THE DESIGN OF A MODULAR MOBILE ROBOT FOR VISION BASED HUMAN ROBOT INTERACTION RESEARCH

Žiga Pisar and Primož Podržaj

Faculty of Mechanical Engineering,
University of Ljubljana, Slovenia

ABSTRACT

This paper presents the design of a mobile robot used for a research of robot human interaction. The communication is achieved via visual information. For this purpose two cameras are mounted on a robot in such a way that pan/tilt rotation is possible. Such a setup resembles a human head and can give some feedback to a human with the movement of the head. For the control and communication purposes a Raspberry Pi computer was used as it is inexpensive but highly capable open platform. The camera and robot movements are accessible through a web user interface designed on a modified PiBorg code. Throughout the design process the main emphasis was on the low cost and high modularity of the mobile process, so that it can be easily used for other applications as well.

KEYWORDS

Mobile Robot, Robot Vision, Human Robot Interface, Raspberry Pi, Web User Interface

1. INTRODUCTION

In future mobile robots will play an ever increasingly important role in our lives. Therefore there is no surprise that up to the moment of writing of this paper there were many papers just in this year dealing with the interaction between a mobile robot and its interface with humans and their environment [1-15]. One of the main means of interaction is vision. Such robots can perform various tasks such as, assessing the environment [2,9-10, detecting a human being and/or performing some tasks related to its state [1,4,8,12]. For a mobile robot to have vision based capabilities it is not necessary to use its own camera. Especially for indoor applications with many mobile robots, the so called intelligent space might be a more suitable approach [16,17]. When mobile robots are applied outdoors, or when there are fewer of them, cameras mounted on mobile robots tend to be a better solution. This not only makes it possible for robots to be more human or pet like (cameras and lenses resemble a human head or an eye, especially if the head can be tilted), but it also enables a mobile robot to move the camera in different positions, in order to get a more appropriate image(s) and consequently better quality assessment of a human or environment.

Mobile robots can be used for various tasks:

- Assistance for elderly or otherwise impaired people [1,7,8,18-19]
- Surveillance and security applications [15,20-22]
- Industrial applications (welding, painting, etc.) [23-24]

DOI: 10.14810/elelij.2017.6401

In all the above mentioned tasks (and many others) it is of main importance for the robot to be able to interact well with the environment. As already mentioned, vision is most commonly applied for that purpose. For this reason a decision was made to design a low-cost mobile robot for vision related research in the field of mobile robotics. The robot should have a head-like mounting which makes it possible to attach two cameras, which can rotate in pan/tilt direction. It should be of modular design, so that components can be changed for various applications. The control/communication should be done on a low-cost computer system, which can easily be mounted on a mobile robot platform. It also should be based on open-source software, which makes is possible to easily include solutions already available.

2. ROBOT DESIGN

The design process will be presented in the way the decisions about the system components were made.

2.1. Robot Drive

In general, there are three possibilities of robot drives. The first one is the so called differential drive, shown in Figure 1.

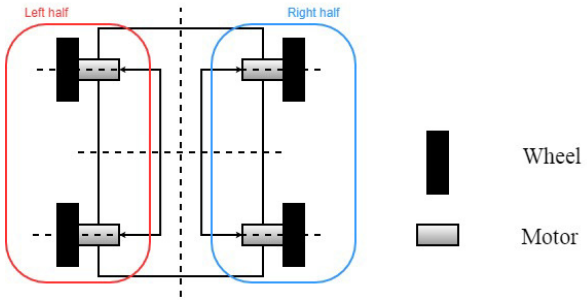

Figure 1. Differential drive

It is the cheapest and can also be applied in outdoor environment where wheels are replaced by tracks. It was for these reasons, that it was chosen for our robot. The second one is the Ackermann drive, shown in Figure 2.

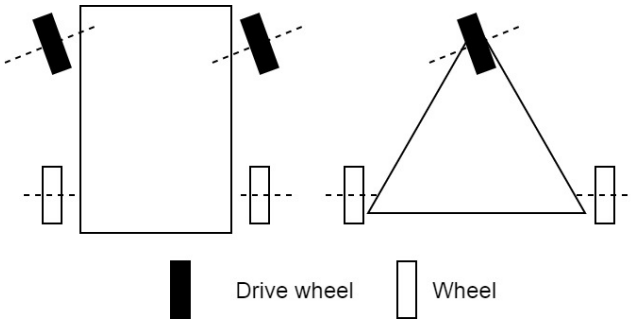

Figure 2. Ackermann drive

In this case a steering mechanism is needed for at least one wheel. Such a configuration is needed if a wheeled robot is to be driven in an outdoor environment. It is of course more complicated, more expensive and has a lower ability to manoeuvre, compared to the differential drive. The

third option is basically the differential drive with a set of special wheels called Mecanum [25]. Such a configuration, for example, makes it possible to move the robot in lateral direction by rotating front wheels forwards and back wheels backwards or vice versa. It is however limited to indoor environment.

2.2. Chassis

The decision about the chassis is the most fundamental one, because it gives the shape of the mobile robot. In our case, an aluminium plate was used because of its low-cost, good weight-to-strength ratio and easy machining. The top view of the chassis is shown in Figure 3.

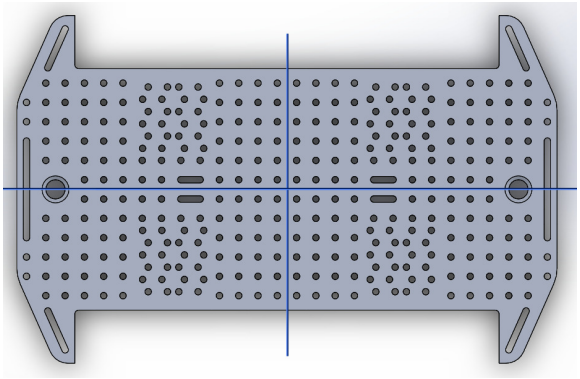

Figure 3. The top view of the chassis (280mm x 180mm x 4mm)

The chassis should be thick enough in order not to twist under the weight of the robot. In our case the assumed weight was 4 kilos, so we decided for a 4mm thickness. A grid of holes (3.4mm in diameter) with a 10mm step was drilled into the chassis in order to offer high modularity of the end product. If needed, various components can be added later. Another set of holes was drilled specifically for each drive motor. The attachment of each motor is shown in Figure 4.

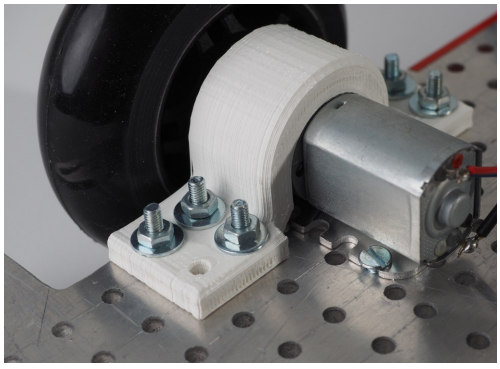

Figure 4. The attachment of the motor on the chassis

A special fixture (seen in white colour in Figure 3) was added in order to ensure high rigidity of the motor. This is especially important when a differential drive is chosen.

2.3. Pan/Tilt System

A very important component of the vision based mobile robot system, is the so called pan/tilt system. If the cameras are rigidly attached to the robot, its field of view is very limited. In

addition to that, the whole robot has to move in order to observe different points of interest. Pan/tilt system shown in Figure 4, offers high versatility and can take relatively heavy loads.

Figure 5. The SPT200 pan/tilt system with HS-5685MH servomotors

Two specially designed parts are needed in order to first fix the pan/tilt system to the chassis, and then to fix the cameras on top of it. Both were designed in Solidworks and then 3D printed from ABS plastics.

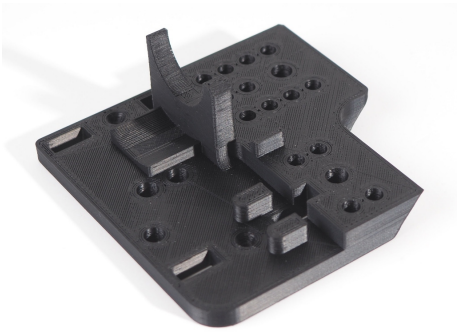

Figure 6. The camera mounting made by 3D printing

Although this part has to be redesigned if other cameras/lenses are used, it doesn't represent a big additional cost because 3D printing is now a quite inexpensive technique.

2.4. Controller and Power Supply

The controller not only has to control the movement of the electric motors used to drive the robot, but also offer highly capable connectivity between the mobile robot system and the PC or a mobile phone. Therefore, our decision to use the Raspberry Pi computer seems a logical one. Raspberry Pi is basically a credit card size PC [26]. Of course in comparison to a typical PC it lacks the peripherals, but these can easily be added if a specific application demands them. In relation to its capabilities, it is really inexpensive (around 40EUR). It is also a highly popular open-source platform with a lot of libraries for specific tasks, which makes it easier to realize complex tasks. It has 64bit architecture with quad-core ARM v8 processor, 1GB of RAM. The size of its memory depends on the size and capabilities of a microSD card. The minimal requirement is 8GB due to the operating system installation. It can be powered via an USB port or GPIO pins. In our case the second option was used. For this purpose a BattBorg battery module was used. It transforms input voltages in the 7V~36V range to the output voltage of 5V, which is

needed in the case of Raspberry Pi. In our case the BattBorg module was used in connection with the PicoBorg module as shown in Figure 7.

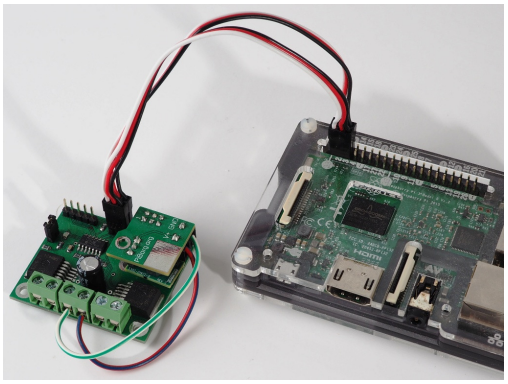

Figure 7. The connection between Raspberry Pi, PicoBorg and BattBorg

PicoBorg module is namely used in order to easily drive the electric motors for wheels.

2.5. The Result of the Design Process

The final product (mobile robot with all the major and some minor components) is shown in Figure 8.

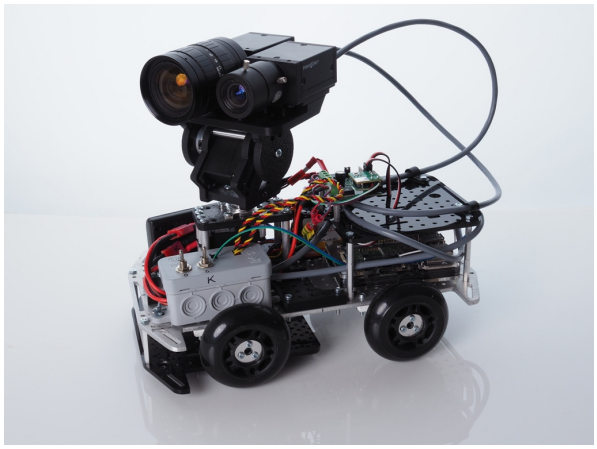

Figure 8. Photo of the whole robot

Some minor (but nevertheless very important) components like batteries and their fixation, switches, Raspberry Pi, PicoBorg and BattBorg fixation have been omitted in the above text. But photo of the robot shown in Figure 8 can give at least some idea about the applied solutions. The camera mount was designed in such manner that the center of mass for the system of bodies (cameras, lenses and camera mount itself) lies on the vertical axis of the pan rotation, thus balancing the robot's head. It also takes into account that no obstruction in the field of view of both cameras occurs.

2.6. Bill of Materials

Table 1. Robot build costs

Components	Weight	Unit price
4x Pololu 20D 73:1 DC motor	400 g	18,00 €
Al Chassis	345 g	74,00 €
2x Grasshopper 3 Camera w/ adapter mount	204 g	/
2x Camera lenses	358 g	/
Raspberry Pi w/ case	97 g	42,99 €
PicoBorg Reverse	17 g	35,10 €
BattBorg	10g	16,45 €
4x Roller wheels	280 g	8,96 €
11,1V 3700 mAh LiPo battery	230 g	37,80 €
SPT200 Pan Tilt System	156 g	38,21 €
2x HiTec Servo	120 g	33,23 €
Second floor acrylic base plate	93 g	27,00 €
ABS Camera mount	38 g	30,00 €
ABS PT system mount	37 g	30,00 €
7,4V 3700 mAh LiPo battery	150 g	36,00 €
GPIO cable, USB cable, Switches	53 g	/

In the Table 1., all the components of the mobile robot and their prices are shown. The cameras were added for weight calculation. The total cost of the robot excluding camera and lenses is less than 500EUR. The cameras were excluded, because for our mobile robot their cost is substantial (around 1000 for each camera with lens). The quality (and consequently price) of cameras is very dependent on the application. The total weight of the robot amounts to 2.6 kilograms.

3. ROBOT CONNECTIVITY

Raspberry Pi 3 model B is a single-board computer which features on-board connectivity standards like WiFi and Bluetooth. In combination with its upgraded hardware characteristics compared to previous generations, it has become a low-cost solution for designing teleoperated mobile robots.

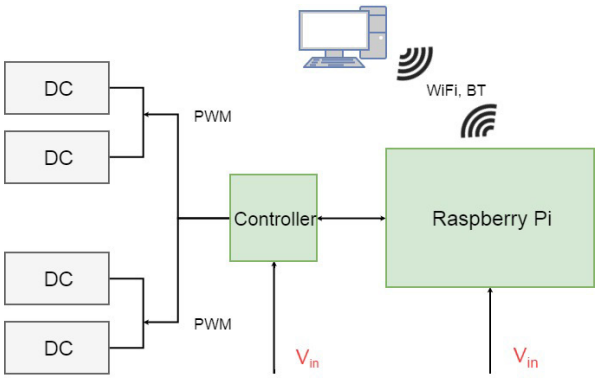

Figure 9. Visual representation of robot connectivity

Serving as the robot's main CPU, Raspberry Pi runs an open-source PiBorg software. The source code sets up a local server, creates a WUI and enables control of the DC motors. By typing the IP address of the computer into a PC or a mobile phone web browser, we can access the user interface and take control over the robot's movement.

In general, the code consists of three programming languages: Python, HTML and Java. While HTML is used for creating buttons and covers the visual design, the latter enables the buttons to carry out functionalities by calling different subpages, for example *<ip-address>/reboot*, which would execute the code under the *elif getPath.startswith('/reboot'):*. The main language is Python, which acts as a frame. It is important to note, that in order for the mobile robot to function properly, the robot and the PC, from which the robot is being operated, have to share the same WiFi network [27]. While the original PiBorg code in combination with the PicoBorg Reverse motor controller provides general movement for the robot, for example driving forward, backward and rotating on the spot, it does not offer any control over the movements of the robot head. Thus, it was imperative to find a solution that would have a low impact on the robot architecture and would be compatible with already used hardware. For integrating two additional motors for the pan and tilt rotations, following the low-cost guideline, we have made use of the ServoBlaster library intended for Raspberry Pi platforms. It provides control for up to 21 servos using only GPIO pins with no additional components or drivers in-between, therefore making it suitable for small to medium sized robots. Servo positioning is achieved by entering commands directly into the LXTerminal. Instead of specifying absolute positions in either microseconds or milliseconds, we can also drive the servos by specifying relative head rotations via adding plus or minus prefixes to the step-size value. We start off by calling the *servod* application using administrator privileges and stating that the pins 11 (GPIO17) and 16 (GPIO 23) of header one (P1) will be used for outputting a PWM signal. Other options include adjusting the lower and upper allowed pulse width of both servos [28].

sudo servod --p1pins=11,16 --min=750us --max=1200us

The Raspberry Pi is now configured to control servos or other analog circuitry. For example, the following line commands the servo on pin 11 to move in the positive direction relative to the current position for a step of 10 microseconds. This will not work unless the previous command is issued first.

echo P1-11=+10 >/dev/servoblaster

3.1. Integration of Servo Control

As mentioned previously, we control servos by entering commands into the terminal. Since the code, that creates the web user interface only contains lines of code for moving the robot itself, it is necessary to add buttons that generate ServoBlaster commands. This is done by running shell commands inside Python. For this we used a *subprocess.call()* function of a Python module called *subprocess*, which allows us to spawn processes, or more precisely, run external command line instructions by way of passing a string variable into a command line interface – terminal [29]. First we add a new section to the user interface where we place an array of buttons using HTML. The idea behind is to add a button for every direction of rotation - two for panning and two for tilting. With a press of a button, we access subpages, for example */rotate_left*. The main Python code checks whether the subpage was accessed or not. If the condition is true, it executes the mentioned s*ubprocess.call()* function that contains a string variable. The string variable is in turn the ServoBlaster command which is sent to the terminal. The following is an example of a basic *subprocess* function with a ServoBlaster command as a string variable.

subprocess.call('echo P1-11=+10 >/dev/servoblaster', shell=True)

4. WEB USER INTERFACE

Control panel of the robot consists of two different sets of buttons. The arrow buttons are there for the basic movements of the robot, which are forwards, backwards, rotation on the spot

clockwise and counter clockwise. There are also two left and right buttons, which only activate left or right side motors respectively. Underneath the first set, there are six buttons intended for controlling the head of the mobile robot. To start using the servos after every restart of the Raspberry Pi computer, it is important to first initialize and configure the GPIO pins, which is done when the INIT button is pressed. This also positions the cameras into a neutral position – horizontal and pointing forward.

The STOP button acts as a kill-switch for the servo control, however it still locks-in the last set position of the servos for a steady image capture. The INIT button has to be pressed after every new session or after stopping the servos. Panning and tilting is done by clicking on either LEFT/RIGHT (pan) or UP/DOWN (tilt). At the footer of the page there are two shortcut utilities for Raspberry Pi reboot and shutdown, which enhance the user experience. The speed is adjusted with the slider. The modified WUI is shown in Figure 10.

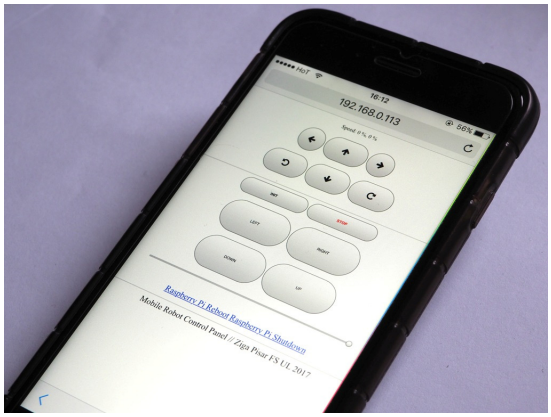

Figure 10. Photo of the WUI on a mobile phone

5. CONCLUSIONS

The paper presents a design process (both from hardware and software point of view) of a highly capable but inexpensive mobile robot. Its main field of application is mobile robot vision research. It can however easily be adapted for other purposes as well. The presented software solution makes it possible to drive the robot and move its cameras via a mobile phone or PC. When this level of expertise is achieved it can easily be modified/upgraded to various more complex applications. For this reason we assume that the presented design process is and can be very helpful for anyone trying to build a capable mobile robot platform.

REFERENCES

[1] Luna, M. A., Moya, J. F., Aguilar, W. G., & Abad, V. (2017). Mobile robot with vision based navigation and pedestrian detection.

[2] Krajník, T., Fentanes, J. P., Santos, J. M., & Duckett, T. (2017). "Fremen: Frequency map enhancement for long-term mobile robot autonomy in changing environments", IEEE Transactions on Robotics.

[3] Fäulhammer, T., Ambruş, R., Burbridge, C., Zillich, M., Folkesson, J., Hawes, N., Jensfelt, P. & Vincze, M. (2017). Autonomous learning of object models on a mobile robot. IEEE Robotics and Automation Letters, Vol. 2, No. 1, pp26-33.

[4] Vassallo, C., Olivier, A. H., Souères, P., Crétual, A., Stasse, O., & Pettré, J., (2017) "How do walkers avoid a mobile robot crossing their way?", Gait & posture, Vol. 51, pp97-103.

[5] Marinho, L. B., Almeida, J. S., Souza, J. W. M., Albuquerque, V. H. C., & Rebouças Filho, P. P., (2017) "A novel mobile robot localization approach based on topological maps using classification with reject option in omnidirectional images", Expert Systems with Applications, Vol. 72, pp1-17.

[6] Ko, B., Choi, H. J., Hong, C., Kim, J. H., Kwon, O. C., & Yoo, C. D. (2017) "Neural network-based autonomous navigation for a homecare mobile robot", In Big Data and Smart Computing (BigComp), 2017 IEEE International Conference on, pp403-406.

[7] Nguyen, Q. H., Vu, H., Tran, T. H., & Nguyen, Q. H, (2017) "Developing a way-finding system on mobile robot assisting visually impaired people in an indoor environment", Multimedia Tools and Applications, Vol. 76, No. 2, pp2645-2669.

[8] Antonello, M., Carraro, M., Pierobon, M., & Menegatti, E., (2017) "Fast and Robust Detection of Fallen People from a Mobile Robot", arXiv preprint arXiv:1703.03349.

[9] Guo, S., Diao, Q., & Xi, F., (2017) "Vision Based Navigation for Omni-directional Mobile Industrial Robot", Procedia Computer Science, Vol. 105, pp20-26.

[10] Savkin, A. V., & Li, H., (2017) "A safe area search and map building algorithm for a wheeled mobile robot in complex unknown cluttered environments", Robotica, pp1-23.

[11] Gupta, M., Kumar, S., Behera, L., & Subramanian, V. K., (2017) "A novel vision-based tracking algorithm for a human-following mobile robot", IEEE Transactions on Systems, Man, and Cybernetics: Systems, Vol. 47, No. 7, pp1415-1427.

[12] Carton, D., Olszowy, W., Wollherr, D., & Buss, M., (2017) "Socio-Contextual Constraints for Human Approach with a Mobile Robot", International Journal of Social Robotics, Vol. 9, No. 2, pp309-327.

[13] He, F., Guo, Y., & Gao, C., (2017) "Human segmentation of infrared image for mobile robot search", Multimedia Tools and Applications, pp1-14.

[14] Lakrouf, M., Larnier, S., Devy, M., & Achour, N., (2017) "Moving Obstacles Detection and Camera Pointing for Mobile Robot Applications", In Proceedings of the 3rd International Conference on Mechatronics and Robotics Engineering, pp. 57-62.

[15] Jing, X., Gong, C., Wang, Z., Li, X., & Ma, Z., (2017) "Remote Live-Video Security Surveillance via Mobile Robot with Raspberry Pi IP Camera", In International Conference on Intelligent Robotics and Applications, pp. 776-788. Springer, Cham.

[16] Lee, J. H., & Hashimoto, H., (1999) "Intelligent Space, its past and future", In Industrial Electronics Society, 1999. IECON'99 Proceedings. The 25th Annual Conference of the IEEE Vol. 1, pp. 126-131.

[17] Podržaj, P., & Hashimoto, H., (2008) "Intelligent space as a framework for fire detection and evacuation", Fire technology, Vol. 44, No. 1, pp65-76.

[18] Gross, H. M., Mueller, S., Schroeter, C., Volkhardt, M., Scheidig, A., Debes, K., Richter, K. & Doering, N., (2015) "Robot companion for domestic health assistance: Implementation, test and case study under everyday conditions in private apartments", In Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on, pp5992-5999.

[19] Fischinger, D., Einramhof, P., Papoutsakis, K., Wohlkinger, W., Mayer, P., Panek, P., Hofmann, S., Koertner, T., Weiss, A., Argyros, A. & Vincze, M., (2016) "Hobbit, a care robot supporting independent living at home: First prototype and lessons learned", Robotics and Autonomous Systems, Vol. 75, pp60-78.

[20] Saitoh, M., Takahashi, Y., Sankaranarayanan, A., Ohmachi, H., & Marukawa, K. (1995) "A mobile robot testbed with manipulator for security guard application", In Robotics and Automation, 1995. Proceedings., 1995 IEEE International Conference on, Vol. 3, pp. 2518-2523.

[21] Flann, N. S., Moore, K. L., & Ma, L., (2002) "A small mobile robot for security and inspection operations", Control Engineering Practice, Vol. 10, No. 11, pp1265-1270.

[22] Treptow, A., Cielniak, G., & Duckett, T., (2005) "Active people recognition using thermal and grey images on a mobile security robot", In Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on, pp2103-2108

[23] Kam, B. O., Jeon, Y. B., & Kim, S. B., (2001) "Motion control of two-wheeled welding mobile robot with seam tracking sensor" In Industrial Electronics, 2001. Proceedings. ISIE 2001. IEEE International Symposium on, Vol. 2, pp851-856

[24] Keerthanaa, P., Jeevitha, K., Navina, V., Indira, G., & Jayamani, S, (2013) "Automatic wall painting robot", International Journal of Innovative Research in Science, Engineering and Technology, Vol. 2, No. 7, pp3009-3023.

[25] Doroftei, I., Grosu, V., & Spinu, V, (2007) Omnidirectional mobile robot-design and implementation, In Bioinspiration and Robotics Walking and Climbing Robots, InTech.Publishers.

[26] Upton, E., Duntemann, J., Roberts, R., Everard, B., & Mamtora, T., (2016) Learning Computer Architecture with Raspberry Pi, John Wiley & Sons.

[27] Freeburn Robotics Limited. (2017, Jul. 3). Diddyborg-web [Online]. Available: https://github.com/piborg/diddyborg-web

[28] Hirst, R. (2012, Aug. 19). ServoBlaster [Online]. Available: https://github.com/richardghirst/PiBits/tree/master/ServoBlaster

[29] Python Software Foundation. (2017, Sep. 10). Subprocess management [Online]. Available: https://docs.python.org/2/library/subprocess.html

AUTHORS

Žiga Pisar finished 1st cycle Professional Study Programme in 2017, earning a Bachelor degree of Applied Science in Mechanical Engineering. He is currently working on his Master's Study Programme in the field of mechatronics. His fields of interests include robotics, machine design and computer programming.

Primož Podržaj received the MSc degree and the PhD degree in automatic control from the University of Ljubljana, Ljubljana, Slovenia, in 2000 and 2004, respectively. He is an Associate Professor with the Faculty of Mechanical Engineering, University of Ljubljana. His research interests include control systems, control system design, and industrial automation.