Hardware and Software Design for Redundant Robotic Manipulators

[1]Lyubomira Miteva and [2]Kaloyan Yovchev

Sofia University, 5 James Bourchier Blvd., 1164, Sofia, Bulgaria

{[1]llmiteva, [2]k.yovchev}@fmi.uni-sofia.bg

Abstract. Robots that have the minimum required degrees of freedom to accomplish a given task are defined as non-redundant robots. When the robots operate with humans or with other robots, they should be able to reach an arbitrary end-effector pose in the whole workspace. When the robots have more degrees of freedom than required to execute an assigned task they are define as redundant robots. Therefore, robots with redundant configurations are more flexible and have a wider applicability. This paper investigates a planar redundant robotic manipulator and its physical characteristics. The goal of this paper is to propose suitable cost-effective hardware and software design for the robot. The robot must be able to execute point to point or trajectory movements. Therefore, it is required to develop a control system that is able to solve the forward and the inverse kinematics problems. The hardware components have to be chosen according to the characteristics of the robot and software requirements. The discussed hardware and software design are validated through real experiments.

Keywords: redundant robot, inverse kinematics, hardware design, software design.

1 Introduction

Nowadays, the usage of robotics systems is rapidly increasing due to the aging of the working population. The robots can replace humans in simple repetitive tasks, in dangerous working environment or in tasks that require execution with high precision [1]. According to the International Federation of Robotics, the most robots are produced for the automotive and electronics industries [2]. Also, robotics systems are successfully integrated into the rehabilitation and training of children with specific needs [3].

The non-redundant robots have the minimum required degrees of freedom (DOF) for execution of their tasks. These robots have the following limitation. When the robotic manipulator has the minimum required number of joints to accomplish an assigned task, the manipulator cannot reach an arbitrary end-effector pose in the whole workspace. Therefore, the workspace, where an arbitrary pose can be reached, is limited. The robots that have more DOF as

needed to accomplish a given task are defined as redundant robots [4]. The redundant robots can increase the dexterity and overcome the aforementioned limitation [5]. Redundant robots offer better flexibility and avoid easily obstacles within their workspace. The redundant manipulators are able to reach a given end-effector position using different configurations of their mechanical structure. That is important for planning complex trajectory motions, such as avoiding obstacles, avoiding singularities, optimizing manipulability, minimizing required joint torques, etc. [6]. Also, the redundant configuration of the robot is applicable for complex industrial and non-industrial tasks such as industrial assembling, high speed object manipulation, domestic robotics, robotized surgery, etc. [7].

The robots should be able to operate right next to human workers or with other robots and to perform tasks autonomously without human supervision [8, 9]. When the robot cannot execute an assigned task, the human should be able to train (program) it easily and in an intuitive way [10]. Therefore, it is important that the robots can receive commands and return responses in every moment with a minimum delay or interruptions. The robots must have a reliable hardware and software control. Different types of hardware and software design and different control strategy are used for robot control [11].

The investigated in this paper robot is a 3D printed educational redundant robotic manipulator. The modern 3D printing technique is useful for the creation of cost-effective robot models or prototypes, that can be used in education or in industry. The 3D printing technology allows to design and creation of robotic parts with complex shapes. Because of that, for example, it can be chosen the best gripper form or the best link length for a given task [12].

The goal of this paper is to investigate the physical characteristics of an already built 3D printed redundant robot and to propose suitable cost-effective hardware and software design for the robot. The robot must be able to execute point to point or trajectory movements. Therefore, it is required to develop a control system, that is able to solve the forward and the inverse kinematics problems. The inverse kinematics problem is more computationally expensive than the forward kinematics. For a particular end-effector position and orientation of the redundant robot there might exist an infinite number of joint configurations or even no solutions at all [13]. In literature, different methods for solving the inverse kinematics problem are investigated, such as Cyclic Coordinate Descent [14], Pseudo Inverse Jacobian method [15], Genetic algorithms [16], etc. For the purposes of this paper the geometric solution of inverse kinematics proposed in [12] will be used. This approach is appropriate for planar robots with an open kinematic chain. Also, the hardware components of the robot have to be chosen accordingly to the characteristics of the manipulator and to the software requirements.

This paper is structured as follows. Section 2 describes the physical

characteristics of the redundant robot and the respective hardware and software requirements for execution of point to point and trajectory movements. Section 3 describes the hardware and software design for the redundant robot. Section 4 provides results from conducted experiments.

2 Problem Formulation

This section considers the 3D printed educational robot shown in Fig. 1.a. This robot has to execute point to point and trajectory movements. This requires the solution of the forward and inverse kinematics problems. Section 2.1 describes solutions of the forward and inverse kinematics problems. Sections 2.2 formulates the hardware and software requirements of the control system.

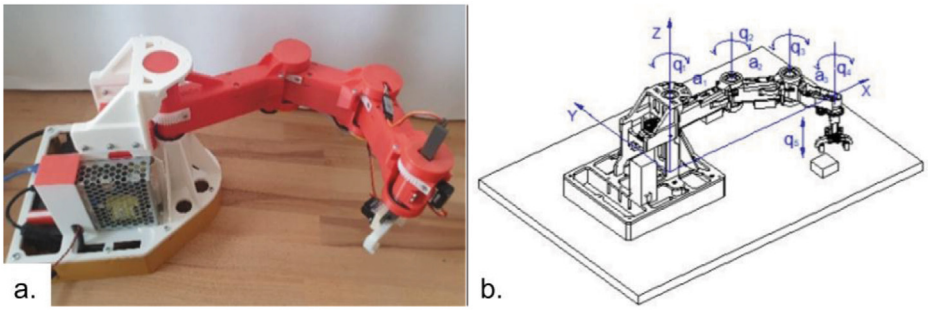

Fig. 1. a. 3D printed redundant robotic manipulator; **b.** Construction of the robotic manipulator.

2.1 Characteristics of the Redundant Robotic Manipulator

The investigated redundant robotic manipulator has four revolute joints, one prismatic joint and a total of five DOF (Fig. 1.b.). The revolute joints make possible infinite sets of joint configurations for the same position and orientation (pose) of the end-effector. Three DOF are enough to position and orientate the end-effector of the robot successfully when planar motion is considered. This robot has four revolute joints and is considered redundant with respect to planar motion.

Translational rail mechanism is attached to the fourth joint. There is a gripper attached to this rail mechanism. The gripper has three different anchor positions which can be used for the specific task. The lengths of the four links are measured as $a_1 = 150\ mm, a_2 = 110\ mm, a_3 = 100\ mm, a_4 = 0\ mm$. The rotational motion of the revolute joints is constrained. The constraints of the joint angles q_i are: $-\frac{\pi}{2} \leq q_i \leq \frac{\pi}{2}$, for $i = 1, 2, 3, 4$. The hardware components should take this constraint into consideration in order to avoid any damage of the robotic manipulator.

Solving Forward Kinematics.
For robotic systems with an open kinematic chain, the joint coordinates determine uniquely the position and orientation of the end-effector. The forward kinematics finds the position and orientation of the end-effector [13].

The fourth joint angle of the redundant robot changes only the end-effector orientation, but not its position. The revolute joint coordinates (q_1, q_2, q_3) determine the end-effector position in the XY-plane (Fig. 2). We are only interested in position of the end-effector, because the orientation can be set by changing the joint angle q_4.

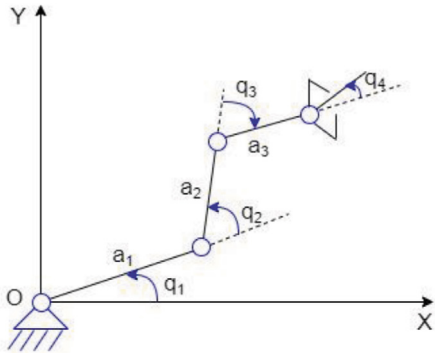

Fig. 2. Kinematics scheme of redundant robotic manipulator.

Let's denote the position of the end-effector in the workspace coordinates (x, y) and the joint angle coordinates in the joint space $\boldsymbol{q} = (q_1, q_2, q_3, q_4)$. The lengths of the links are respectively a_1, a_2, a_3, a_4. Since the robot is planar and the XY-plane is considered, the forward kinematics problem can be solved by trigonometry. Solution for the position (x, y) of the end-effector can be expressed as follows:

$$\begin{aligned} x &= a_1 \cos(q_1) + a_2 \cos(q_1 + q_2) + a_3 \cos(q_1 + q_2 + q_3) \\ y &= a_1 \sin(q_1) + a_2 \sin(q_1 + q_2) + a_3 \sin(q_1 + q_2 + q_3) \end{aligned} . \quad (1)$$

Solving Inverse Kinematics.
We want to find a set of joint configurations, that achieves a desired end-effector pose. For this purpose, it is required to solve the inverse kinematics problem [13]. For the purposes of this paper the geometric solution of inverse kinematics proposed in [12] will be used (Fig. 3).

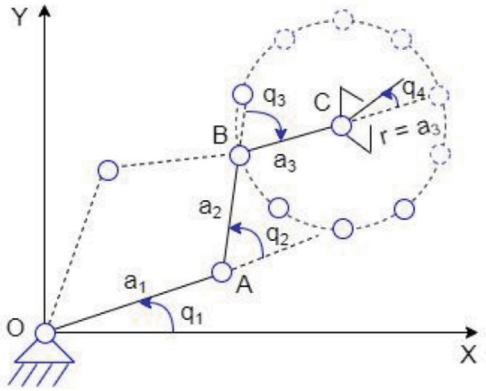

Fig. 3. Solution of the inverse kinematic problem of redundant robot manipulator.

Let's consider, that the end-effector has to be positioned at point C. Solution exists if the point C is at a distance $R \leq (a_1 + a_2 + a_3)$ relative to the center O. The length of link a_4 is zero. A set of points B is generated for the point C. The set of points B belongs to a circle with a center C and radius $r = \overline{CB} = a_3$. Our goal is to find a solution for each point from the set B for the inverse kinematics problem of two link mechanism consisting of the first two links. Due to the robot construction two solutions are possible. If at least one solution exists, it is checked, if the calculated joint angles q_i, for $i = 1, 2, 3$ are within the joint constraints. The joint angle q_4 changes only the orientation of the end-effector. The joint angles, that satisfy the joint constraints define the joint configurations, which can be executed by the robot. The hardware components and the software system must be able to implement and execute this solution.

2.2 Hardware and Software Requirements

The motion of all revolute joints is limited to 180 degrees by the construction of the robot. So, the selected motors and gearboxes for these joints should allow full 180 degrees motion. It is preferable that the motors have both position and velocity control.

The robot should be able to execute both single and a sequence of predefined movements. Therefore, a software needs to be developed, that allows the user to program movements and operate the robot. The software must provide user interface, that is intuitive and easy to use. The user must be able to track the current position of the robot and to send commands to the robot. The current position of the robot and the given commands must be either in joint space coordinates (q_1, q_2, q_3, q_4) or in workspace coordinates (x, y, Θ). Also, information about the position of the translational mechanism z and the gripper g should be provided.

The software must solve the forward and the inverse kinematics problems in order to achieve correct execution of the assigned commands. The user must be able to enable or disable the motors and send to the robot a command that sets the robot in home position. Also, the user must have the opportunity to monitor the communication with the robot.

The designed hardware and software systems should be easily expandable. Both wired and wireless communication methods are preferred. This will make the designed robotic system versatile and suitable for various educational, research and industrial tasks.

3 Hardware and Software Design

This section presents the selected hardware components and the design of the software control system. Cost-effective smart servo motors are selected for the actuators. The control electronics is based on the Arduino platform combined with ESP8266 Wi-Fi module.

3.1 Hardware Components

The Arduino Mega 2560 and Wi-Fi module WeMos D1 mini are chosen. It is required to control the revolute joints by position. Therefore, for the control of the revolute joints are used smart servo motors HerkuleX DRS-0101. For the control of the gripper and the translational mechanism are used the mini servo motors Feetech FT90M.

Actuator Motors.
The fourth revolute joints of the redundant robot are powered by smart servo motors HerkuleX DRS-0101. The translational mechanism and the gripper are powered by mini servo motors Feetech FT90M. The smart servo motors are controlled by UART serial communication and they can asynchronously return information about their current position and velocity.

These smart motors allow smooth control of their movement and they can hold their position. By using UART serial communication, we can easily change the rotational velocity, the current position and the operational status of up to 254 smart servo motors simultaneously. These motors are especially suitable for actuators of robotic manipulators. Their operating voltage is 7.4V DC [17].

DRS-0101 motors have an operating angle of 320 degrees, but the robot joints have joint constraints of 180 degrees. This requires gear reduction of 1:0.5625 in order to use the full range of motion of the servo motors. This gear reduction also ensures that the joint angle constraints cannot be violated. The usage of the 3D printing technology allows the design of custom gearbox with the required ratio.

The mini servo motors Feetech FT90M are not providing position feedback and

they cannot be controlled with velocity commands. They are managed by Pulse Width Modulation signals (PWM). PWM is a technique for managing analog and digital circuits. PWM uses a rectangular pulse wave, which pulse width is modulated resulting in the variation of the average value of the waveform [18]. The operating voltage of the motors is 4.8-6V DC.

Control Electronics.
The robot is controlled by an Arduino Mega 2560 and a Wi-Fi module WeMos D1 mini. The Arduino Mega 2560 can communicate with a computer, another Arduino, or another microcontroller and has four hardware serial UART ports. It can be programmed with the free and open source Arduino IDE through a USB connection without needing any extra hardware thanks to its pre burnt bootloader [19]. This makes it suitable for the control of the redundant robot since three serial UART ports are needed: one for the communication with the smart servo motors, one for the Wi-Fi module and one for the USB UART communication with the computer of the operator. This Arduino board can generate PWM signals for the mini servo motors of the translational mechanism and the gripper. ATmega2560 is capable to implement the solutions of both the inverse and the forward kinematics problems.

The Wi-Fi module allows the robot to receive commands through WebSocket communication and it can be used as an HTTP server for the developed web user interface. WeMos D1 Mini is a mini Wi-Fi device based on ESP8266EX chip [20]. This device can be reprogrammed via Wi-Fi connection. Additional logic-level converter is needed for connection of the Wi-Fi module to the Arduino due to the 3.3V operating voltage of the Wi-Fi module.

The fully assembled and mounted control electronics is shown in Fig. 4.a.

3.2 Communication Scheme

The described hardware components communicate as shown in Fig. 4.b. As already mentioned, three of the hardware UART ports of the Arduino microcontroller are used for communication with the DRS-0101 motors, with the Wi-Fi module and with the computer of the operator. These UART interfaces allow two-directional communication. The four DRS-0101 motors are connected to the same communication bus. The position of the two mini servo motors are controlled by the generated PWM signals from the Arduino board. This type of connection is single directional, and the motors cannot return any information about their current position.

The Wi-Fi module provides both WebSocket and HTTP servers. The WebSocket server acts as a proxy and forwards requests and responses from and to the Arduino board. The HTTP server serves the developed graphical web-based

interface. This interface uses the WebSocket connection to control the robot.

Also, the robot can be controlled directly with the same set of commands through the wired USB UART connection between the PC and the Arduino board.

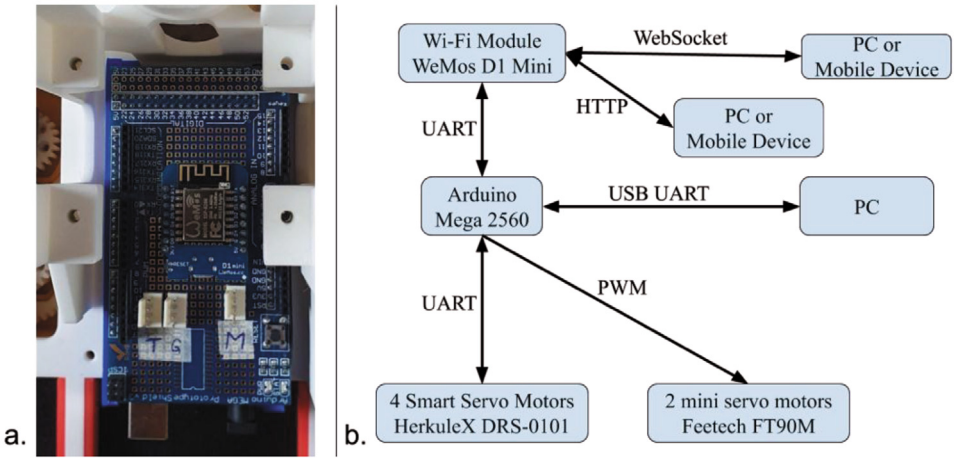

Fig. 4. a. Assembled control electronics; **b.** Communication of redundant robot manipulator.

3.3 Software Design

Software that satisfies the described requirements in Section 2.2 was developed. We have three main components: the Arduino board, the Wi-Fi module and the graphical interface for the end-user's operating device. For all of them specific software has to be implemented. Also, communication protocol consisting from text-based commands was developed. These commands start with the symbol 'C' and end with ';'. For example, the command "CTN;" enables the motors. Some of them return responses. The responses start with the symbol 'R'. For example, the command "CP;" returns in response the current positions of each motor. This protocol is processed by the Arduino board.

The Arduino board and the Wi-Fi module have restricted computational and memory resources. In order for the Arduino board to be able to find solution of the inverse kinematics problem the circle described in the inverse solution in Section 2.1 was linearly discretized to 360 points. Solving the inverse kinematics problem is computationally expensive and the approach in Section 2.1 is iterative. So, it is not executed for a constant time. This leads to an unpredictable computational time. When single position command in workspace coordinates is send to the robot the Arduino board can compute the solution in real time. However, if sequence of positions (trajectory) is sent to the robot, the Arduino board have to first solves the inverse problem for every trajectory point and only after that the robot is able to execute the preprogrammed trajectory in real time. The Arduino board has a

very limited memory and only 256 positions can be preprogrammed. However, the travel time from one position to the next can be set as a parameter. This allows different maximum execution time of the preprogrammed trajectories. For the low-level Arduino module only the open-source Arduino libraries for regular and smart servo controls are used. The libraries for the communication processing and solving of the kinematics problems were written in C++.

The Wi-Fi module is programmed to provide three services: Wi-Fi hotspot, HTTP server and WebSocket server. The HTTP server provides a graphical web-based user interface, that is accessible from any modern web browser after connecting to the Wi-Fi hotspot of the robot. For the Wi-Fi module the ESP8266 open source libraries for managing Wi-Fi hotspot and WebSocket and HTTP servers were used. The code for the communication processing is written as a C++ library.

For the graphical web-based interface pure HTML/CSS and JavaScript code is used with no additional libraries, because the WeMos board does not have enough memory. It only has 2 MB, because Over-The-Air update was enabled for easier development process. This user interface consists from several sections: navigation, information, main commands, program and console. Through the navigation section the user can access the following sections: main commands, program and console, which are described below (Fig. 5).

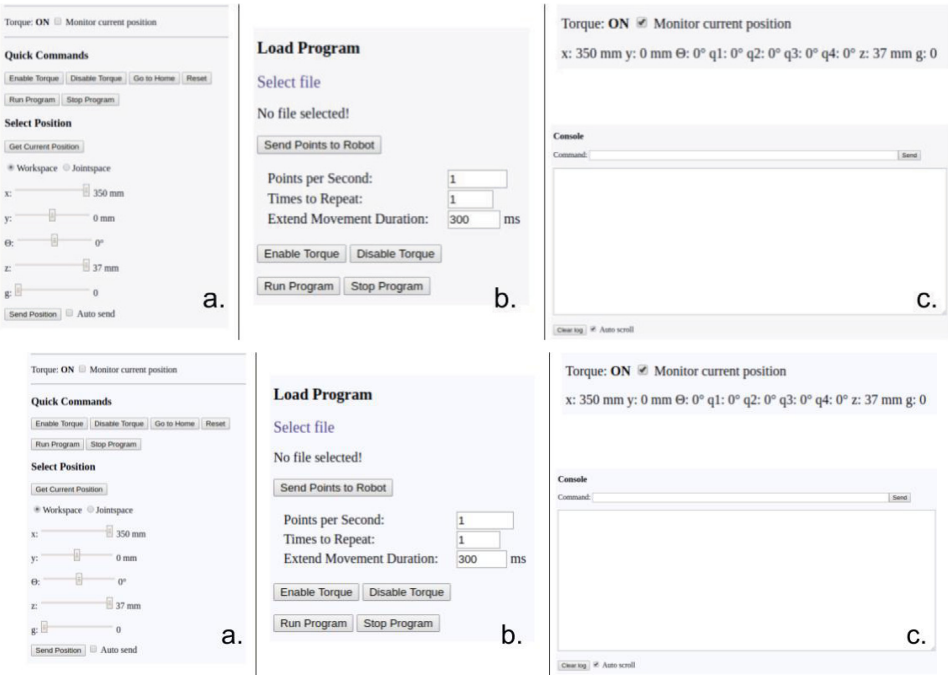

Fig. 5. User Interface: a. Quick Commands; b. Program Section; c. Status and Debug Console.

The information section provides status information and whether motors are enabled. When “Monitor current position” is checked, the user can see the current position of the end-effector. Also, the robot provides information about the four joint angles, the translation position z and about the gripper g. The displayed information about x and y coordinates is calculated locally from the user interface by solving the forward kinematics problem.

The main commands section is separated into two subsections: “Quick Commands” and “Select Position”. The “Quick Commands” subsection includes commands for enabling and disabling of the motors, go to home command (the robot will move to position with coordinates (350, 0)), reset command and run and stop program commands, that start and abort the execution of preprogrammed sequence of positions. The “Select Position” subsection includes commands for getting the current position of the robot. There are sliders, which indicate the current position of the robot and allow the user to control the redundant robot. The commands to the robot can be send either in workspace coordinates or in joint space coordinates. When the commands are in joint space coordinates, the forward kinematic problem is solved to find (x, y) coordinates of the desired end-effector pose. The checkbox “Auto send” allows the user to activate automatically sending of the commands from the sliders to the robot.

Up to 256 positions can be stored in the robot’s memory via the program section. These positions can be in workspace or in joint space coordinates. Through this section the user can determine how many positions will be executed for one second and how many times the stored sequence will be repeated. The user can also extend the movement duration. This allows smooth trajectory motion.

The console section allows direct sending of commands to the robot. Also, it displays the logged responses of the sent commands.

The WebSocket interface will also allow the building of a complex network of cooperating robots. As seen in Fig. 6 multiple robots can be controlled by a single control device. Also, there might be multiple control devices for a single robot.

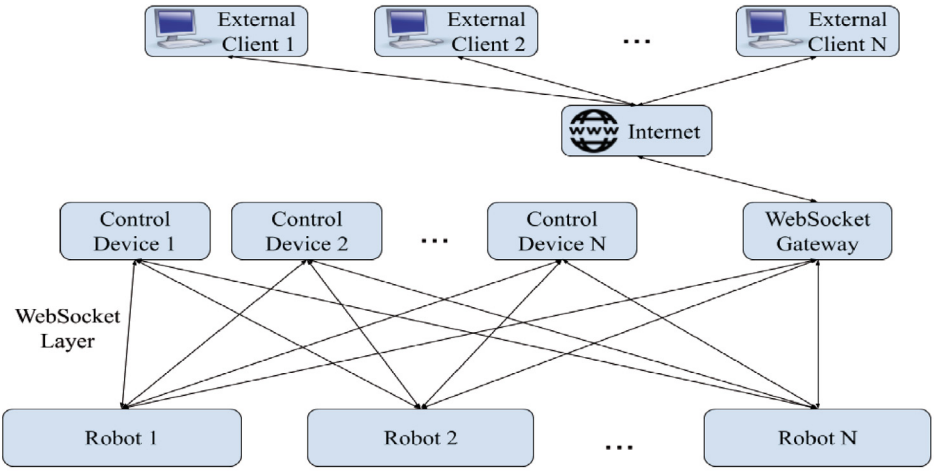

Fig. 6. Network of cooperating robots.

Furthermore, an additional WebSocket gateway node can be added to the network which can allow multiple external client devices to control and monitor multiple robots through Internet.

4 Experiments and Results

Two experiments were conducted with the described in this paper redundant robot and control system. They have the purpose to evaluate if the described hardware components, the communication scheme and the developed software are suitable for control of this redundant robot. The first experiment had to survey, if the redundant robot can position and orientate its end-effector in the desired pose. The user sends to the robot commands in joint space coordinates. The robot receives the given joint angles from the user correctly and solves the forward kinematics problem to find the desired end-effector position and orientation. The experiment was successful. The redundant robot positioned correctly its end-effector in the desired position. The measured delay of the communication through the Wi-Fi module was less than 10 msec. It was measured while there were multiple connected devices to the Wi-Fi hotspot provided by the WeMos board.

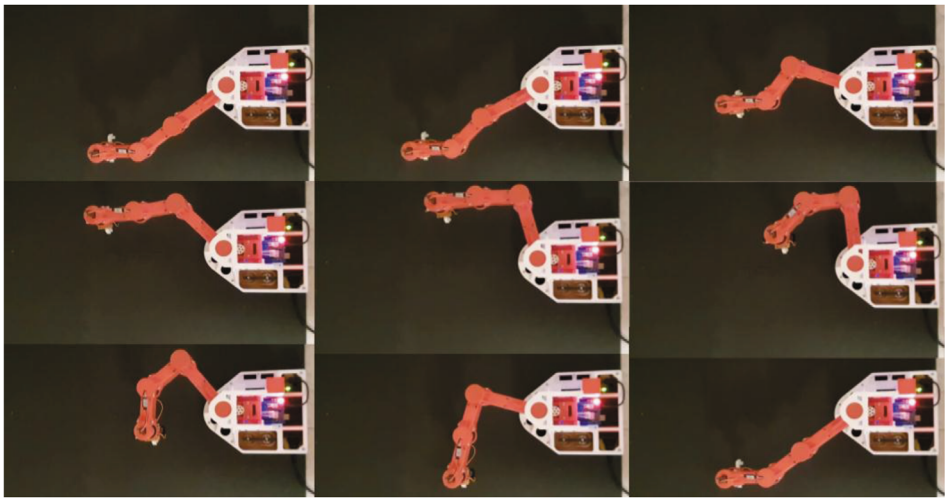

Fig. 7. Execution of trajectory movement.

The second experiment had the purpose to evaluate if the redundant robot can execute a trajectory movement provided as a sequence of positions (Fig. 7). The desired trajectory describes a rectangle within the workspace. The redundant robot receives the position commands as a workspace coordinate. Therefore, the inverse kinematics problem is solved for each position in order to find the desired joint angles. The found solutions were correct and the second experiment was successful. Also, the user was able to monitor in real time the current position of the robot from the web-based user interface.

Further investigations can be made with different strategies for solving the inverse kinematics problem of the redundant robot. The conducted experiments confirmed that the complete robotic system is suitable for trajectory movements, but the internal memory of the Arduino board is very limited. It will be useful if an external memory (SD card slot) is added to the control hardware.

5 Conclusion

This paper proposes a cost-effective hardware and software design for redundant robotic manipulators. The redundant robots are the best option for the environment with obstacles. Because of their versatility, they can do better work right next to humans than the non-redundant robots.

For designing an appropriate robot control, knowledge of the kinematic characteristics of the robot is required as well as solutions of the forward and the inverse kinematics problems. The robot is controlled by an Arduino Mega 2560 board and a WeMos D1 mini Wi-Fi module. These hardware components

are chosen, because they are cost-effective and easy to program and control. The selected actuators and gear ratio physically constrain the motion of each joint within its joint constraints. This eliminates the possibility of an unwanted damage due to incorrect control commands.

Software, that solves the forward and the inverse kinematics problems and allows the user to operate the robot has been developed. Experiments with the redundant robot are conducted in order to evaluate whether the described software and hardware components are suitable. Those experiments were successful. The robot receives the sent commands immediately without any delay and returns responses without delay. The redundant robot can position and orientate its end-effector in the desired pose and the robot can handle commands either in workspace coordinates or in joint space coordinates. So, the chosen hardware components are suitable for control of the robot.

The provided communication interface for the robot allows the control system to be open and expandable. This will allow the redundant robot to be used in further research about motion planning, manipulability optimization, obstacle and singularity avoidance, etc.

Acknowledgments

This research is supported by a scientific-research project No. N 17/60 "Investigation and modeling of new robots through non-traditional technologies and materials" under contract No. DN 17/10 with Bulgarian National Science Fund and by the Fund for Scientific Research at Sofia University "St. Kliment Ohridski" under grant 80-10-23/2020.

References

1. Okuma, T.: Editorial. International Federation of Robotics, https://ifr.org/downloads/press2018/Editorial_WR_2019_Industrial_Robots.pdf, last accessed 2020/03/10.
2. Executive Summary World Robotics 2019 Industrial Robots, International Federation of Robotics, https://ifr.org/downloads/press2018/Executive%20Summary%20WR%202019%20IndustriIn%20Robots.pdf, last accessed 2020/03/10.
3. Riener, R.: Control of Robots for Rehabilitation. In: The International Conference on "Computer as a Tool, pp. 33-36. Belgrade, (2005).
4. Žlajpah, L.: On Orientation Control of Functional Redundant Robots. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 2475-2482. Singapore, (2017).
5. Chiaverini S., Oriolo G., Maciejewski A.: Redundant Robots. In: Siciliano B., Khatib O. (eds) Springer Handbook of Robotics. Springer Handbooks. Springer, Cham, (2016).
6. Nemec, B., Zlajpah, L.: Force Control of Redundant Robots in Unstructured Environment. In: IEEE Transactions on Industrial Electronics, vol. 49, no. 1, pp. 233-240, (2002).
7. Potkonjak, V., Popovic, M., Lazarevic, M., Sinanovic, J.: Redundancy Problem in Writing: from Human to Anthropomorphic Robot Arm. In: IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), vol. 28, no. 6, pp. 790-805, (1998).

8. Peng, Y., Carabis, S., Wen, T.: Collaborative Manipulation with Multiple Dual-Arm Robots under Human Guidance. In: Int. J. Intell. Robot Appl. 2, pp. 252–266, (2018).
9. Abdelmalek, K.: Dexterity of Manipulator Arms at an Operating Point. Vol. 82. (2000).
10. Breazeal, C., Hoffman, G., Lockerd, A.: Teaching and Working with Robots as a Collaboration. In: Proceedings Third Int. Joint Conf. Autonom. Agents Multiagent Syst., vol. 3, pp. 1030-1037. (2004).
11. Saliyah, K., Riza, S., Satria, A., Nahdatul A., Mohammad A., Abu, H.: A Review of Wireless Technology Usage for Mobile Robot Controller. IPCSIT vol. 34 (2012).
12. Chavdarov, I., Nikolov, V., Naydenov, B., Boiadjiev, G.: Design and Control of an Educational Redundant 3D Printed Robot. In: International Conference on Software, Telecommunications and Computer Networks (SoftCOM), pp. 1-6. Split, Croatia (2019).
13. Kevin M., Lynch, Frank C.: Modern Robotics Mechanics, Planning and Control. Cambridge University Press (2017).
14. Martín, A., Barrientos, A., Cerro, J.: The Natural-CCD Algorithm, a Novel Method to Solve the Inverse Kinematics of Hyper-Redundant and Soft Robots. Soft Robotics, vol. 5. (2018).
15. Chen, D., Zhang, Y., Li, S.: Tracking Control of Robot Manipulators with Unknown Models: An Jacobian-Matrix-Adaption Method. In: IEEE Transactions on Industrial Informatics, vol. 14, no. 7, pp. 3044-3053. (2018).
16. Momani, S., Abo-Hammour, Z., Alsmadi, O.: Solution of Inverse Kinematics Problem using Genetic Algorithms. In: Applied Mathematics & Information Sciences, pp. 1-9. (2015).
17. Smart Servo Motors HerkuleX DRS-0101, https://www.robotshop.com/en/herkulex-drs-0101-robot-servo.html, last accessed 2020/03/10.
18. Homepage Feetech, http://www.feetechrc.com/, last accessed 2020/03/10.
19. Arduino Mega 2560, https://store.arduino.cc/arduino-mega-2560-rev3, last accessed 2020/03/10.
20.. Wemos D1 mini, https://docs.zerynth.com/latest/official/board.zerynth.wemos_d1_mini/docs/index.html, last accessed 2020/03/10.