DOI: 10.2478/ijasitels-2024-0018

A robotic arm for playing chess

Cristian Florian PAȚANGHEL[1], *Macarie BREAZU*[1]

[1]*Computer Science and Electrical and Electronics Engineering Department, Faculty of Engineering, "Lucian Blaga" University of Sibiu, Romania*
{cristian.patanghel, macarie.breazu} @ulbsibiu.ro

Abstract

The main goal of this research was to design and implement a robotic arm for playing chess on three standard real boards handling real regular pieces. Research, design and redesign were done on all aspects of the development: mechanical, hardware and software. A lot of problems emerged, but were overcome, and the robotic arm met its goal: to reliably moves chess pieces on the boards with millimetric positioning precision.

Interfacing the PC to 3 DGT autosensory boards and interfacing the PC with a classic software chess engine should be further considered.

Keywords: chess, robotic arm, SCARA robot, kinematics

1 Introduction

The main goal of this research was to design and implement a robotic arm for playing chess on three real boards handling real regular pieces. The three boards must be in front, to the left and to the right of the robotic arm. The size of the board must be the official one (field size 55x55mm) and the pieces should be any classical set without any modifications.

2 History of chess-playing robots

The fascination with chess-playing robots is not a recent phenomenon but dates back over two centuries. The notable starting point was the renowned 'The Turk', an intricate contraption that claimed to be a fully operational chess automaton. This chapter is adapted mainly from [1].

2.1 The Turk, the first (hoax)

The Turk, created in 1770 by Wolfgang von Kempelen, appeared to be a chess-playing automaton but was actually a clever illusion with a human chess master inside. Despite its deceptive nature, The Turk gained popularity. Kempelen initially enjoyed its success but later became reluctant to share it. The machine's interior was designed to mislead observers, and the operator remained hidden. Even so, The Turk was a remarkable feat of ingenuity.

After Kempelen's death, it was sold to Johann Nepomuk Mälzel, who restored and

Figure 1. Boris Handroid [1]

Figure 2. Novag Robot Adversary [2]

enhanced it. Although it was a hoax, customers still played against skilled chess masters. Notably, chess master William Schlumberger operated The Turk many years.

2.2 'Ajedrecista', the first true automaton

It took over 140 years for the first genuine chess automaton to emerge. In 1912, Spanish engineer Leonardo Torres y Quevedo created 'Ajedrecista,' widely recognized as the first computer chess machine and the first computer game in general. Using electromagnets beneath the board, Ajedrecista could autonomously play the endgame of rook and king against a lone king. Remarkably, it could play without external assistance and even detected illegal moves by the opponent, signalling them with a light. If three illegal moves were made, the automaton would cease playing.

2.3 Boris Handroid, the first commercial one

The Boris Handroid, a unique chess robot, became the first commercially available device of its kind. Introduced in 1980, it could only be obtained through special mail order. For a long time, the Boris Handroid remained a rare and elusive item, with collectors and enthusiasts questioning its existence until a single unit surfaced in the possession of Swiss owner Rolf Bühler. This remarkable device not only recognized (based on Hall sensor) the movements of human-controlled chess pieces but also executed its own moves using a robotic arm controlled by three servomotors. It offered seven game levels and featured the same Sargon 2.5 chess program as the MGS multi-game system.

2.4 Novag Robot Adversary, the first available in stores

In 1982, the iconic Novag Robot Adversary marked the introduction of chess robots in department stores and specialty shops. This legendary machine featured a sleek mechanical arm that gracefully folded and unfolded while delicately manipulating chess pieces with its pincers. It was truly a captivating sight. However, despite its remarkable design, the Novag Robot Adversary faced challenges with reliability, resulting in a relatively high failure rate. Consequently, only a limited production run of approximately 2000 units was achieved.

2.5 Milton Bradley "Grandmaster", the sliding one

Following shortly after, in 1983 Milton Bradley released its own rendition of a robot chess machine known as the Milton Bradley "Grandmaster". Instead of using a mechanical arm, this machine employed magnets beneath the chess pieces, allowing them to glide across the board during gameplay. The Grandmaster quickly gained popularity and paved the way for a series of machines utilizing this innovative technology from Milton Bradley. Fidelity, a company that later acquired the patent, introduced their own version called the Fidelity Phantom in 1988. The Phantom, boasting a strong program developed by the Spracklens, resembled its predecessors and was highly regarded.

While these were not the last chess-playing robots, they were indeed the most significant and groundbreaking contributions to the field. Although dedicated chess computer manufacturers have largely faded away, today individuals still can create their own chess-playing machines.

2.6 The Raspberry Turk

The Raspberry Pi, an affordable and highly versatile single-board computer (SBC) developed in the United Kingdom, has become the go-to tool for DIY (Do-It-Yourself) electronics enthusiasts. It is often hailed as the "Swiss army knife" for computer and electronic hobbyist projects due to its low cost, modular design, and open nature, along with its support for HDMI and USB standards.

Unsurprisingly, someone has taken advantage of the Raspberry Pi's capabilities to create detailed plans for building your own chess-playing robot. Joey Meyer, the creator of “The Raspberry Turk”, has dedicated a website ([3]) to guide individuals through the process, providing open-source instructions.

2.7 Use of industrial robots

With the development of industrial robots ([4]), various demonstration meetings were organized between robots and top grandmasters, like the ones with Vladimir Kramnik and Alexander Grischuk.

The (industrial) chess playing robots become very popular (typical using 3 chess boards, like in our design), leading also to a well-known accident, when a 7-year-old child finger was broken by the robot (considered mainly a human error, [5]).

3 Kinematics in SCARA robots

3.1 SCARA robots

SCARA is an industrial robot type initially developed by Prof. Makino at Yamamachi University in Japan, starting with 1978. According to its designer ([6]) “*The name SCARA stands for Selective Compliance Assembly Robot Arm, where “Selective Compliance” means that the robot’s compliance differs selectively with*

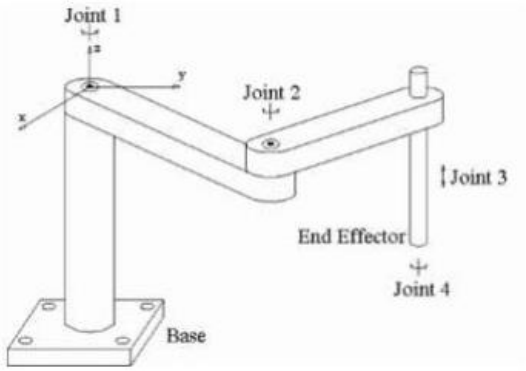

Figure 3. Simplified SCARA robot [8]

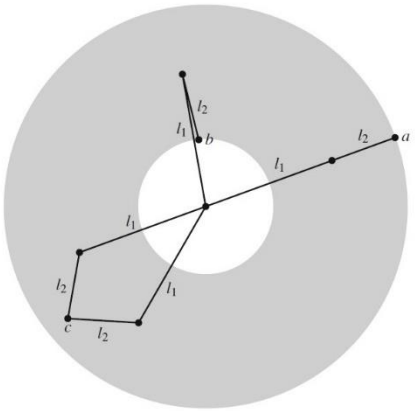

Figure 4. Range covered [9]

its direction". Today, some authors replace "Assembly" with "Articulated" in the definition ([7]). Its motion capabilities primarily consist of rotations within the horizontal plane, along with translation within that plane, while the gripper axis operates vertically.

We explore (adapting from [9]) the kinematics of a simplified SCARA robotic arm operating in a two-dimensional space (Fig. 3). This arm consists of two links, two rotational joints (Joint 1 and Joint 2), and an end effector (such as a gripper, welder, etc.).

The grey area displayed in Fig. 4 represents visually the workspace of the arm, illustrating the range of positions achievable by the end effector. Assuming $l_2<l_1$ (as in practical cases), the workspace exhibits circular symmetry, allowing for (theoretical) unrestricted joint rotations ranging from -180° to 180°. The outer circle corresponds to the farthest limit, l_1+l_2. Conversely, the inner circle represents the nearest limit, l_1-l_2. The other reachable positions, labelled as c, can be obtained by two different joint rotation configurations that position the arm at that point.

3.2 Forward Kinematics

In Fig. 5 we consider the first joint as the origin (0, 0) of the coordinate system. The lengths of the two links are l_1 and l_2, respectively. The first joint is rotated by an angle α, and the second one by an angle β (relative to the first joint).

Based on the l_1, l_2, α, β parameters, the position of the end effector results easily as:

$$x = x' + x'' = l_1 \cos(\alpha) + l_2 \cos(\alpha + \beta) \quad (1)$$

$$y = y' + y'' = l_1 \sin(\alpha) + l_2 \sin(\alpha + \beta) \quad (2)$$

In Fig. 5 the value of β is negative (corresponding to a clockwise rotation).

3.3 Inverse kinematics

In this case we must solve the inverse problem: given the desired coordinates (x, y) and the values for l_1 and l_2 what are the values for α and β to reach the desired position.

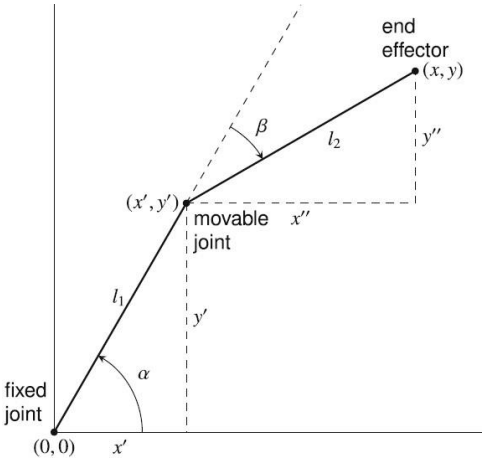

Figure 5. Forward kinematics [9]

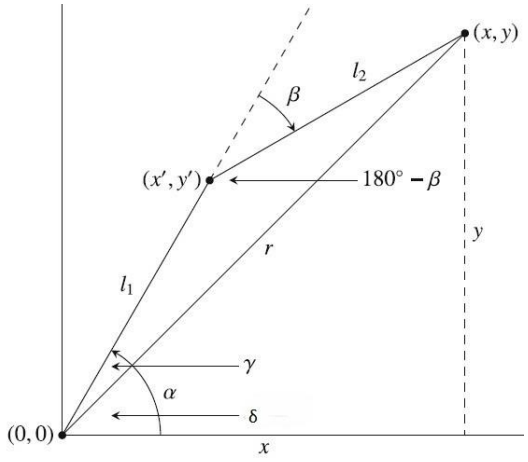

Figure 6. Inverse kinematics [9]

As discussed before, the inverse problem can have no solution (outside the grey area), a single solution (on the borders of the grey area) and two solutions (inside the grey area). When we have two possible solutions, we will select the one so that the arm avoids negative y coordinates as much as possible.

In Fig. 6, by applying the law of cosines in the triangle l_1, l_2, r, we get

$$l_1{}^2 + l_2{}^2 - 2l_1l_2\cos(180° - \beta) = r^2 \tag{1}$$

which can be rearranged to solve for β

$$\cos(180° - \beta) = \frac{l_1{}^2+l_2{}^2-r^2}{2l_1l_2} \tag{2}$$

$$\beta = \pm\left(180° - \operatorname{acos}\left(\frac{l_1{}^2+l_2{}^2-r^2}{2l_1l_2}\right)\right) \tag{3}$$

To obtain γ, and subsequently α, we utilize the law of cosines again, this time considering γ as the central angle

$$\cos\gamma = \frac{l_1{}^2+r^2-l_2{}^2}{2l_1r} \tag{4}$$

$$\gamma = \operatorname{acos}\left(\frac{l_1{}^2+r^2-l_2{}^2}{2l_1r}\right) \tag{5}$$

Also, we have

$$\tan(\delta) = \frac{y}{x} \tag{6}$$

So, we get

$$\delta = \operatorname{atan}\left(\frac{y}{x}\right) \pm \gamma \tag{7}$$

thus

$$\alpha = \operatorname{atan}\left(\frac{y}{x}\right) \pm \operatorname{acos}\left(\frac{l_1{}^2+r^2-l_2{}^2}{2l_1r}\right) \tag{8}$$

To restrict, as much as possible, the arm from moving to negative y coordinates (where a wall could be), we choose a specific solution from the two that are available, according to the quadrants we must reach. For quadrants I and IV (the

right ones) we add the γ in (7) and consider negative values for β in (3). For quadrants II and III (the left ones) we subtract γ in (7) and consider positive values for β in (3).

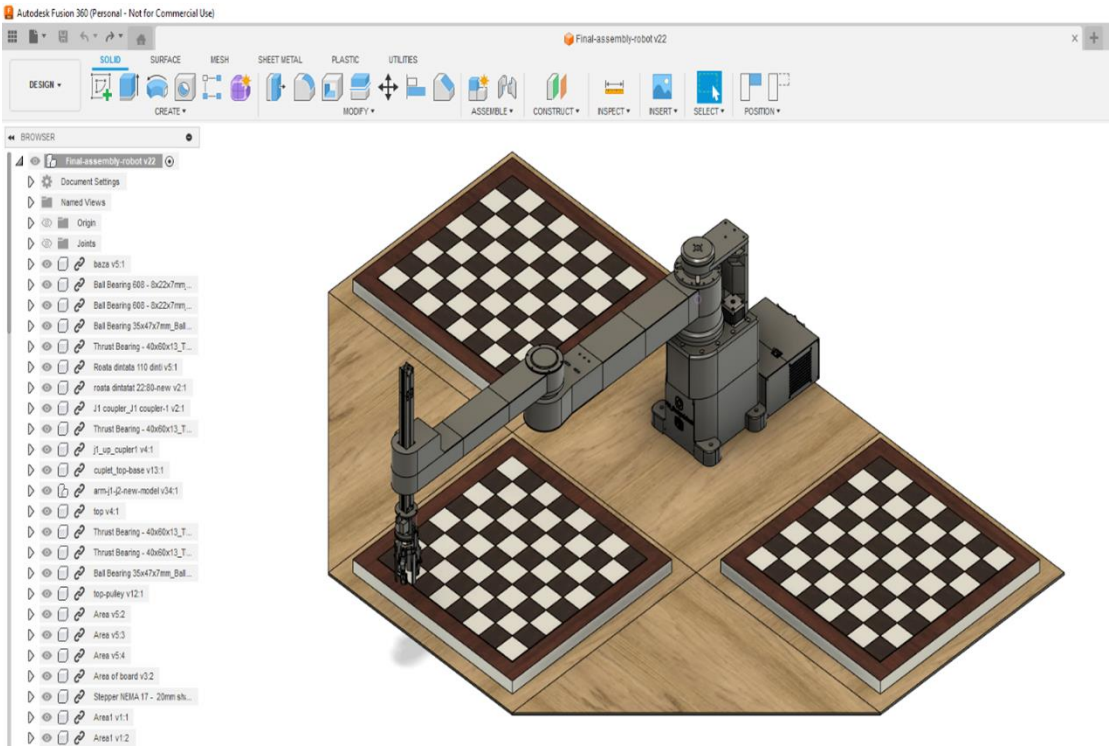

Figure 7. View of assembled robot in Fusion 360

4 Design of robotic arm

4.1 Mechanical design

The arm was designed in compliance with the board requirements: board size of 520x520mm with field size 55x55mm (as for the autosensory DGT Smart board [11]). The robot was designed to have one board in front, one on left and one on right, thus being able to play chess with three human players simultaneously. The mechanical part of the robot (Fig. 7), a classic SCARA one, was developed using Fusion 360 ([11]) powered by Autodesk.

The robot has a total length of 129 cm and is fully 3D printed. We chose the 3D solution because it is a low-cost and accurate solution. The model is split in more pieces to fit for printing using a small 3D printer (up to 22x22x25 cm). Total amount of time to print the robot was around 300 hours.

The motor responsible for rotating joint 1 is situated within the base. By utilizing two gears a ratio of 1:22 can be achieved. These gears are driven by the motor via toothed belts. To enhance the resistance of the plastic arm, a clamp-type coupling is used at the base, meaning that arm 1 is secured to the base at both top and bottom.

To minimize the weight of the arm, the motor responsible for rotating joint 2 is positioned at the centre of joint 1. Two gears are utilized to achieve a combined ratio

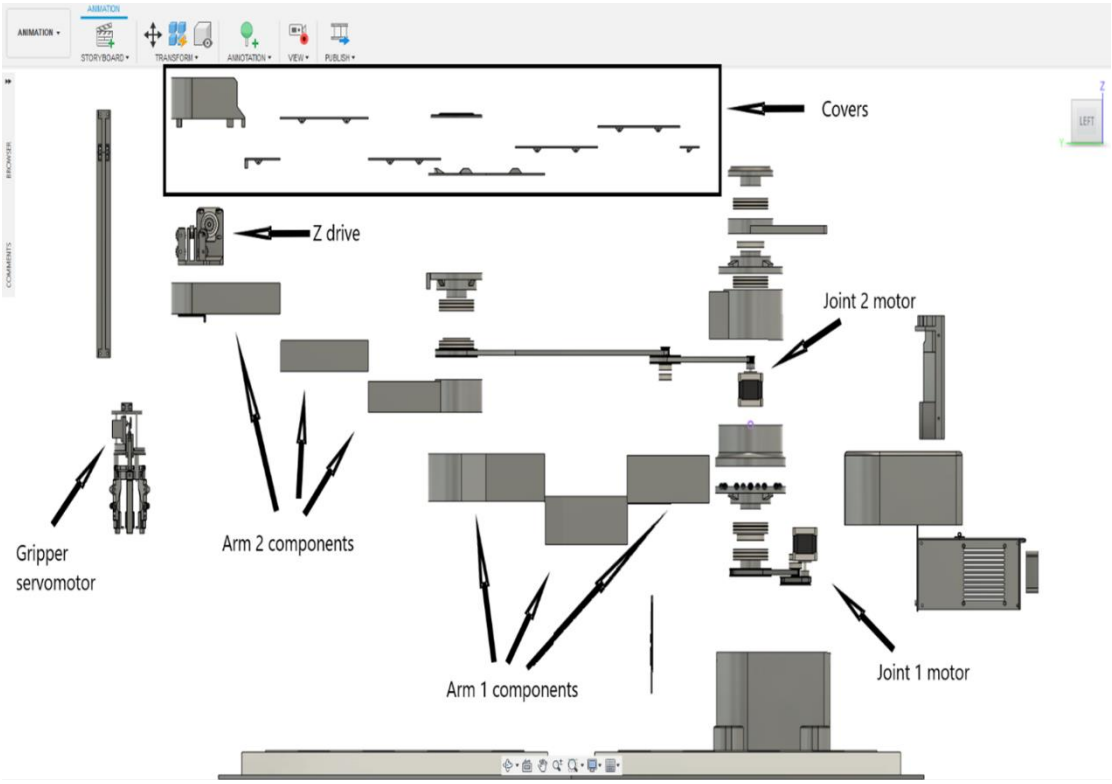

Figure 8. Components of the robotic arm

of 1:14.6. Additionally, two toothed belts are employed to drive these gears and rotate the wheels.

Joint 3 is specifically designed for the linear movement on the Z axis. It features a 1:1 ratio and consists of four gears for driving the axis and bearings for ensuring smooth linear movement. The Z-axis has a total length of 30 cm, but only 20 cm are usable.

At the base of the Z-axis the gripper is positioned, specially designed to securely hold all chess pieces. In the centre of the gripper a camera is strategically placed, enabling future projects involving automatic corrections based on visual location of

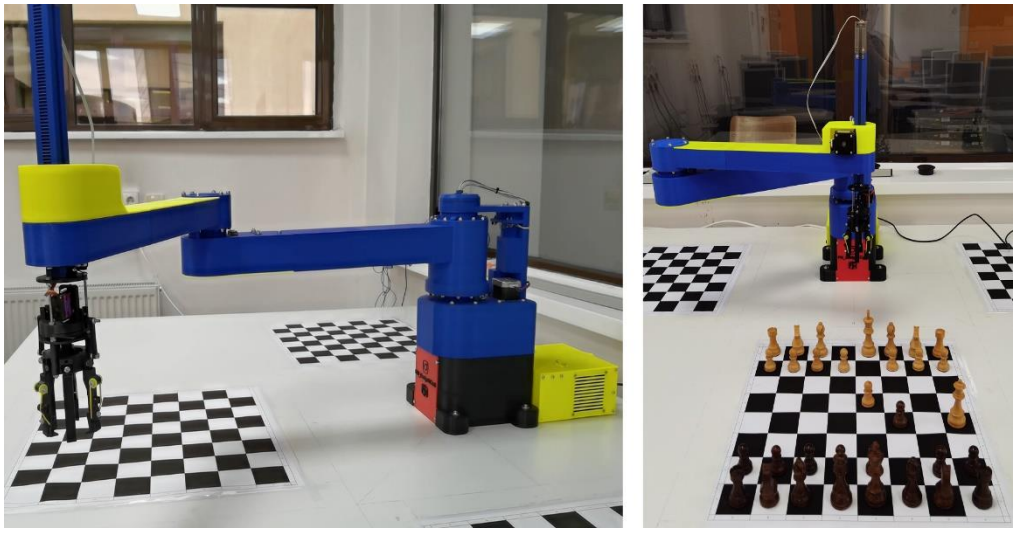

Figure 9. The final stage of the robotic arm

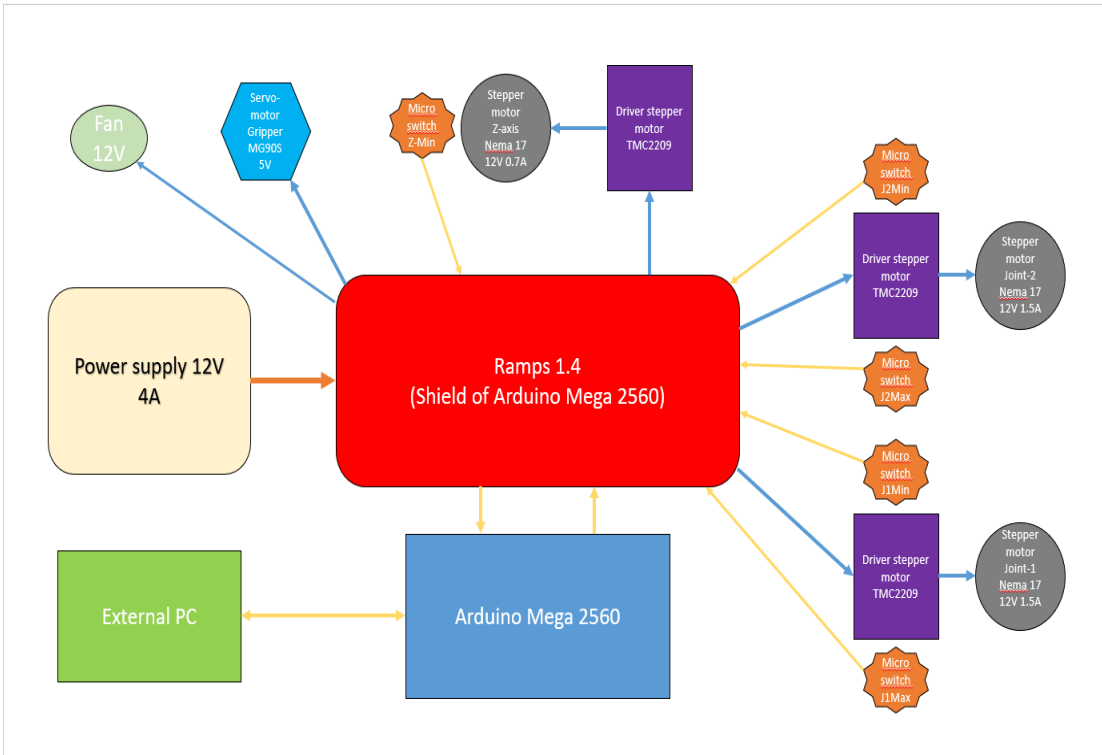

Figure 10. Block diagram of the hardware part of the robot

pieces on the board. To control the opening and closing of the gripper a servomotor is utilized, providing precise control over its movements.

The robot has a maximum operational radius of 95 cm (l_1=50 cm, l_2=45 cm) and a minimum radius of 20 cm (because of the limiters). Joint 1 is equipped with two position limiters, enabling the arm to rotate up to 272 degrees. Similarly, at the base of joint 2 there are also two position limiters that allow the arm to rotate up to 312 degrees. The robot was specially designed with internal wire integration in mind.

4.2 Hardware design

The board used as the controller of this robot is the Arduino Mega 2560 ([12]). With an impressive array of features, it offers 54 digital input/output pins, among which 15 can function as PWM outputs. Additionally, it boasts 16 analog inputs, 4 UARTs, a 16 MHz crystal oscillator and a USB connection. We use it in combination with a Ramps 1.4 shield ([13]) than offers precise control over up to five stepper motors, allowing for 1/16 stepping precision. Moreover, it seamlessly interfaces with various components, being used by RepRap enthusiasts to build and customize their 3D printing machines with ease. The choice of this board and shield was made considering also future developments of this project.

The block diagram reveals that the robot controller operates under the command of an external computer, establishing a connection with the Arduino board. To operate the robot only three stepper drivers are required, one each for Joint 1, Joint 2, and Joint 3. Additionally, five position limiters were used, the microswitches defining the range of motion. The gripper function is controlled by a servo motor, providing precise control over its movements.

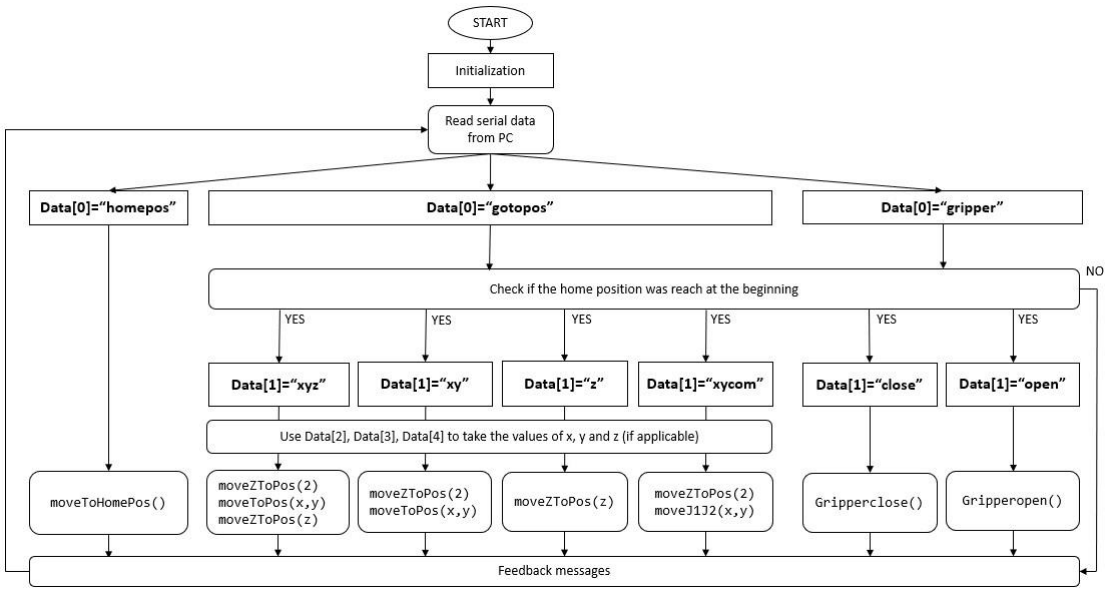

Figure 11. Flowchart of the source code (simplified)

For Joint 1 and Joint 2, Nema 17 motors (12V,1.5A) were selected. These motors belong to the category of stepper motors and provide 200 steps per rotation. To avoid excessive strain on the end effector of the arm, a smaller Nema 17 motor (12V,0.7A) was chosen for joint 3. This motor also offers 200 steps per rotation.

To effectively control these motors TCM2209 bipolar stepper motor drivers were employed. This type of driver is renowned for its silent and precise operation. It enhances the motor's performance by multiplying the number of steps by 8 (microsteps), resulting in an increased precision during movement and positioning.

For the gripper mechanism a 5V MG90S servomotor was employed. This specific servomotor offers ease of control and adjustment, allowing for precise gripping of chess pieces based on their respective sizes. The MG90S servomotor can be easily tightened to ensure a secure grip on the chess piece during manipulation. Its reliability and versatility make it an ideal choice for the gripping function of the robot.

4.3 Software implementation

The source code of the robot has been developed in the C programming language using the Arduino IDE. The code incorporates various control functions necessary for the operation of the robot. Additionally, a custom protocol has been created to receive commands through the serial port and process them effectively, simplifying the control of the robot. The combination of the C language code and the custom protocol enhances the versatility and ease of use of the robot, enabling seamless control and interaction with the external computer.

The logic diagram of the robot's software (Fig. 11) illustrates the sequence of actions within the main loop of the source code. After initialization, the code awaits incoming commands through serial communication.

The software had to solve 3 main problems:

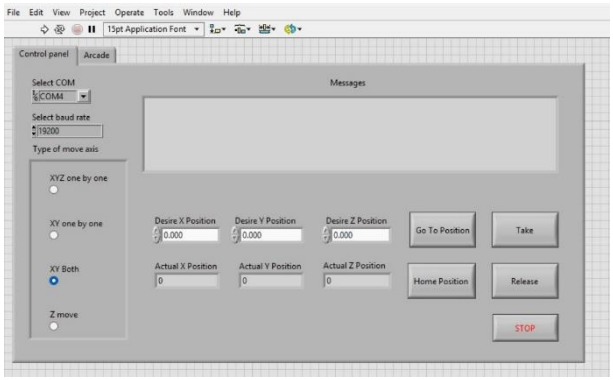

Figure 12. Control Panel interface

- Design and implement a protocol to communicate via serial line with the PC.
- Implement the actions described in the protocol, based on the available hardware and inverse kinematics (`moveToHomePos()`, `moveToPos(x,y)`, `moveZToPos(z)`, `moveJ1J2(x,y)`, `Gripperclose()`, `Gripperopen()` being the most important functions).
- Design and implement the main loop of operation (described in Fig. 11).

The command `moveZToPos(2)` reflects the need to elevate the gripper to avoid hitting pieces during the movement (the gripper z-coordinate is measured from the home position at the upper limit).

4.3.1 Communication protocol

The communication between the command computer and the controller is established through a serial connection using an USB cable. In the main loop of the source code we have implemented a function that reads the incoming data string received from the computer. This string is structured to transmit five groups of data separated by commas. The format of the message is as follows:

`data[0], data[1], data[2], data[3], data[4]`

The first data element, `data[0]`, represents the primary command. Depending on the values of this field, the other data fields are interpreted accordingly. The complete protocol is described in Fig. 11. The `data[2]`, `data[3]` and `data[4]` coordinates represents positions with a millimetric accuracy.

We highlight only the difference between `gotopos` command with subcommands `xyz` and `xy` (where the movement is done first for Joint 1 and only after for Joint 2) and the subcommand `xycom` (where the movement of Joint 1 and Joint 2 is done at the same time as much as possible). This was an improvement introduced in the later stages of the project, to speed up the movement.

4.3.2 User interfaces

To interface with the user, we have developed an application using LabVIEW software. The application features two menus: Control Panel and Arcade.

The first one, the Control Panel interface (Fig. 12), is a general one. Once connected to the appropriate COM port, users have the flexibility to execute various commands

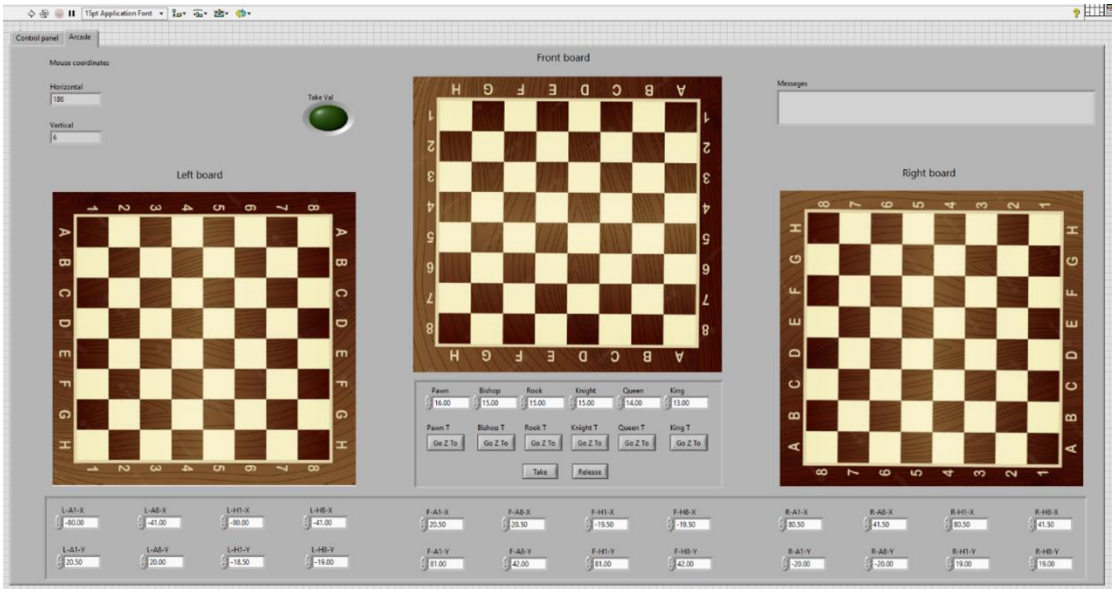

Figure 13. The Arcade Interface (boards arranged from the robot's point of view)

and perform different actions to control the robot's movements and operations. This is a general interface, not dedicated to chess playing.

The second one, the Arcade interface (Fig. 13), is specific to the use of the robotic arm for chess. There are three chessboards available, and users provide commands to the robot, instructing it to move to specific positions on the chessboards (by clicking the desired fields). This feature enables the robot to autonomously move and play chess on the 3 boards simultaneously.

To compensate problems related to the positioning of the boards relative to the robot position, the position of each board is configured by the x,y coordinates of the chess board corner fields (A1, A8, H1, H8).

To adapt to different chess sets, the z coordinate where a specific chess piece must be gripped, we configure that value for each type of piece individually.

5 Conclusions and future work

The design and implementation of the robotic arm for playing chess was not as trivial as it might look at first view. Research, design and redesign were required in all aspects of the robot: mechanical, hardware and software. For example: the motor used in the early stages of the project for the Z-axis needed to be upgraded and, for the gripper, the initial stepper motor needed to be replaced with a servo motor. Redesigning the gripper and the z-axis occurred multiple times, a total of five iterations took place until the final stage was reached. The first driver considered also had to be improved. In the software part the need to move Joint 1 and Joint 2 at the same time emerged and was implemented.

But, in the end, all these problems were overcome, and the robot met its goal: **to reliably move chess pieces on the boards with millimetric positioning precision**.

Certainly, the project can be further developed, mainly in 2 directions:

- Firstly, to use the camera that was installed in the gripper. This will enable real-time visual feedback to the PC, allowing for corrections to be made in case the chess pieces are not precisely positioned on the chessboard by the human player. The camera can contribute also to the safety aspects of the robotic arm, preventing accidents or injuries during operation.
- Secondly, the main development needed by the project is to complete the whole system by interfacing the PC to 3 DGT autosensory boards to read the positions of the pieces and by interfacing the PC with a classic software chess engine, to decide the moves to be made by the chess playing robot when playing against 3 human opponents.

References

[1] *Robots and chess*, Available from: https://en.chessbase.com/post/robots-and-chess

[2] Mark Weeks, *Novag Robot Adversary*, Available from: https://chessforallages.blogspot.com/2015/07/novag-robot-adversary.html

[3] https://www.raspberryturk.com/

[4] A. Gasparetto, L. Scalera *A Brief History of Industrial Robotics in the 20th Century*, Available from: https://air.uniud.it/retrieve/e27ce0c7-1e99-055e-e053-6605fe0a7873/AHS_2019021414562634.pdf

[5] https://www.chess.com/news/view/7-year-olds-finger-broken-by-chess-playing-robot

[6] H. Makino, "Development of the SCARA," J. Robot. Mechatron., Vol.26 No.1, pp. 5-8, 2014, DOI: 10.20965/jrm.2014.p0005

[7] https://www.fanuc.eu/de/en/robots/robot-filter-page/scara-series/selection-support

[8] https://robohub.org/how-many-axes-does-my-robot-need/

[9] Ben-Ari, M., Mondada, F. (2018). *Kinematics of a Robotic Manipulator*. In: *Elements of Robotics*. Springer, https://doi.org/10.1007/978-3-319-62533-1_16

[10] https://www.chessshop.eu/electronic-chess-boards/45-smart-board-usb.html

[11] https://www.autodesk.com/products/fusion-360

[12] *Mega 2560 Rev3*, Available from: https://docs.arduino.cc/hardware/mega-2560

[13] https://reprap.org/wiki/RAMPS_1.4