education process is highly desirable. As Phil Race pointed out: 'Students can gain a lot of feedback while they do practical work. They get very rapid feedback just by seeing how the work itself proceeds, and often get even more feedback by watching and talking with fellow students working alongside them.' [11].

Many areas of study involve practical work. Let us mention physics, chemistry or biology as the typical examples; however, practical work is also often incorporated in curricula of many less traditional subjects. It is not surprising that practical work is used in artificial intelligence (AI) courses.

Within AI courses, robotics is particularly preferred when developing practical work. As Lloyd et al. specified: 'robotics is a remarkable domain that may be successfully employed in the classroom both to motivate students to tackle hard AI topics and to provide students experience applying AI representations and algorithms to real-world problems' [5].

Numerous teaching aids based on robotics were developed for AI courses. These teaching aids may significantly vary. While Sebastian van Deleden used an industrial robotics laboratory to practice some fundamental AI topics [2], Dogmus et al. [3], Mariška et al. [10] or Marcelo Fernandes [4] developed software based solutions aimed at practicing AI techniques related to planning. Both these approaches are limited. While the industrial robotics laboratories are expensive

and might be oversized, students lose the connection with reality when working in virtual environments of the software based solutions.

A good practice is using of small robots within AI classes because they do not have the above stated insufficiencies. Very popular are commercial robot kits, such as products of companies LEGO [9, 5, 1] or K-Team Corporation [6, 12]. Nonetheless, the commercial products are not always suitable. They might be expensive and they may not be perfectly fitting to a specific teaching purpose. In such cases, development of a specific teaching aid is highly desirable.

In order to improve AI courses lectured at the University of Pardubice, Czech Republic, we developed a teaching aid aimed at practicing search strategies. The lack of suitable inexpensive solutions was our main motivation for the development. We used a proven concept of a mobile robot operating in a maze. A good practical experience with the created solution, motivated us to share its integral description for the purpose of its wider utilization. Due to its high complexity, we published its description in several thematic contributions. In this paper, we would like to detail its most complex part - the high-level control system. Specifically, we focused on an analysis of the software requirements, the software design, and the software testing.

2 Background of the Practical Work

In this section, we describe an objective of the intended practical work (subsection 2.1), and we provide basic information about the teaching aid (subsection 2.2).

2.1 Objective of the Practical Work

The goal of the intended practical work is to support an understanding of a transition system, as well as practicing search strategies lectured within a course 'Introduction to Artificial Intelligence 1'. All the treated search strategies work with a discrete model of a workspace. Within the practical work, the students utilize the curriculum when developing a *path-planning routine* according to a given assignment. Specifically, searching for a single pair shortest path, using various search strategies, is the task to be practiced [13, 17]. The students verify their knowledge when applying the routine by searching for the shortest path between an initial (current) position of the robot and a target position in the maze. The robot is placed in the maze; hence, both positions are in the maze. The robot executes a path-plan, scheduled by the routine.

2.2 Teaching Aid

The teaching aid consists of the maze, a camera system, the high-level control system, and a mobile robot. All these parts are further described in necessary detail.

Maze The maze is the workspace of the robot. We developed a maze building kit which allows us to create workspaces of various layouts [15]. We designed the kit as a modular system which consists of partitions of a fixed size, floor blocks, and posts. The partitions and the posts are obstacles from the perspective of the robot [18].

When arranging a maze layout, two basic requirements must be respected: the outer shape of the maze must be rectangular, and the maze must be a closed system (isolated from a surrounding world) [15]. It means that only workspaces of a rectangular layout can be assembled. It allows a simple conversion of the workspace into the discrete model using an exact cell decomposition [14]. In our solution, borders of cells are determined by feasible positions of the partitions [15, 18], as is shown in an example (Fig. 1).

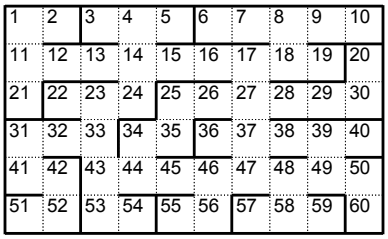

Fig. 1. Example of the maze layout where partitions are sold lines, borders of cells are dashed lines, and an index of a cell is in its top left corner.

Mobile Robot The role of the mobile robot is to substantiate path-plans. We used a differential wheeled mobile robot which was developed at the University of Pardubice, Faculty of Electrical Engineering and Informatics, Department of Process Control. The robot can autonomously execute a path-plan provided by a path-planning routine; however, the plan must be converted into a *sequence of actions* [18]. The path-plan in this form must be provided by the high-level control system. The communication with the high-level control system is supplied by a Bluetooth module [18, 8].

Camera System The camera system provides information about the real-world. It consists of a stand arm and a camera. In the recent version of the camera system, an IP camera D-LINK DCS-935L is implemented. The camera is fixed on the stand arm and the whole camera system is placed such that the camera is above the maze [15]. It means that a top view of the maze is streamed by the camera to the high-level control system. For communication, WiFi technology is used [7].

High-level Control System The high-level control system is the 'brain' of the teaching aid. It consists of a hardware platform and software. We used a lenovo YOGA Tablet 2-851F as the hardware platform. The software, which is the subject-matter in this paper, must ensure data collection, data processing, and data distribution. Although path-planning routines developed by students are used for the path-planning, the high-level control system must also serve all path-planning related tasks.

3 Design of the High-Level Control System

In this section, we focus on the analysis of the software requirements (subsection 3.1), and on the software design. In the context of the software design, we provide a description of a graphical user interface (GUI) design (subsection 3.2), and a description of the workflow of activities in the presented solution (subsection 3.3).

3.1 Analysis of the Problem

We distinguish between two roles. While the students are considered to be regular *users*, the teacher is a *supervisor*. The role of the user should allow the students to connect their path-planning routines, select the target position, select the path-planning routine, and initialize a path-plan execution. The supervisor must have the same authorization as the user; however, the supervisor is allowed to change the maze layout and carry the robot to any position in the maze.

The high-level control system must provide to the user and to the supervisor at least all the above stated functionalities. Nevertheless, it must also solve other tasks. Desired functionalities of the software can be divided into three groups. We distinguish among functionalities related to the path-planning, functionalities related to the image processing, and functionalities ensuring a communication. Herein, we summarize all technical specifications which are relevant for the development of software.

Path-Planning Related Tasks Basically, searching for path-plans is supposed to be carried out by path-planning routines developed by students. Although the routines may employ various search strategies, they have identical inputs and outputs. To solve a path-planning problem, the initial position, the target position and the discrete model of the workspace must be provided to the routines. We used an adjacency matrix for the model representation [15, 17, 18]. Thus, the initial and the target position must be specified in accordance with indexing used in the matrix. Consequently, the routines return path-plans as sequences of vertices the robot should pass through. However, path-plans in such a form cannot be executed by the robot. Thus, a conversion of the plans into sequences of actions must be carried out by the high-level control system [18].

Image Processing Related Tasks Images provided by the camera contain information about the maze layout, as well as about a current *state of the robot*. We developed a procedure aimed at an extraction of the information about the maze layout [15]. The output of the procedure is the adjacency matrix. We also proposed a solution aimed at a localization of the robot in the maze [16]. It returns information about a current position of the robot in the maze, as well as information about a robot's orientation. These two parameters determine the state of the robot. Since the information about the position is in pixels, a mapping of the position to the discrete model must be carried out by the high-level control system.

Communication Related Tasks In the context of the analysis, the term communication is perceived from a broader perspective. It covers the interaction of a user with the high-level control system; a communication of the system with the robot; a communication of the system with the camera; and a communication of the system with the path-planning routine. In our approach, the path-planning routine is a standalone application, which is run from the hardware platform, e.g. in a student's laptop. The communication between the high-level control system and the routine is realized via an IP socket where the connection is established by the routine. For the communication with the camera, we used the WiFi standard IEEE 802.11g. For communication with the robot, Bluetooth technology is used [8]. The communication with the user is realized using the GUI.

3.2 Design of the GUI

Once the teaching aid is switched on and the robot is placed in the maze, any interaction of the user with the teaching aid might be realized entirely via the high-level control system. Thus, the software must provide the information about the maze layout and the robot's state. Further, the software must allow the determining of the target position, the selection of a desired path-planning routine, and the activation of the robot's movement. Since the supervisor may modify the maze layout during exercises, the analysis of the maze layout, only by software initialization, is insufficient. On the other hand, a periodical execution of the analysis would unnecessarily burden the high-level control system. Considering these facts, an execution of the analysis on a user request seems to be the best approach.

The above stated use cases, determine functionalities that the GUI should provide. A layout of the GUI, a graphic design, as well as a user control were optimized for the used hardware platform, which is designed to be controlled via a touchscreen. In order to ensure a simple and intuitive control of the software, only four control buttons were used in the GUI. They are placed in one column on the right side of the GUI, while a current real-world situation is shown on the left side (Fig. 2).

The maze layout is depicted using black bold lines. The robot is symbolized by a car. The state of the robot is expressed by a position of the car and its

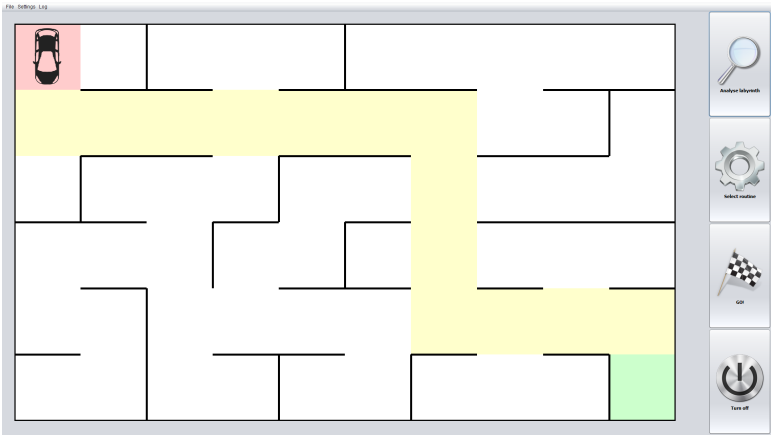

Fig. 2. Graphical user interface of the high-level control system.

orientation. The target position can be determined by touching a finger anywhere in the scheme of the maze layout. A selected cell is considered to be the target position. Once the target position is determined, a selected path-planning routine is requested for a path-plan. A received path-plan is then displayed in the scheme. Three colours are used to emphasize the start position (red), the target position (green), and the shortest path between them (yellow).

The rest of the functionalities are provided via the control buttons. Pressing the button 'Analyse labyrinth' invokes the analysis of the maze layout. Pressing the button 'Select routine' opens a dialog window for the selection of the path-planning routine. Pressing the button 'GO!' initializes a physical execution of the path-plan by the robot. Finally, pressing the button 'Turn off' causes a shutdown of the software.

3.3 Workflow of Activities

The basic requirements on the functionalities of the software (subsection 3.1), extended by the requirements on the GUI (subsection 3.2), determined a final structure of the software. We used an UML (unified modeling language) activity diagram to describe the structure of the presented solution (Fig. 3). The diagram shows the workflow of stepwise activities in the context of the whole teaching aid.

Logically, the first action is switching on the teaching aid. This activity covers turning on the camera, the robot, the tablet, and the software. The second step is establishing connections of the high-level control system (software) with the camera and the robot. In this moment, a status of the high-level control system is set to 'on'. The next action is the analysis of the maze layout, followed by a creation of the adjacency matrix, and display of the maze layout on the screen. After that, the communication with the robot should be served.

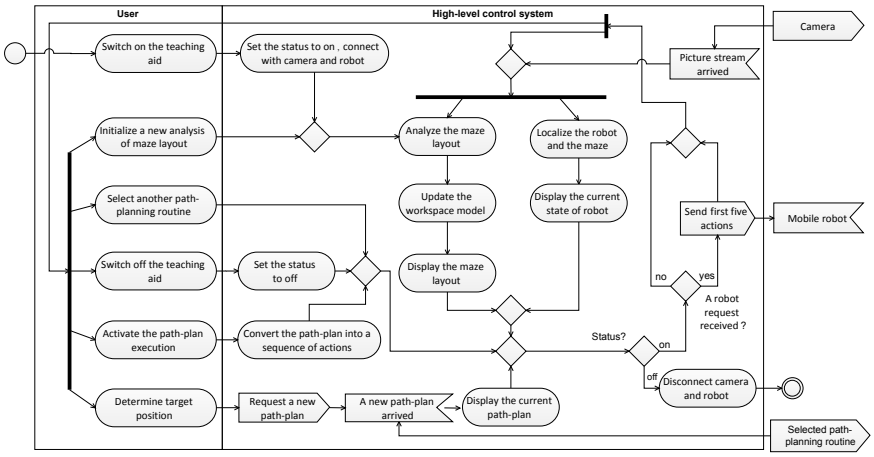

Fig. 3. UML activity diagram of the teaching aid with a special emphasis on the user and the high-level control system.

The communication of the software with the robot is usually started on the request of the robot. The only exception is the situation when the robot does not move. Once a request is received, the software sends an actual schedule of actions. Specifically, first five actions remaining in an actual sequence of actions are sent to the robot. In the case that all actions were executed by the robot, a command 'stop' is sent [18, 8]. Within the initialization phase, no sequence of actions exists and the robot does not move, therefore the communication with the robot is never established within the first pass.

Once the initialization is completed, entering commands via the GUI is allowed. Let us focus on the control elements. Using the button 'Analyse labyrinth', the sequence of actions described above is invoked. Using the button 'Select routine', the analysis is performed in the same way as was described in the second paragraph. Using the button 'GO!', the conversion of an actual path-plan into the sequence of actions is invoked. The new sequence of actions is then transmitted to the robot as was described in the third paragraph. Using the button 'Turn off' leads to a change of the status from 'on' to 'off'. It might be noted that the status is verified in each pass before communication with the robot is served. Once the status 'off' is identified, the software is turned off.

Once a new target position is selected, the connected path-planning routine is requested for a new path-plan. The path-plan is saved in memory of the high-level control system for its later conversion to the sequence of actions. Next, the path-plan is displayed in the GUI. It might be noted in this context, that the position and orientation of the robot are refreshed independently in a predetermined period. Within this loop, the position of the camera is also

verified. Once a change of its position is detected, the software tries to adapt to the new situation.

4 Testing the Proposed Solution

The software part of the high-level control system was created according the requirements which were stated in section 3. The testing of the developed software was carried out by testers. They assessed the response of the teaching aid using test scenarios. The test scenarios were designed with respect to all requirements of the teaching aid.

4.1 Test Scenarios

In this subsection, all used test scenarios are detailed. Each test scenario was designed to verify one specific aspect of the teaching aid. All generally valid assumptions, which are relevant for a scenario, are given first. The tests scenarios consist of test cases. The following notation is used for their description: **Important characteristics describing a current state of the teaching aid** > *an action performed by the user* (optional) > consequences. A symbol + is used as a logic connector AND in the state description (in bold). Aspects describing a response of the teaching aids are separated either using a semicolon or using an arrow. If the order of events does not matter, the semicolon is used. Otherwise, the arrow is used to emphasize a sequence of two events.

Scenario 1 Check the start of the software

Assumptions: the software is switched off but the hardware is switched on; the supervisor starts the software.

a) **The camera is on while the robot is off** > the buttons are displayed but they are inactive except the 'Turn off' button; the scheme of the maze layout is displayed; an error message is displayed; the software tries to establish a connection with the robot.
b) **The robot is on while the camera is off** > the buttons are displayed but they are inactive except the 'Turn off' button; an error message is displayed; the software tries to establish a connection with the camera.
c) **Both the robot and the camera are off** > the buttons are displayed but they are inactive except the 'Turn off' button; an error message is displayed; the software tries to establish connections both with the robot and the camera.
d) **Both the robot and the camera are on + the robot is in the maze** > the buttons are displayed but only the buttons 'Turn off' and 'Analyse maze' are active; the scheme of the maze layout is displayed; the actual position and orientation of the robot is displayed.
e) **Both the robot and the camera are on + the robot is not in the maze** > the buttons are displayed but only the buttons 'Turn off' and 'Analyse maze' are active; the scheme of the maze layout is displayed; a warning message is displayed.

Scenario 2 Check the communication with path-planning routines

Assumptions: the software is switched on.

a) **No routine is connected** > the button 'Select routine' is inactive.

b) **At least one routine is connected** > the button 'Select routine' is active.

Scenario 3 Check the selection of the path-planning routine

Assumptions: the software is switched on, at least one path-planning routine is connected.

a) **No routine is selected** > *the user selects a routine* > the selection menu is opened → the selected routine is highlighted → the selection menu is closed.

b) **A routine is selected** > *the user selects another routine* > the selection menu with a highlighted routine is opened → the newly selected routine is highlighted while the previous highlight is cancelled → the selection menu is closed.

c) **A routine is selected** > *the user selects the identical routine* > a selection menu with a highlighted routine is opened → the user selection is highlighted → the selection menu is closed.

d) **A routine is selected** > *the user cancels the selection* > a selection menu with a highlighted routine is opened → the selection menu is closed.

e) **Connection with a routine is lost within the selection process** > the routine is erased from the list.

Scenario 4 Check the communication between the software and the camera

Assumptions: the software is switched on.

a) **The camera does not stream** > the buttons 'Analyse labyrinth' and 'GO!' are inactive; an error message is displayed.

b) **The camera streams + position of the camera is out of an operating range** > the buttons 'Analyse labyrinth' and 'GO!' are inactive; an error message is displayed.

c) **The camera streams + the robot is in the maze + position of the camera is within the operating range** > the button 'Analyse labyrinth' is active (when a path-plan exists, the button 'GO!' is active as well); the actual position and orientation of the robot is displayed.

d) **The camera streams + the robot is not in the maze + position of the camera is within the operating range** > the button 'Analyse labyrinth' is active; the button 'GO!' is inactive; a warning message is displayed.

Scenario 5 Check the selection of the target position

Assumptions: the software is switched on; the maze layout is displayed; the camera streams; the robot is in the maze.

a) **No path-planning routine is selected** > *the user selects a position* > the selection is not highlighted.

b) **A path-planning routine is selected + the connection with routine works + the robot rests + no target position is selected** > *the user selects a target position* > the selection is highlighted; a path-plan is displayed.

c) **A path-planning routine is selected + the connection with the routine works + the robot either rests or moves + a target position is selected** > *the user selects the identical target position* > the selection is highlighted; a path-plan respecting a current position of the robot is displayed instead of the original one.

d) **A path-planning routine is selected + the connection with the routine works + the robot moves + a target position is selected** > *the user selects a new target position* > the selection is highlighted; a path-plan respecting a current position of the robot is displayed instead of the original one.

e) **A path-planning routine is selected + the connection with the routine fails** > *the user selects a position* > the selection is highlighted; an error message is displayed.

Scenario 6 Check the initialization of the robot's movement

Assumptions: the software is switched on; the maze layout is displayed; the camera streams; the robot is in the maze; a valid path-plan is available.

a) **The robot rests** > *the user presses the button 'GO!'* > the button selection is highlighted; the robot starts to move.

b) **The robot moves** > *the user presses the button 'GO!'* > the button selection is highlighted; the robot continues moving; a latest path-plan is followed.

c) **The robot is already in the target position** > *the user presses the button 'GO!'* > the button selection is highlighted; a warning is displayed; the robot rests.

Scenario 7 Check the analysis of the maze layout

the software is switched on; the camera streams; position of the camera is within the operating range.

a) **The maze is assembled according to rules** > *the user presses the button 'Analyse labyrinth'* > the button selection is highlighted; the scheme of a current maze layout is displayed instead of the original one.

b) **The maze is not assembled according to rules** > *the user presses the button 'Analyse labyrinth'* > the button selection is highlighted; an error message is displayed.

Scenario 8 Check the communication between the software and the robot

Assumptions: the software is switched on; the camera streams; the position of the camera is within the operating range; the robot is in the maze; a valid path-plan is available.

a) **The connection with the robot is established + the robot is being executed the path-plan** > the robot successfully reaches the target position according the path-plan.

b) **The connection with the robot is lost** > an error message is displayed.

Scenario 9 Check the shutting down of the software

Assumptions: the software is switched on.

a) **The robot moves** > *the user presses the button 'Turn off'* > the button selection is highlighted → the application is closed; the robot finishes the movement according to actions in its memory.
b) **The robot rests** > *the user presses the button 'Turn off'* > the button selection is highlighted → the application is closed; the robot rests.

4.2 Evaluation Results

Two testers evaluated the response of the teaching aid using the test scenarios. The goal of the evaluation experiments was the verification of the system performance. We required a hundred percent positive response of the teaching aid in accordance with all the test cases. The teaching aid equipped with a final version of the presented software solution, fulfilled this requirement.

5 Conclusion

The inclusion of practical works into AI course curricula is a rewarding step which supports an understanding of a studied topic. The discussed teaching aid was aimed at practicing search strategies. Specifically, searching for a single pair shortest path is the task to be practiced. However, the using of the teaching aid is not limited to this single task. For example, it could be used for practicing algorithms aimed at a traveling salesman problem. Naturally, a modification of the teaching aid would be required in this case.

The modification of the teaching aid, with a view to its alternative use, would in fact require only an adaptation of the software part of the high-level control system. A degree of the adaptation is problem dependent; however, the presented description of the software would significantly simplify the adaptation process. Thus, we hope that the teaching aid will find a wide range of applications in the near future.

Acknowledgement: The work was supported by the Funds of University of Pardubice, Czech Republic, grant No. SGS-2017-027.

References

1. Cuéllar, M.P., Pegalajar, M.C.: Design and implementation of intelligent systems with LEGO mindstorms for undergraduate computer engineers. Computer Applications in Engineering Education 22(1), 153–166 (2014)
2. van Delden, S.: Industrial robotic game playing: An ai course. Journal of Computing Sciences in Colleges 25(3), 134–142 (jan 2010)
3. Dogmus, Z., Erdem, E., Patoglu, V.: React!: An interactive educational tool for ai planning for robotics. IEEE Transactions on Education 58(1), 15–24 (Feb 2015)
4. Fernandes, M.A.C.: Problem-based learning applied to the artificial intelligence course. Computer Applications in Engineering Education 24(3), 388–399 (2016)
5. Greenwald, L.G., Artz, D., Mehta, Y., Shirmohammadi, B.: Using educational robotics to motivate complete AI solutions. AI Magazine 27(1), 83–95 (2006)

6. Harlan, R.M., Levine, D.B., McClarigan, S.: The khepera robot and the krobot class: A platform for introducing robotics in the undergraduate curriculum. SIGCSE Bull. 33(1), 105–109 (feb 2001)
7. IEEE Computer Society: IEEE Std 802.11-2012. Tech. rep. (March 2012), http://standards.ieee.org/getieee802/download/802.11-2012.pdf
8. Ihnát, V.: Communication protocol design and software development for the wireless data transmission among mobile robots and MATLAB. Master's thesis, University of Pardubice, Pardubice, Czech Republic (jun 2016)
9. Kumar, A.N.: Three years of using robots in an artificial intelligence course: Lessons learned. Journal on Educational Resources in Computing 4(3) (sep 2004)
10. Mariška, M., Doležel, P.: Multi agent environment for modelling and testing of cooperative behaviour of agents. Advances in Intelligent Systems and Computing 289, 301–306 (2014)
11. Race, P.: The Lecturer's Toolkit. Routledge, 711 Third Avenue, New York, NY 10017, 4th edn. (2015)
12. Rubenstein, M., Ahler, C., Hoff, N., Cabrera, A., Nagpa, R.: Kilobot: A low cost robot with scalable operations designed for collective behaviors. Robotics and Autonomous Systems 62(7), 966–975 (2014)
13. Russell, S., Norvig, P.: Artificial Intelligence: A Modern Approach. Prentice Hall Press, 3rd edn. (2009)
14. Siegwart, R., Nourbakhsh, I.: Introduction to Autonomous Mobile Robots. Bradford Book (2004)
15. Škrabánek, P.: Labyrinth arrangement analysis based on image processing. In: Mendel 2015. Advances in Intelligent Systems and Computing, vol. 378, pp. 305–316. Springer International Publishing (2015)
16. Škrabánek, P., Doležel, P.: Attractive robot's design suitable for image processing. In: Proceedings of the Mendel 2014: 20th International Conference on Soft Computing. pp. 217–222. University of Technology Brno (25–27 June 2014)
17. Škrabánek, P., Mariška, M., Doležel, P.: The time optimal path-planning of mobile robots motion respecting the time cost of rotation. In: Proceedings of the 20th International Conference on Process Control. pp. 232–237. STU Bratislava (9–12 June 2015)
18. Škrabánek, P., Vodička, P., Yildirim-Yayilgan, S.: Control system of a semi-autonomous mobile robot. In: Proceedings of the 14th IFAC Conference on Programmable Devices and Embedded Systems. pp. 460–469. Brno University of Technology, Brno/Lednice, Czech Republic (5–7 October 2016)