Robotic Finger

Farnoush Baghestani (810196422)
Setareh Soltanieh (810196640)
Behnam Moradkhani (810196559)
Niusha Mirhakimi (810196569)
Erfan Vahabi (810196677)

August 2021

Contents

Abstract

In this project, a robotic finger is built which tracks the motion of a human finger. Despite its simplicity, the project consists of distinct items relating to different fields of mechatronics that all come together in harmony to result in the real-time motion tracking of the robotic finger. 3D-printing of the designed parts in SOLIDWORKS, applying robot vision using mediapipe library in python, followed by coding Arduino and building the ultimate circuit by attaching the Arduino to the servo motor, are the major tasks to implement to build the proposed robot.

The function of the robot was first tested in Matlab SimMechanics environment, before having the real robot actually taken action; and the final experiments on the robotic finger were done due to the suitable results of the simulation responding to the commands of the python code that was responsible for resolving the state of a human index finger, changing position in front of a computer webcam. It is worth to mention that the simulation part was not implemented in real-time, but the the orders were stored in an excel file and later inserted in Simulink environment.

The aim of the whole this efforts is to build a tracking robotic finger with the minimum cost and complexity but able of all the bending expected from a human finger to be used in more complex robots that are supposed to grab and pick stuff.

key words: robotic finger, real-time motion tracking, robot vision, servo motor, 3D-printing, Arduino coding.

1 Introduction

Among the researches in the field of robotics, the bio-inspired robots and robotic prosthetic are playing a fairly important role [1] and also the tracking of human motion using computer vision and pattern recognition algorithms has been very exciting and developing widely, for which Kinect [2] can be referred to as a strong example.

Figure 1: Microsoft Kinect Camera for Xbox One.

One of the most important roles of these robots is helping people with physical disabilities to have better performance in their daily life activities. Disabled limbs will be replaced with mind-controlled artificial limbs. These artificial limbs get brain's signals and transfer them to motion. Also, human shaped robots can be counted as one of the most long-standing dreams of human race, by the use of these types of bio-inspired robots this dream is more achievable. The research proposed here can be accounted for one of the basic parts of the aforementioned projects.

Robot hands are intended to realize the same dexterous and versatile manipulation that we humans can do. Thus, for robot-hand research, understanding the anatomy and motion of the human hand is fundamental. On the other hand, from the point of view of human-hand research, describing the mechanisms and mechanics behind hand posture and motion helps to understand how we execute such dexterous and versatile manipulations. In this study, the design of the previous research on robot-hands in [3] is used and with the help of 3D-print, an artificial hand is obtained.

In the next step of the research the detection part is accomplished. The proposed system detects one finger's 9 different postures using media pipe in python which is explained in the following parts in detail. The base of this code is obtained from [4]. There are other methods for detecting hand poses such as vision which is explained in [5] or it can be detected using CreateML explained in [6].

The rest of this report is organized as follows. In Section 2, the construction of the considered model is explained step by step. The experimental results are also discussed at the end of section 2. Finally, the discussion and concluding remarks are reported in Section 3.

2 Robotic Finger

2.1 3D-Printing

The STL files applied in this project are obtained from [3].

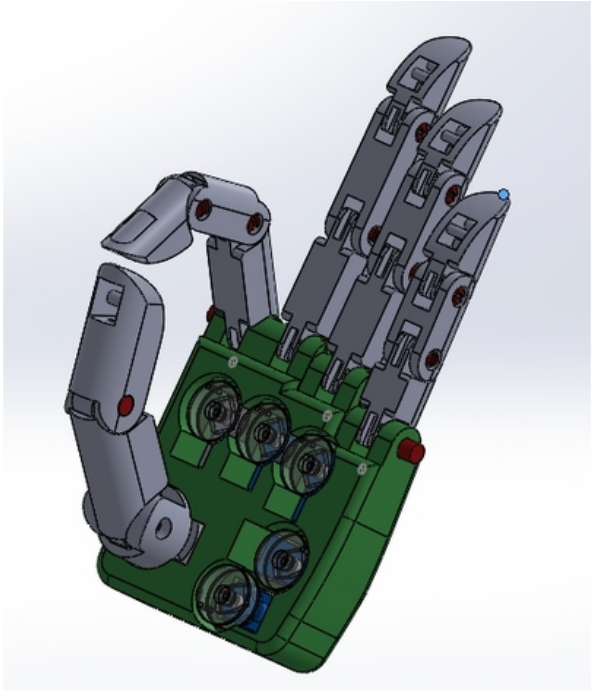


Figure 2: The Complete Model of The Hand

The aforementioned project had all the parts for a complete robotic hand including all five fingers and all the kinematic mechanisms as shown in figure 2. Only the parts necessary to build the index finger are used in this project: the palm of the hand, three links of the index finger and the corresponding pins shown in Figure 3.

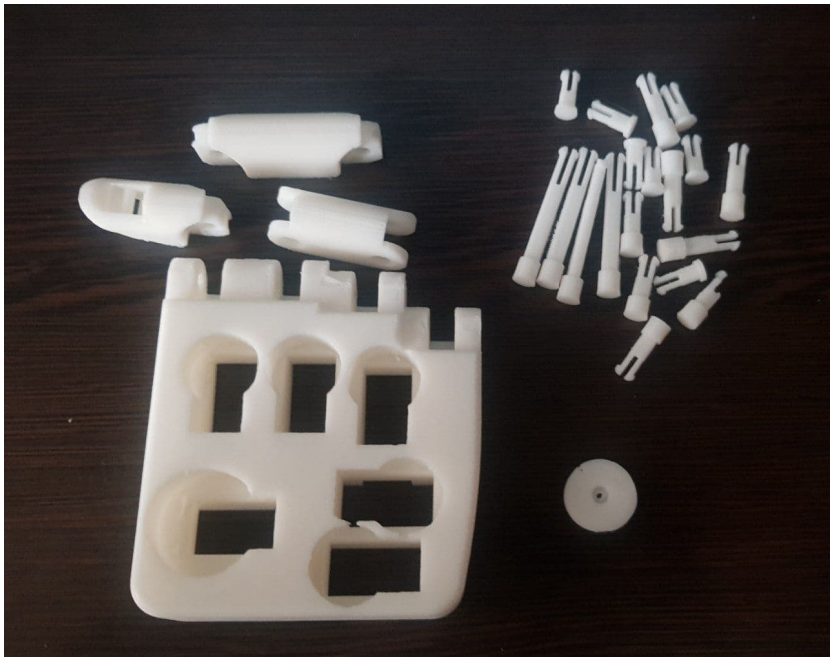
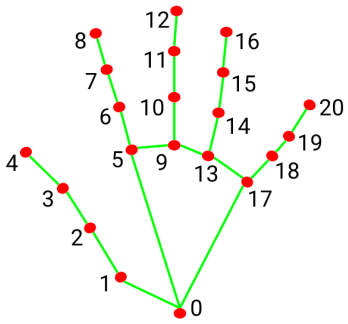


Figure 3: The Printed Parts.

2.2 Robot Vision

In this part, we used mediapipe and OpenCV libraries for tracking the finger. Mediapipe is a high-fidelity robust, and real-time body tracking solution. It employs machine learning to infer 21 landmarks of the hand. In this project Hands partition library is used for tracking hand gestures and controlling the robotic finger. Figure 4 is the map of the 21 landmarks.



0. WRIST
1. THUMB_CMC
2. THUMB_MCP
3. THUMB_IP
4. THUMB_TIP
5. INDEX_FINGER_MCP
6. INDEX_FINGER_PIP
7. INDEX_FINGER_DIP
8. INDEX_FINGER_TIP
9. MIDDLE_FINGER_MCP
10. MIDDLE_FINGER_PIP
11. MIDDLE_FINGER_DIP
12. MIDDLE_FINGER_TIP
13. RING_FINGER_MCP
14. RING_FINGER_PIP
15. RING_FINGER_DIP
16. RING_FINGER_TIP
17. PINKY_MCP
18. PINKY_PIP
19. PINKY_DIP
20. PINKY_TIP

Figure 4: Hand Landmarks

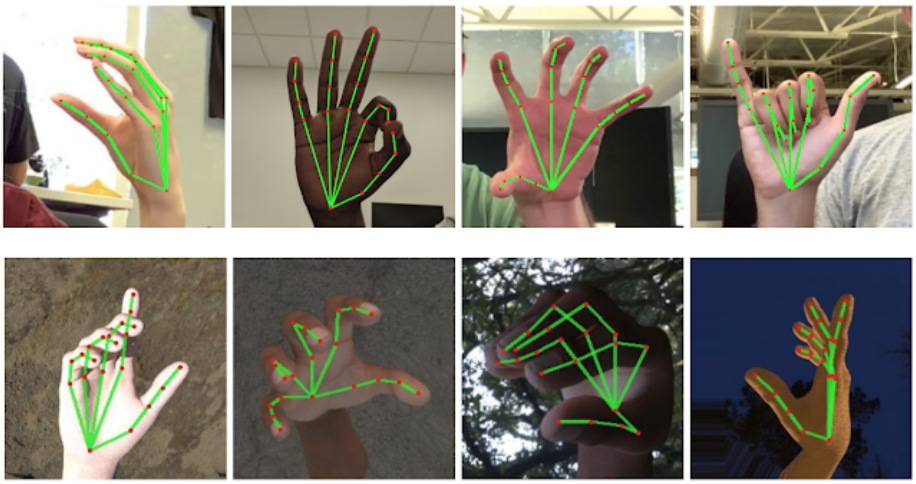


Figure 5: Hand Detection Examples

The algorithm first detects the presence of a hand in the frame when the camera is open, then reformats the BGR frame picture to RGB and passes it to a function for detecting landscapes. After catching the hand's topography, the three angles of the finger must be calculated to find its posture.

For angle calculation, the coordination of the related three joints is needed. After extracting the joints' coordination, formula 1 is used to find angles.

$$\theta = tan^{-1}(c_1 - b_1, c_0 - b_0) - tan^{-1}(a_1 - b_1, a_0 - b_0) \tag{1}$$

In Figure 6 a, b, and c are the three landmarks' (x,y) pair coordination. For better clarification, please pay attention to figure 6.

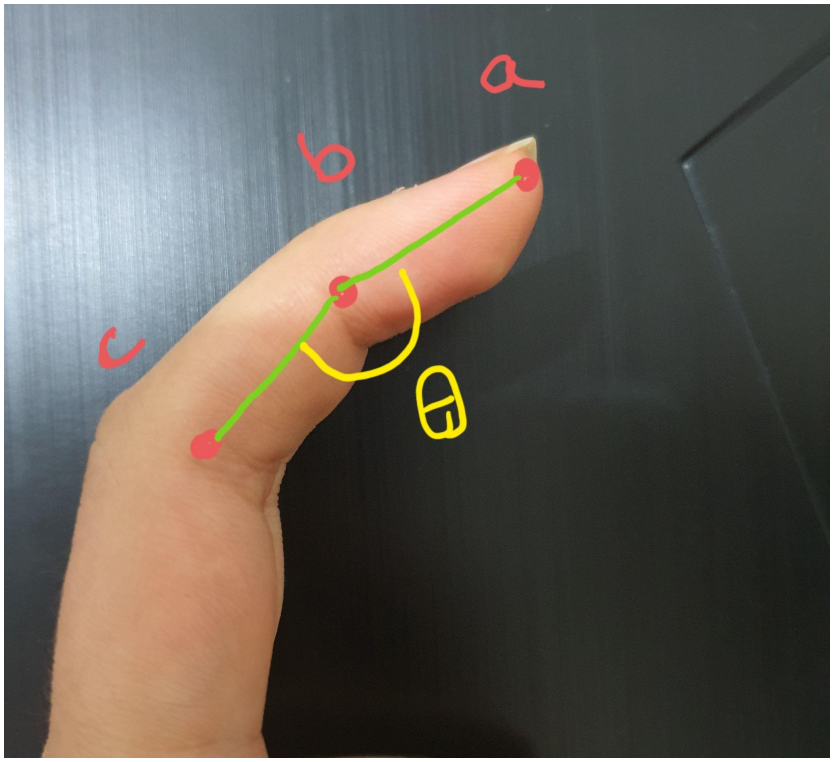


Figure 6: Landmarks Indicating θ Angle.

After realising three finger joints, the detected hand, its landmarks, and the three marked angles indicated in the monitor, simultaneously, the angles are discretized into 10 steps between 0-9 and sent by the serial ports to Arduino. Then these angles are rescaled between 0-180 by multiplying the inputs by 20. The aim of reducing the precision is to have a better real-time performance.

Since some of the angles detected may be incorrect due to the 2D video capturing or other reasons that lead to mistakes, such orders must be taken into consideration in case they do not damage the mechanical structure of the robot by some extreme movements. To solve the aforementioned problem, three last

states of the hand are considered to make the decision. It is obvious that there is a trade-off between the accuracy and the lag of the system. Also, these angles data of real-time movements are recorded in a .CSV file to be used in the simulation.

2.3 Matlab Simulation

Using SimMechanics, a Simulink file is made that simulates the final robot to build. Although this simulation does not illustrate the exact mechanism of the proposed model which is a real-time tracker, but it represents the appearances and functionality of the model to some extent.

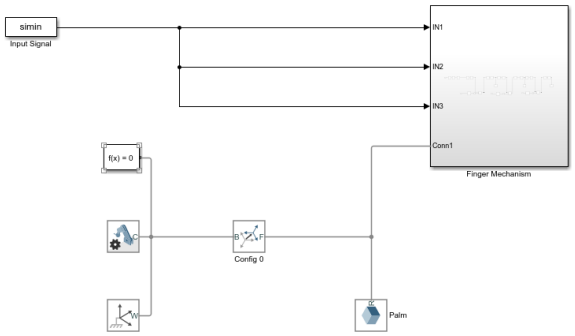


Figure 7: Simulink File Content

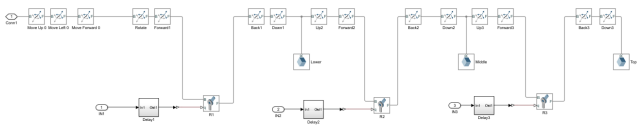


Figure 8: The Configuration of Finger Mechanism Subsystem (Figure 7)

As mentioned before, the graphics of the simulation can be accessed in the STL files from [3]. In the simulation, position control of the joints' configuration defined in the Simulink environment are used, that is the position of the joints are specified using the input signal; since this option gives the capability of moving the robot with the minimum error possible, no outputs are considered to be used as feedback or for other purposes.

By conducting the experiment, it is desired to resolve whether the designed structure behaves in the expected way or can be modified in some ways. While testing the robot in Matlab Simulink environment, there is no need to watch the reactions of the finger immediately, but it would be satisfactory if it is able to react to some predefined commands; hence the orders of the robot vision stating which pose to emulate are already stored in an excel file before the test.

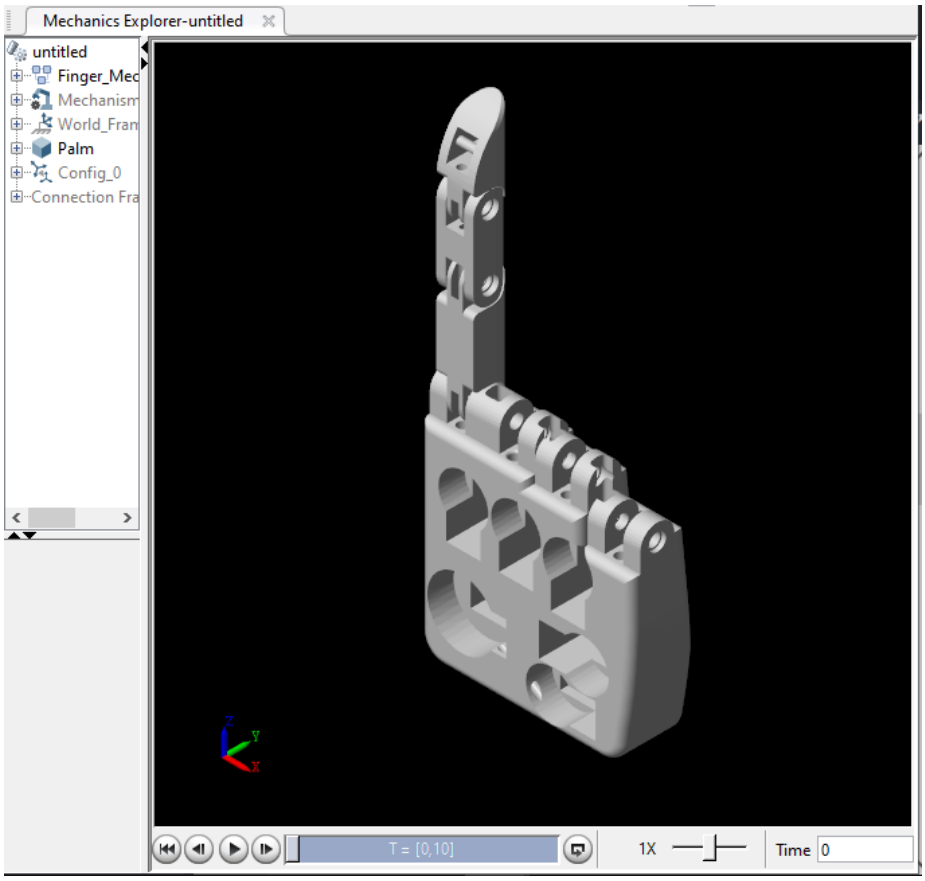


Figure 9: The Simulation of The Robotic Finger in The Pointing Position.

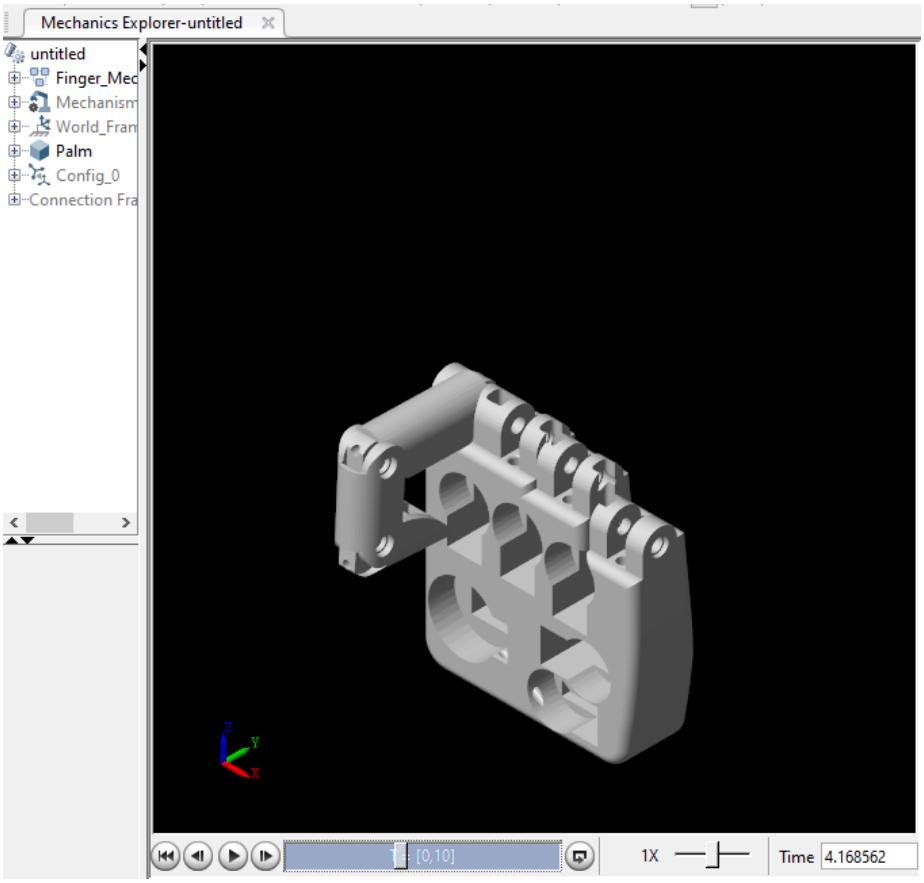


Figure 10: The Simulation of The Robotic Finger in The Bending Position.

2.4 Building The Robot

After the 3D-printing, the final model is assembled easily. The robot has 8 parts that are shown in figure 3. Here comes the description:

- 3 parts for the body of the index finger
- 3 pins to hold the parts together (since these pins are so delicate, it is highly recommended to print some extra of them.)
- 1 part that represents the palm and has some holes to place the motors (only one is needed when just the index finger is attached.)
- 1 pulley

Equilibrium position for the robot is the pointing position. A fishing line is penetrated through all the parts of the finger and is fixed on the tip of the finger. This string, on the other side, is warped around the pulley. With this mechanism, the finger is ready to bend. Still other actions are essential to bring the finger back to its initial pointed pose, which is considered the equilibrium state. On the opposite side of the finger, which is aligned with the back of the hand, an elastic string is installed to keep the finger in the pointing position; meanwhile according to its elasticity, it does not conflict with other parts when bending.

For moving the structure, a SG90 servo motor shown in Figure 12 is placed in its special place in the palm and the pulley is installed on the horn of the motor. A servo motor enables us to specify the angle of the motor in the command sent to the motor, so no sensors are needed (encoder, switch, potentiometer, etc) as a position feedback.

Servo motor is installed on the robot in a way that 0 degrees (lower limit) of the motor puts the finger in the pointing state and 180 degrees (upper limit) in the bending pose. As stated before, the angle of the servo motor can be determined in its command, so it makes it possible to put the finger in a useful number of states between the two limits.

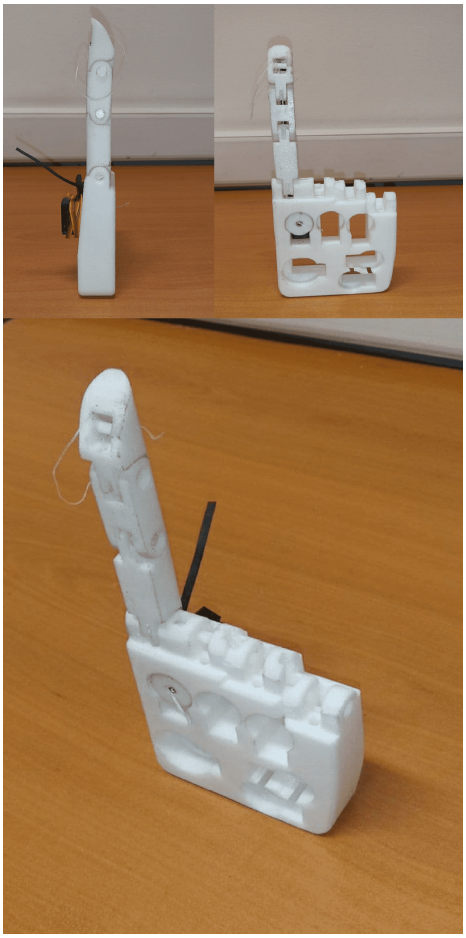


Figure 11: The Assembled Robot

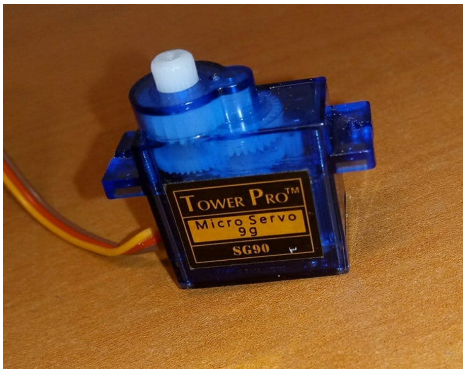


Figure 12: SG90 Servo Motor.

2.5 Arduino Coding

In this project an "Arduino Uno" shown in figure 13 is used to handle two major tasks:

1. Sending commands to the servo motor and moving the robot
2. Providing a connection between python code and the robot using the serial port (serial window in Arduino).

Figure 13: Arduino Uno

To work with servo motors, "Servo.h library" must be added. Pin number 9 on Arduino board is selected to transfer the servo commands to the motor and the suitable the commands are received via the serial port. As mentioned in section 2.2, the commands are generated by the python code.

In the Arduino code, first of all a variable must be initialized in order to hold the position of the servo motor. This variable is named *pos* that is a variable in the range of 0 and 180 degrees. Two methods from the Servo.h library has been used in the code:

1. Servo.attach : to determine the pin on Arduino board that sends the position data to the motor.
2. Servo.write : to apply the change in the variable "pos" to the motor.

The variable "pos" is updated simultaneously and repetitively according to the data received in the serial port from the python code.

3 Conclusion and Future Works

Figure 14 shows the connection between the different parts of the project explained in the previous parts. This robotic finger is successfully launched and tracks the motion of a finger in the sight of the installed camera.

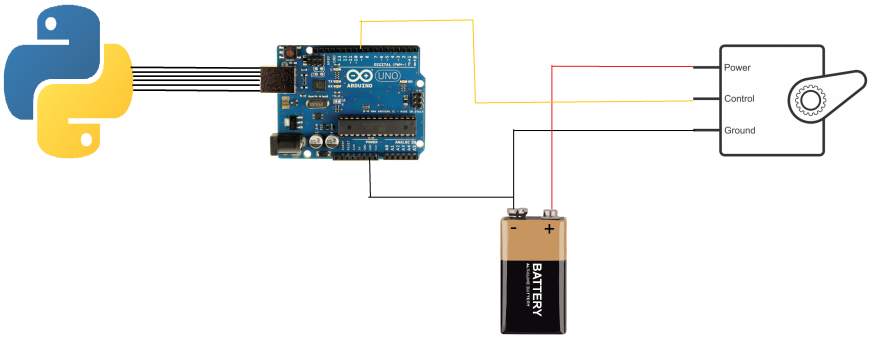


Figure 14: Connection of the Different Parts of The Project.

In the beginning of launching the robot, it was intended to specify only two states for the robotic finger: pointing and bending states. After witnessing the successful performance of the robot, and in order to modify the smoothness and robustness of the system, some other interval states were considered and added to the python code. In the final version of the python code, 10 states for the robot are determined and specified by numbers 0 to 9; this modification in the system made the performance of the robot so much better and closer to the behaviour of a real finger.

The next step in this project is to build the whole hand and do the same for all the fingers. When it is done, a complete model of a hand would be available, that can be used in a wide range of robotic structures. In order to raise the capabilities of this robot, some alterations to both the physical structure and the code behind the mechanisms can be made, for instance some other motors can be applied to increase the degrees of freedom of the robot and by this means, configuration and working spaces of the robot is widen.

Another suggestion for future work is implementing learning algorithms on the robot. One of the most novel applications of the humanoid robots and manipulators is that an operator shows them what movements should be done and the robot will learn these movements thanks to the learning and pattern recognition algorithms!

References

[1] S. R. Kashef, S. Amini, and A. Akbarzadeh, "Robotic hand: A review on linkage-driven finger mechanisms of prosthetic hands and evaluation of the performance criteria," Mechanism and Machine Theory, vol. 145, p. 103677, 2020/03/01/ 2020, doi: https://doi.org/10.1016/j.mechmachtheory.2019.103677.

[2] Z. Zhang, "Microsoft Kinect Sensor and Its Effect," IEEE MultiMedia, vol. 19, no. 2, pp. 4-10, 2012, doi: 10.1109/MMUL.2012.24.

[3] Robotic Hand EPS 2018 mechanic bionic robot, Available Online: https://www.thingiverse.com/thing:2926729

[4] MediaPipe project by google. Available Online: https://github.com/google/mediapipe.

[5] [Video] Detect Body and Hand Pose with Vision, Available Online: https://developer.apple.com/videos/play/wwdc2020/10653/.

[6] [Video] Detect Body and Hand Pose with Vision, Available Online: https://developer.apple.com/videos/play/wwdc2021/10653/.

4 appendices

Beside this report document, a presentation file (PowerPoint) and two videos, one of them of the robot in action and the second one of the matlab simulation, has been uploaded.

This project has been published in the form of a GitHub Repository. Click here to visit!

Contributors' Contact information:

moradkhanibehnam@gmail.com
erfanv1@gmail.com
niushamirhakimi@gmail.com
setareh.soltanieh@ut.ac.ir
farnoushbaghestani@gmail.com