Control Design of an Office Mail Delivery Robot Based on the Festo Robotino Platform

Nathir A. Rawashdeh, Hisham Alwanni, Nazeeh Sheikh Ali, Bader Al Afghani
Department of Mechatronics Engineering
German Jordanian University
Amman, Jordan
Email: nathir.rawashdeh@gju.edu.jo

Abstract- This paper details the hardware and software development of an indoor mail robot that is based on the Festo Robotino platform. The Festo Robotino is a battery powered indoor robotic platform equipped with omni-directional wheels, sensors, a controller, a camera, a simulator, and wireless connectivity. The implementation of the robot entailed the mechanical expansion of the Robotino base robot to include compartments for holding office mail and a touch screen interface for entering commands. An arm was added to knock on closed office doors. The robot receives user commands through a PHP website, which communicates with the Robotino controller through Python software functions. This work was a bachelor level graduation project spanning two semesters at the Department of Mechatronics Engineering at the German Jordanian University. It demonstrates the versatility of the Festo Robotino platform in bachelor level education and even master level research in the areas of vision, navigation, autonomy, control, and localization. The mechanical, electrical, and software, i.e., mechatronic design are presented. The robot was tested indoors between two offices and a home position.

Introduction

Ground robots usually carry an array of sensors to implement obstacle avoidance[1,2,3], mapping, and autonomous navigation[4]. Self-driving cars for example, can use cameras and visual odometry algorithms[5,6] to calculate vehicle movement in global positioning system (GPS) denied environments such as tunnels and urban canyons, as well as indoor environments such as parking structures. Indoor robots often use a differential drive or omnidirectional wheels in order to gain maneuverability and a zero turning radius. Examples of indoor robot applications include robotic vacuum cleaners, tele-presence robots, person following robots[7,8], as well as robots for warehousing, security, store shelf restocking and restaurant catering. This paper details the design and development of an indoor office mail robot based on the Festo Robotino mobile robot platform, which features an omnidirectional drive system, graphical motion programming, and an array of sensors for obstacle avoidance, line following, and localization.

The work is a bachelor level graduation project spanning two semesters in the Department of Mechatronics Engineering at the German Jordanian University. The Robotino platform is versatile and expandable, making it ideal for educational and research projects[9,10]. Fig. 1 shows two Robotino based robots at the 2012 RoboCup World Championship in the logistics and production competition[11]. The project presented in this paper was performed by three mechatronics engineering students as their senior design project. The students had creative freedom in developing the technologies required to implement the robot's functionality. A mechatronics design approach was used, where the development of mechanical, electrical, and control software was performed concurrently to achieve an efficient and compact end result. The learning experience was enriched by the fact that each team member was building on their strengths while learning new problem solving skills from the others.

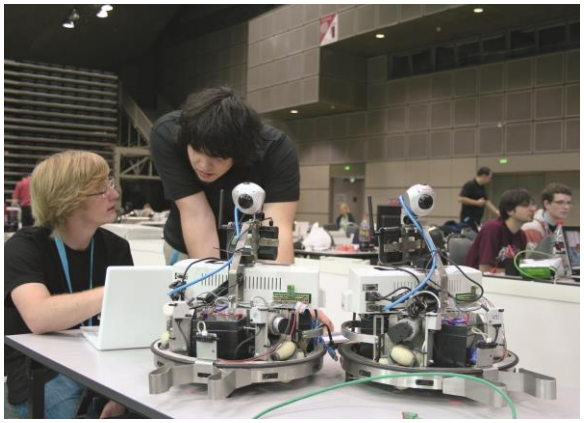

Fig. 1. Students working on Robotino at the RoboCup World Championship. Image from press release[11]

Current literature contains many projects and research implementations based on the Festo Robotino platform. Examples of articles based on Festo Robotino include: an e-learning remote laboratory based on the Robotino and other logistics management systems[12]; localization of Robotino in the Logistics League competition 2017[13]; speed control of Robotino by brain waves[14]; robot position control design and simulation[15,16]; energy saving path planning for the Robotino[17]; indoor localization[18]; control and trajectory generation[18,19] using the robotic operating system (ROS); multi-robot collaborative localization[20]; wireless connectivity in factories of the future[21]; obstacle avoidance using fuzzy a logic controller[4,22]; localization of multiple Robotinos in a warehouse setting[23].

The Festo Robotino platform is popular with students and researchers because it is adequately sized and can carry a relatively heavy payload of custom equipment. It weighs about 20 kg (44 pounds) and is about 30 cm (12 inches) high and 45 cm (18 inches) wide, with a carrying capacity of 30 kg (66 pounds) for custom hardware. In addition, it has flexible programming, simulation and interfacing options, an omnidirectional drive, as well as an array of sensors. Fig. 2 shows the Festo Robotino and the location of its main components. Table I shows the values of some of these main features.

Fig. 2. Main components of the Festo Robotino mobile robot platform. Picture from Robotino Manual[24]

Table I. Main Festo Robotino Features

Feature	**Specifications**
Power	two 4A 12Vdc lead-gel batteries in series. 4 hours operation time
Drive	three 24 Vdc motors with omnidirectional wheels
Dimensions	45 cm wide, 29 cm high
Computer	Embedded PC with a compact flash memory running Linux
Programming	Robotino View, C, C++, Java, .NET, MATLAB/Simulink, LabVIEW, ROS, Microsoft Robotics Developer Studio.
Connectivity	Wireless LAN communication, two USB, VGA, Ethernet
Weight and max. payload	Weighs 20 kg. Can carry 30 kg
Max. speed	10 km/h
Sensors	9 IR distance sensors (4-30cm), 3 wheel encoders, color camera, gyroscope, optical line detector, collision detection strip
Digital inputs	8 (24 volt)
Digital outputs	8 (24 volt)
Analog inputs	8 (0-10 volt)
Output relays	2

Mechanical Design

Using the Robotino as a base, a mechanical vertical chassis extension was designed to mount above it on three short columns and a round base plate as depicted in Fig. 3. It was designed to ergonomically suit the standing user with the top mounted tablet computer and mailbox door at waist level. The structure is made of stainless-steel sheet metal with a total weight of 2 kg and a slope of 50 degrees at the top (mail box) half. The tablet displayes the web interface code that includes buttons to open and close the delivery and mail box doors. There is also an internal trap door that separates the two boxes. This is activated to move mail from the delivery to the mailbox when the recipient user does not answer when the robot knocks on the office door using the knocker arm.

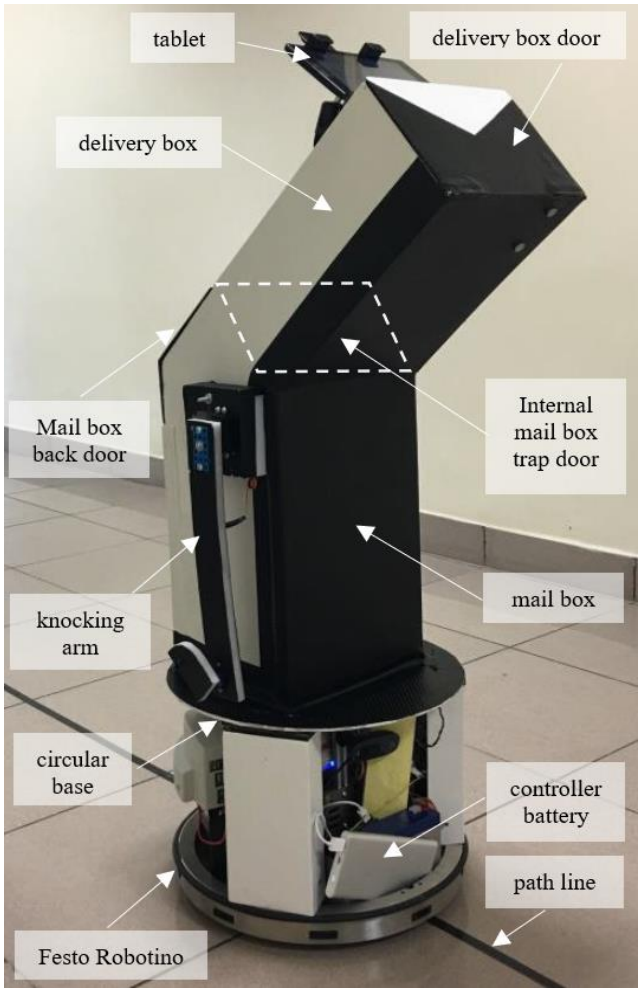

Fig. 3. Components of the developed Robotino based mail robot

To ensure the structure would not fall over forwards or snap under its own weight, a static stress analysis was performed using CATIA while applying the material specifications to the designed CAD model, adding constrains (clamp, virtual slider, and virtual bolt …. etc.), and using von Mises criteria to rate the maximum load capacity of each element model as shown in Fig. 4. The results show the points experiencing the most force, i.e., the back side between the delivery and mail boxes, and the rear side of the base plate connection to the rear column. To ensure the integrity of the body in more stressful than nominal use, the safety factor was chosen to be 2.1 for the base plate and 1.57 for the box chassis.

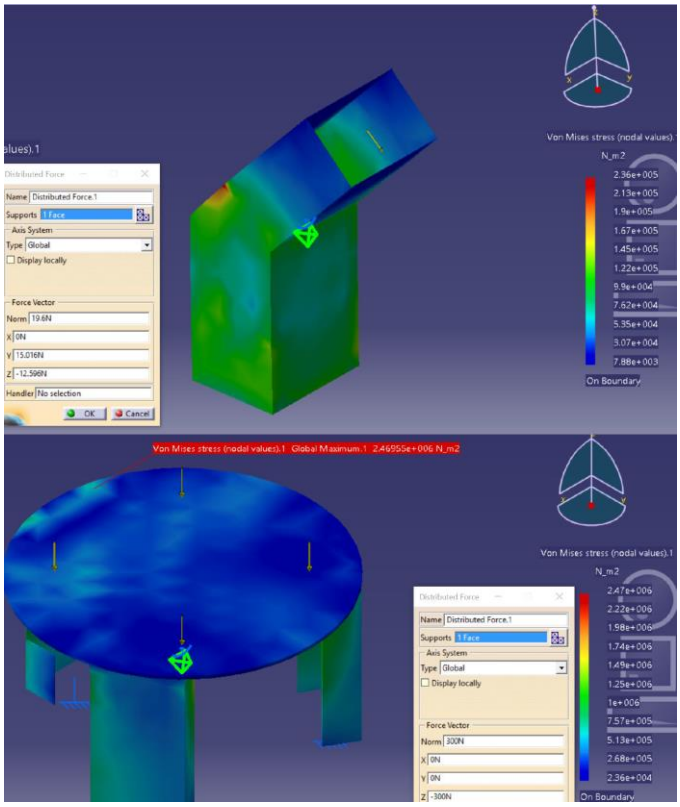

Fig. 4. Static load analysis using CATIA

Electrical Design

A Raspberry Pi 2 Model B with 1GB of RAM clocked at 900 MHz was used as the main control computer. It features WiFi connectivity and I^2C communication interfacing. It runs Python code to control the robot's movement and hosts the website code for the user office PCs and the mobile touchscreen tablet. The electrical design is illustrated in Fig. 5 and shows how the servo motors are connected to the (PCA9685 16-Channel, 6 Volt) servo motor driver board, which is actuated via the I^2C bus and pulse width modulation (PMM). There are 6 servo motors, while only 1 is depicted in the figure. These are used as follows: 1 for the delivery box door; 1 for the internal trap door; 1 for the rear mailbox door; 2 for the knocker arm shoulder and wrist.

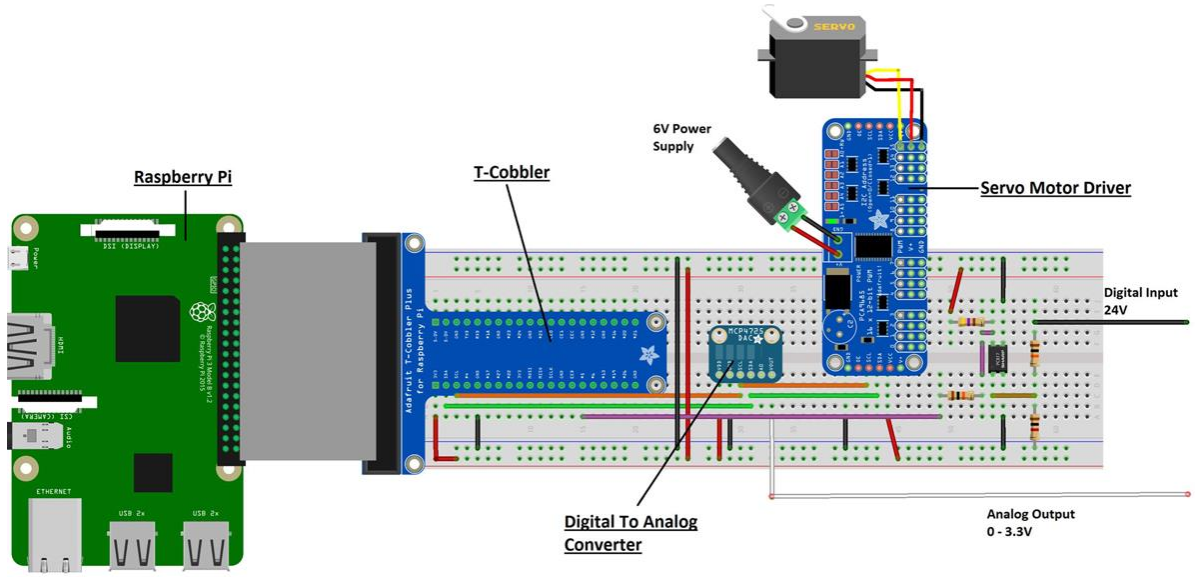

Fig. 5. Implementation of the electrical control hardware

An I^2C controlled digital to analog converted is used to send an analog signal between 0 and 3.3 Volts to the Robotino navigation program. This signal is modulated to represent the numerical addresses of the offices the mail robot should travel to, based on user web requests. Additionally, a digital acknowledge-arrival signal from the Robotino navigation program is converted from 24 to 3 Volts using an optocoupler as shown in Fig. 6. When the 24 Volt signal from the Robotino is high, it pulls the Raspberry Pi pin to ground through the optocoupler output side. Otherwise, the pin is high at 3 Volts.

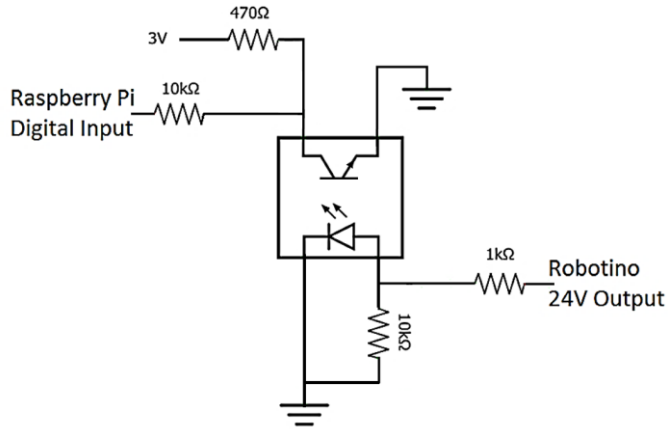

Fig. 6. Optocoupler use for relaying a Robotino digital signal to the Raspberry Pi

Software Design

Three types of software programs were implemented to work in unison to achieve the mail robot's functionality. Firstly, Robotino View[24], which is a graphical programming environment for the Robotino allowing control over the motors and sensor readings. Secondly, a Hypertext Processing (PHP[25]) website, hosted on the WiFi connected Raspberry Pi. The main page of this interface is depicted in Fig. 7. It allows the user to request the robot from an office PC, open & close mailbox doors, and issue a delivery command from the mobile tablet touch screen. Lastly, Python programs that manage user requests and robot action sequences. Both the PHP and Python code run on the Raspberry Pi board and they communicate via reading and writing text files.

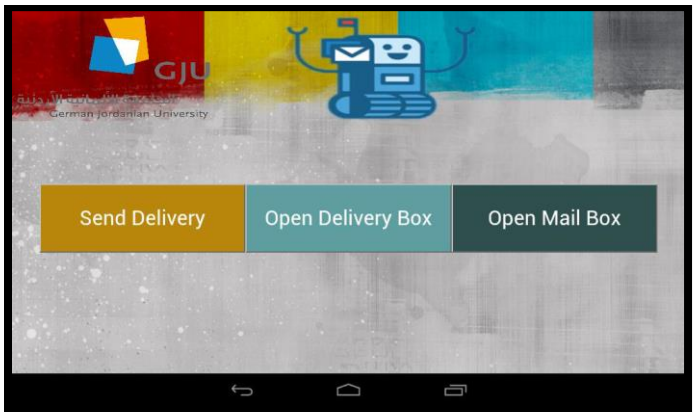

Fig. 7. Main PHP page of the user web-interface on tablet and PC

The user interface webpages unlock specific options to the user according to the state diagram shown in Fig. 8. If a user requests the mail robot to their office, a "Send Request" option is available of the office PC. Once the request is made, the interface updates a text file that is read by the Phython control program, which signals the user's address to the Robotino to travel to the user's office door and knock on it. The user then has the option to open the delivery box door and insert their mail. The door can then be closed, which returns the web interface to the main page, allowing the user to place a new send request to a colleague's office. Alternatively, the delivery box, or mail box doors can be opened to retrieve the mail.

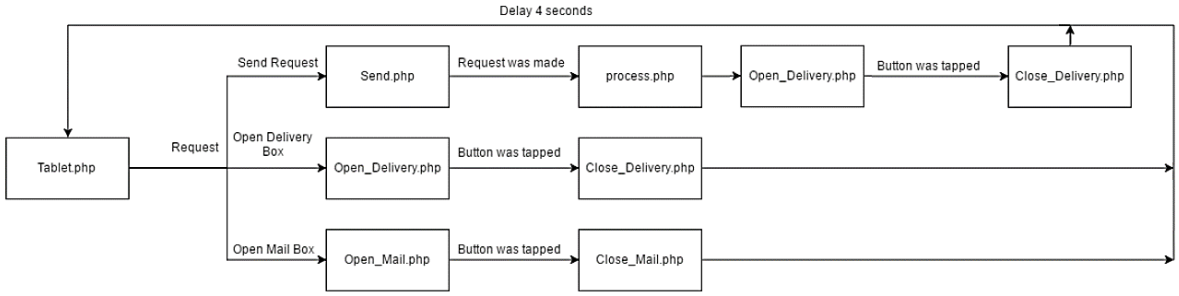

Fig. 8. PHP user web interface state diagram

The Raspberry Pi Python code communicates with the website's PHP code via text files and executes a parallel control flow shown in Fig. 9. The request buffer saves requests made while the robot is in the process of completing an earlier request. The address of the office door to head towards is encoded in an analog voltage signal sent to the Robotino navigation program as explained earlier. The servo motors are actuated when the robot reaches the desired office door and the Robotino sends a digital acknowledgement signal to the Raspberry Pi controller board through the optocoupler circuit depicted in Fig. 6.

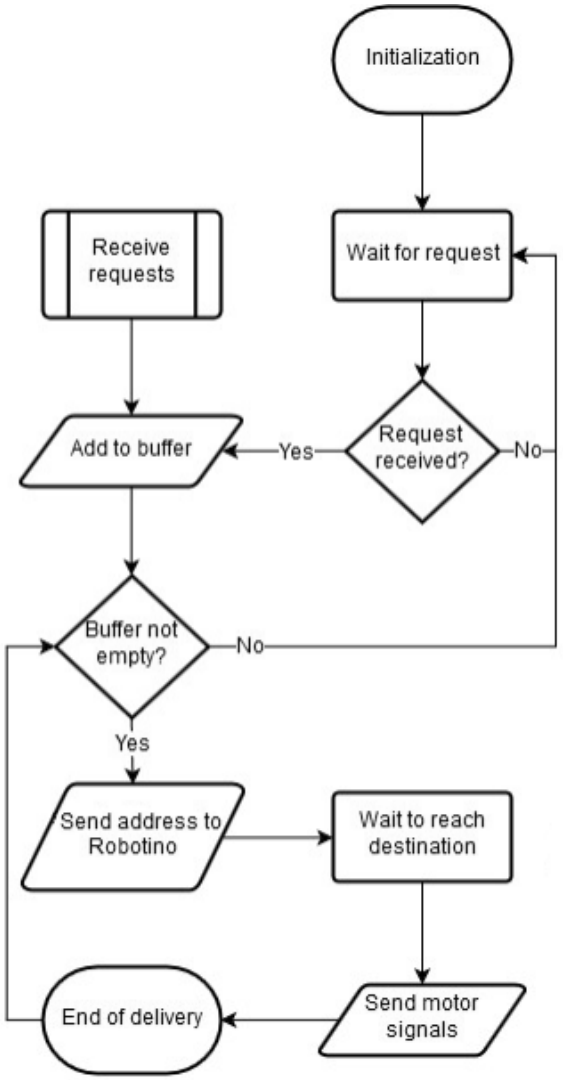

Fig. 9. Mail robot Python software control flow

The Python functions and their interoperability are depicted in Fig 10. There are two main programs, four communication text files, three client, i.e., user information definition functions, and two log print functions. The most important functions and files are explained in more details here.

Main_Program.py is where all the processes are triggered and decision making is done. It includes the code for web hosting, acquiring inputs, sending signals to servo motors, and updating all the middleware text files. I2C_Program.py executes all I^2C bus related processes. It receives input signals from Main_Program.py though the I2C_MW.txt file. It is responsible for controlling all servo motors and the digital to analog conversion board. This program is separated from Main_Program.py because it has delays within code lines that are necessary for servo motor control, which can affect the resposnsiveness of Main_Program.py since users will not be able to pass on requests while the program is delayed. Clients_List.txt is a text file containing clients' names and the number of their saved mails. This file is used in the initialization of Main_Program.py and is continuously updated. It is also used by Print_ClientsList.py. I2C_MW.txt is a text file that saves data (servo motor commands) from Main_Program.py. The data is continuously read by the I2C_Program.py program.

History.txt is a text file with records of all the events that occurred during the run of Main_Program.py. Records are added right after the event occurs in a time-independent manner. RequestsBuffer.txt is a text file conatianing the names of clients who requested the robot. It is called by Main_Program.py during initialization and is updated in every loop. This file is useful in case of a sudden shutdown, maintaining the requests list and enabling the program to continue from where is was interrupted.

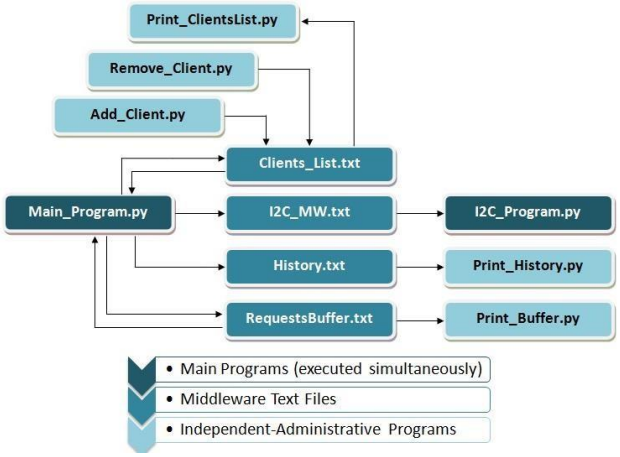

Fig. 10. Python control functions interfacing

Finally, the line following navigation program is implemented the graphical Robotino View software, which is triggered by the analog voltage from the Raspberry Pi that encodes the destination addresses, i.e., office door tags in a voltage between 0 and 3.3 volts. Once the destination is reached, the robot sends a digital acknowledgement signal to the Raspberry Pi.

Results

The implemented mail robot's functionality was tested between three offices. A black line was created on the floor for the robot to follow. The maximum speed of the robot is 10 km/m (2.8 m/s), but the test was performed at about 1 m/s. The robot waits for user requests at a defined home position, where a charging station could be located.

User requests are entered via the web interface PHP website. The robot responds to a request by driving to the user's office door and knocking on it as shown in Fig. 11. Fig. 12 shows the user opening the delivery box door from the top-mounted touch screen tablet computer running the web interface. The user issued a send request to a colleague's office after inserting the mail. The box opens and closes via touch screen commands. This sequence of operations can be seen in the control program's log file shown in Fig. 13.

The robot's functionality was tested between three offices, in addition to the home position. The robot was programmed to know these four locations. The tested features included moving the mail from the delivery box, when the recipient does not answer the door, to the mail box compartment. The PHP website operation was tested as well. The user inputs include opening and closing the delivery and mailbox doors, as well as sending the robot to deliver mail to one of the three defined offices.

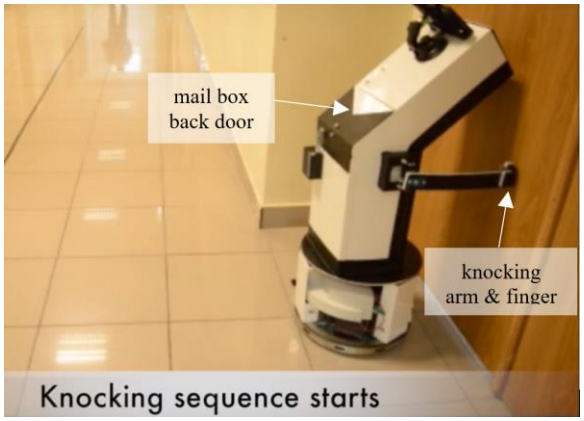

Fig. 11. The mail robot arriving at, and knocking on the user's office door

Fig. 12. The user opens the dellivery box door from the touch screen web interface

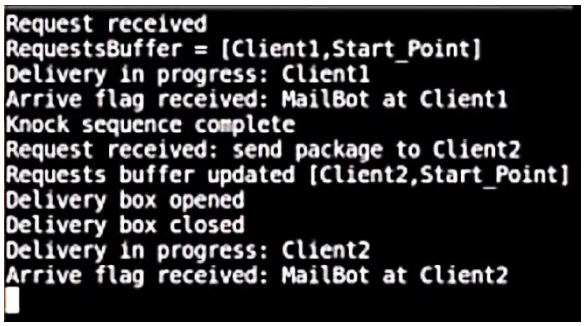

Fig. 13. Example execution sequence of mail dellivery from client1 to client2

Conclusions

This work detailed the mechanical, electrical, and software design and development for an indoor office mail delivery robot based on the Festo robotino platform. The platform is a popular development platform for student projects, mobile robot competitions, and research problems. It features an array of sensors, omni directional wheels, wireless and wired connectivity, a simulation environment, and numerous programming and interfacing options with popular softwares. The developed mail robot was successfully tested for mail delivery between three offices, while commanded by the user through a custom website viewable on office PC's and the robot's touch screen tablet.

Acknowledgements

This work was supported by an equipment and mobility fund by the Deanship of Scientific Research at the German Jordanian University.

Biography

1. Rawashdeh NA, Jasim HT. Multi-sensor input path planning for an autonomous ground vehicle. In: *2013 9th International Symposium on Mechatronics and Its Applications, ISMA 2013*. ; 2013. doi:10.1109/ISMA.2013.6547399
2. Rawashdeh NA, Alkurdi LM, Jasim HT. Development of a low cost differential drive intelligent ground vehicle. In: *2012 8th International Symposium on Mechatronics and Its Applications, ISMA 2012*. ; 2012. doi:10.1109/ISMA.2012.6215184
3. AlHamouz SO. Controlling A Robot in Unreachable Places with GPS and Ultrasonic Sensors. *GSTF J Comput*. 2018;5(3). http://dl6.globalstf.org/index.php/joc/article/view/1224. Accessed January 30, 2019.
4. Oltean SE, Dulau M, Puskas R. Position control of Robotino mobile robot using fuzzy logic. In: *2010 IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR)*. IEEE; 2010:1-6. doi:10.1109/AQTR.2010.5520855
5. Aladem M, Rawashdeh S, Rawashdeh N. *Evaluation of a Stereo Visual Odometry Algorithm for Passenger Vehicle Navigation*. Vol 2017-March.; 2017. doi:10.4271/2017-01-0046
6. Rawashdeh NA, Rawashdeh SA. Effect of Structural Scene Content on Feature-Based Visual Odometry Performance. *SAE Tech Pap*. 2018;2018-April. doi:10.4271/2018-01-0610
7. Rawashdeh NA, Haddad RM, Jadallah OA, To'Ma AE. A person-following robotic cart controlled via a smartphone application: Design and evaluation. In: *Proceedings - 2017 International Conference on Research and Education in Mechatronics, REM 2017*. ; 2017. doi:10.1109/REM.2017.8075245
8. Norell J. Prototyping an automated robotic shopping cart with visual perception. Bachelor Thesis, Luleå University of Technology, DiVA portal, 2018. http://www.diva-portal.org/smash/record.jsf?pid=diva2%3A1191653&dswid=-5985., Accessed January 30, 2019.
9. Robotino® – For research and education: Robotino® - Learning Systems - Festo Didactic. https://www.festo-didactic.com/int-en/learning-systems/education-and-research-robots-robotino/robotino-for-research-and-education-premium-edition-and-basic-edition.htm?fbid=aW50LmVuLjU1Ny4xNy4xOC44NTguODAyNQ. Accessed January 30, 2019.
10. M. Bliesener, Ch. Weber, K. Kling, U. Karras, D. Zitzmann CB. *Workbook Robotino® - Learning Systems - Festo Didactic*. Festo Didactic GmbH & Co. KG. - Germany; 2013. https://www.festo-didactic.com/int-en/learning-systems/workbook-robotino.htm?fbid=aW50LmVuLjU1Ny4xNy4xOC4xMTcyLjgwMjY. Accessed January 30, 2019.
11. Wielandt H. Press Release: ¡Hola! Mexico: RoboCup 2012 calls for full commitment. Festo Media Service. https://www.festo.com/net/sv_se/SupportPortal/Details/247596/PressArticle.aspx?show=image. Published 2012. Accessed January 30, 2019.
12. Ak A, Topuz V, Altıkardeş A, Oral B. Development of a Remote Laboratory Infrastructure and LMS for Mechatronics Distance Education. *Eurasia J Math Sci Technol Educ*. 2018;14(6):2493-2508. doi:10.29333/ejmste/89947
13. Chenatti S, Previato G, Cano G, et al. Deep Reinforcement Learning in Robotics Logistic Task Coordination. In: *2018 Latin American Robotic Symposium, 2018 Brazilian Symposium on Robotics (SBR) and 2018 Workshop on Robotics in Education (WRE)*. IEEE; 2018:326-332. doi:10.1109/LARS/SBR/WRE.2018.00066
14. Katona J, Ujbanyi T, Sziladi G, Kovari A. Speed control of Festo Robotino mobile robot using NeuroSky MindWave EEG headset based brain-computer interface. In: *2016 7th IEEE International Conference on Cognitive Infocommunications (CogInfoCom)*. IEEE; 2016:000251-000256. doi:10.1109/CogInfoCom.2016.7804557
15. Nan X, Xiaowen X. Robot experiment simulation and design based on Festo Robotino. In: *2011 IEEE 3rd International Conference on Communication Software and Networks*. IEEE; 2011:160-162. doi:10.1109/ICCSN.2011.6013566
16. Abdelwhab M, Abouelsoud AA, Elbab AMRF. Tackling Dead End Scenarios by Improving Follow Gap Method with Genetic Programming. In: *2018 57th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE)*. IEEE; 2018:1566-1571. doi:10.23919/SICE.2018.8492687
17. Datouo R, Motto FB, Zobo BE, Melingui A, Bensekrane I, Merzouki R. Optimal motion planning for minimizing energy consumption of wheeled mobile robots. In: *2017 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE; 2017:2179-2184. doi:10.1109/ROBIO.2017.8324742

18. Revelo C, Trujillo M, Rosales A, Pozo D. Indoor localization by using stereoscopic vision, odometry, and the Kalman Filter. In: *2014 IEEE ANDESCON*. IEEE; 2014:1-1. doi:10.1109/ANDESCON.2014.7098554
19. Mercorelli P, Voss T, Strassberger D, Sergiyenko O, Lindner L. A model predictive control in Robotino and its implementation using ROS system. In: *2016 International Conference on Electrical Systems for Aircraft, Railway, Ship Propulsion and Road Vehicles & International Transportation Electrification Conference (ESARS-ITEC)*. IEEE; 2016.
20. Tang Q, Eberhard P. *Relative Observation for Multi-Robot Collaborative Localization Based on Multi-Source Signals*. Volume 26, 2014 - Issue 4, pp 571-591, Taylor & Francis, 2014.
21. Engelhard P, Holfeld B, Schulz-Zander J, Oberle M. Software-Defined Networking in an Industrial Multi-Radio Access Technology Environment. In: *Proceedings of the Symposium on SDN Research - SOSR '18*. New York, New York, USA: ACM Press; 2018:1-2. doi:10.1145/3185467.3190789
22. Ismail ZH, Naim S, Ayob AF, Mahyuddin MN. Obstacles Avoidance Control for Autonomous Mobile Robot Based on Fuzzy Logic Controller. *Adv Sci Lett*. 2018;24(11):7895-7899. doi:10.1166/asl.2018.12451
23. Beregi R, Szaller Á, Kádár B. Synergy of multi-modelling for process control. *IFAC-PapersOnLine*. 2018;51(11):1023-1028. doi:10.1016/J.IFACOL.2018.08.473
24. Bliesener, M., Weber, C., Kling, K., Karras, U., & Zitzmann D. *Festo Robotino Manual*. Denkendorf, Germany: Festo Didactic GmbH & Co. KG.; 2007.
25. Thies C. Arntzen, Stig Bakken, Shane Caraveo, Andi Gutmans, Rasmus Lerdorf, Sam Ruby, Sascha Schumann, Zeev Suraski, Jim Winstead AZ. PHP: Hypertext Preprocessor. PHP Group. http://php.net/. Published 2001. Accessed February 2, 2019.