Telepresence Robotic Hand For Videoconferencing

Jungsoo Park
Cornell University
Ithaca, US
jsp283@cornell.edu

Kui Cao
Cornell University
Ithaca, US
kc975@cornell.edu

Jessica Chang
Cornell University
Ithaca, US
hc483@cornell.edu

INTRODUCTION

Telepresence robots refers to a set of technologies that allows the experience of being present with another person at a place other than the human operator. First coined by Marvin Minsky in the context of teleoperation in 1980, Telepresence robots have a large variety of applications in areas including but not limited to video conferencing, remote surgery, pipeline inspection, and education. For our group project, we aim to design a telepresence robot specifically for the purpose of long-distance video conferencing.

We believe that telerobotics is the future in how we connect and interact with people at both personal and professional level. Some high-level examples of this technology's application are for a long-distance couple engaged in a romantic relationship to be able to interact with each other at a deeper level and for working professionals to be able to hold remote meetings over video conferencing without compensating for the efficiency and the quality of in-person collaboration. Social networking giants like Facebook are already developing VR/AR technology to create this sense of telepresence for users connecting on their platform, but we see a greater potential for improved immersive experience by adding a physical component in the form of a telepresence robot. Motivated by this, we aim to recreate and improve upon already existing telepresence robots that functions and operates effectively while deepening the human-to-human interactive experience involved in long-distance video conferencing.

We had three main goals for this project. The first goal was to make videoconferencing a more fun and interactive experience for users. The second was to design a system that is easy to use by the operator. Lastly, the third goal we had in mind was building a working prototype that operates effectively.

For prototyping, we were considering three alternatives to incorporate in our initial design. One of them was to incorporating accelerometers and distance sensors to detect the tilt angle of the arm and its movement in space, which is then translated into the movement of robotic arm. The second alternative was using flex sensors attached to gloves to detect and translate hand gestures into robotic movement. The last alternative was using kinect sensors to capture full body movements that allow the robot to perform more complex combinations of movements than the other two alternatives. Among the three design alternatives, we decided to focus our efforts on the second option that incorporates using flex sensors on a glove as input for the robotic hand movement based on technical feasibility assessment and limited time constraint to complete the project on-time.

RELATED WORKS

The field of teleoperation of robots has been an active research field and it has been getting noticeable advances for the past few decades. The related works presented followed below are the works we found that relate to the recognition of human motions through various sensors and performing imitation and control of humanoid robots.

Molina-Tanco et al conducted a study on the imitation of human motions using a stereo vision system to estimate the upper-body movements in regard to the angular range of the body joints. Stereo vision systems are computer vision implementations that use stereoscopic ranging techniques to estimate a 3D model of a scene, and it employs triangulation techniques to compute the depth from 2D images. The authors considered hands and the head were the unique parts of the user, therefore estimating the depth of those limbs was the main objective in their studies.

Cheng and Kuniyoshi used skin-color segmentation techniques to track the head and the two forearms of the user to detect the body movements. They used the head abstraction in order to orient the head-neck-torso of the robot to keep the user in a stable field of view of the camera. The authors also demonstrated how the segmented human arms were translated to one degree of freedom of the elbow and two degrees of freedom of the shoulders.

Nakaoka et al's study presented a humanoid robot imitating the dancer's dynamic body movements. Nakaoka et al. used inverse kinematics to compute the configurations of the arm joints and their own primitives defined by themselves to

estimate the configurations of leg joints. The authors also used an optical motion capture system of eight external cameras allocated throughout the room and wearable sensor with 30 detectors on users to capture the joint movements. Although the study was not performed in real time, the study carries its significance with the fact that the humanoid robot was able to mimic difficult postures of dancers and keeping its balance during its performance.

Vukobratović and Borovac also worked on maintaining the dynamical stability of humanoid robots during its performance. By adding the trajectory of Zero Moment Point, also called ZMP, to the inside the polygon of the support, they were able to make the humanoid robot to maintain its stability during dynamical movements.

Riley et al's study focused on creating human-like behaviors for humanoid robots such as catching a ball. With the use of colored marks on the human body to estimate the angular range of body joints, it allowed the robot to mimic the human motion using vision system based external camera and a head-mounted camera that is attached to the robot.

Ott et al also used a motion capture system to abstract 34 markers on the user's body and two on a conductor stick to make a humanoid robot mimic in real-time human motions in regard to a Cartesian control approach. With Microsoft's motion sensor add-on for the Xbox 360 gaming console, Kinect, a number of applications and research projects have implemented Kinect with the robot.

Chernova and Suay used Kinect to control a humanoid robot so that it can imitate the movements of human arms such as ordering the walking and gaze direction of the robot. Veltrop also used Kinect but made the addition of a Wiimote controller that has an infrared and inertial sensors for the accurate recognition of motions and also a head-mounted display for users to see the view of the robot and make its neck move according to the user's view.

As we looked into more recent projects for further research on our related works, we were able to find a few projects that gave us insights for possible features and implementations. In the project *CakeRobot*, Madhusudana designed a gesture-driven robot that follows the user's hand movements. The Arduino board is mounted on the robot car to drive the robot. As the robot car and the Kinect sensor are connected via Bluetooth, the Kinect sensor continuously tracks down a skeletal frame of the car by calculating the key points in the frame.

Madhusudana's study gives us a comprehensive demonstration of how we will be implementing Kinect sensors in our project, what data output Kinect sensor generates and how to connect the Kinect to the Arduino board.

We also found a similar project called *Kinect & Arduino Controlled Stickman* that uses Kinect to keep track of 20 skeleton joints and passes the coordinate of the left hand and right-hand joints to Arduino to control the prototype. This project indicates that controlling a human-like robot using the Kinect sensor is possible.

Another project we found is *IR helicopter*, which carries its relatively high resemblance with CakeRobot, but the addition of IR LED in the Arduino board was made to control the helicopter, showing that the control signals can be transmitted via IR. This feature can be adopted by our project to make our robot separate from the Arduino board in terms of a short distance.

DESIGN DESCRIPTION

Our design is centered around one basic functional mechanism, which is using sensor gloves to detect operator's hand gestures so that a robotic hand can imitate the movement. The mechanism involves four main functional components: Sensor gloves, microprocessors, actuators, and robotic hand.

From a high-level point of view, a glove with built-in sensors captures human hands movements like clench and loose. Those data are then processed by the microprocessor to map control signals to the actuators attached to the robotic hand. The actuators then execute the instructions from the microprocessor to move fingers in the robotic hand. As illustrated in the figure below, it's a simple sensor-microprocessor-actuator structure. The whole system can be segmented into the following functional blocks:

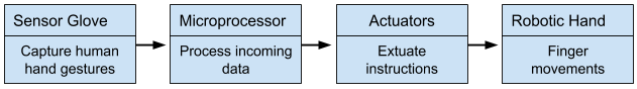

Figure 1. Functional Blocks

The first functional block is the sensor glove. We expected the sensor glove to capture movements of each of the five fingers. We also expected it to meet the requirements of wearable technology in terms of being light and convenient to put on or perform gestures with. The second functional block is microprocessor, which holds the responsibility of processing data we collected from the sensor. By mapping the incoming data to the output instruction, we expected the actuators to copy the movements of how much our fingers bend and stretch. The third functional block is actuators. We wanted actuators to be able to cooperate hand-in-hand with the robotic hand. We expected them to be able to effectively control and drive the finger components, but we didn't want too many actuators attached to the robotic hand because it can be expensive and can make the movement feel unnatural. The fourth functional block is the robotic hand. We expected each finger to have its own flexibility so that it can move independently from one another. This

means that the operator will be able to make wide range of hand gestures that greatly enhances the interactive component than simply clenching and loosening fist (2 motions). We also expected the robotic hand to emulate the shape and movements of a real human hand as much as possible. In the following section, we will elaborate on each functional block and feature.

Sensor Glove

The first functional block or feature is the sensor glove. Our goal was to attach a total of five flex sensors on top of a glove that can capture the movements of each finger independently. Flex sensors are ideal because they are compact, affordable, and easy to implement. The flex sensor we used is shown in figure 2. This sensor can detect flexing or bending in one direction. One of the advantages in using flex sensors is that they are extremely easy to implement because they are essentially resistors that change value based on how much they are bent. If they're not bent, the resistance is at around ~25KΩ. When they are bent all the way, the resistance rises to approximately ~120KΩ [1]. The sensor was tested by using it as an analog input on a microcontroller with a pullup resistor. The value was reported on the terminal. When we bend the flex sensors, we could observe varying values proportional to the degrees bent. Thus, the bending degree of each finger can be measured.

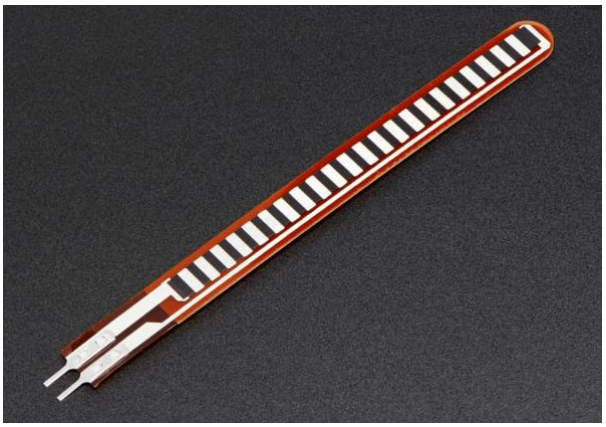

Figure 2. Flex Sensor

A total of five flex sensors was used to monitor the bending degree of each finger. The five flex sensors were combined together with its connection to a microprocessor shown in figure 3 [2].

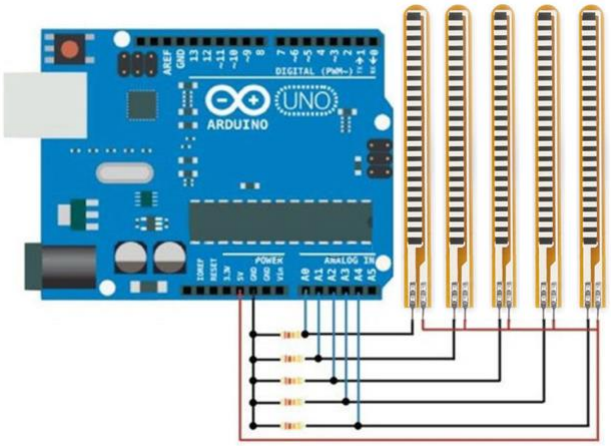

Figure 3. Five Flex Sensors Connected to Microprocessor

Our initial design of the sensor glove involved simply taping the two ends of the flex sensors on top of the finger parts of the glove. We later realized that simply taping the ends of the sensors on the fingers compromised accurate reading of the sensors and bending degree. Thus, we cut another glove to use it as a cover for the sensors to tightly stick to the finger. The performance is evaluated by how well the flex sensor can fit with the movement of a finger so that the bending degree of a finger is the same as the bending degree of the flex sensor. After using the new method of covering another glove on top of the sensor glove, we could intuitively find out that the flex sensor matches perfectly with the bending finger. This significantly improved the accuracy and reliability of sensor readings that will be used as an input for the microprocessor and actuators, which will be discussed in the next section.

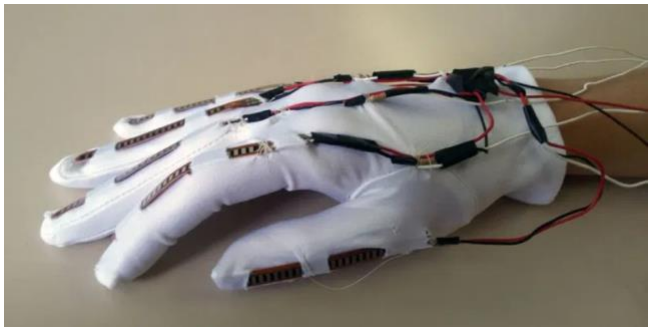

Figure 4. Sensor Glove

Microprocessor

The second functional block is the microprocessor. For our selectin of microprocessor, we used ESP32 over Arduino Uno because the latter only has gpio pins. Also, in comparison to Arduino Uno, ESP32 has both Bluetooth and WiFi capabilities. Implementing wireless connection was not our design goals for the purposes of our final project, but it is one of the primary features we are considering

implementing in the future, which will be discussed in the later section. Another advantage of using ESP32 is that it is fast and has a large memory space. On top of technical advantages, we also have familiarity with ESP32 and function coding from other in-class exercises and projects dealing with potentiometer and pressure sensors.

Figure 5. ESP32 Microprocessor

The main function of this block is to process the analog input data from flex sensors and outputs the digital signal for servos. Since the properties of the flex sensor is very unstable, the resistance can change rapidly even for an un-bended state during measurements. This could result in constantly turning of the servo motor. To address this, we developed a filtering function program in microprocessor that integrates a sliding window algorithm with a window size of ten. During each iteration of the program, we took the average of the previous ten readings of the flex sensors and served as an input to the servo. After testing the program with a real servo, the algorithm proved its capability of filtering out the noise coming from unstable sensor readings.

We used only one microprocessor for our initial design hoping it could power five servos at the same time. However, the servos responded erratically to the sensors. Five servos began to vibrate when the power is on. No matter how much the flex sensor is bending, the servo didn't follow the established coding procedure. It took several minutes for these servos to finally become stable and respond correctly to the bending sensors. We ignored the abnormal behavior in our initial design since it didn't case too much trouble. Only when we increased our testing time for our final prototype, something bad happened. There was smoke coming out of the isolator with a burning smell. We had to go back to find out what causes the erratic movements of servos and the smoke of isolator. It turned out to be the power provided by one microprocessor is not enough to support five servos. That's why the servos vibrated without following the codes in the beginning. Somehow, as we consistent bent the flex sensors, which forced the servos to move, the servos eventually drained a lot of power from the isolator and burned it. Therefore, in our final design, we added one more microprocessor that runs the same code as the previous microprocessor. We assigned two flex sensors and two servos for one microprocessor and assigned the remaining three flex sensors and three servos for the other. After that, the servos responded normally to the sensors and the isolator never burned no matter how long we were testing the system. We consider that a better performance compared to the initial design for its robustness.

Actuators

The third functional block is actuators. For our selection of actuators, we decided to implement servos (shown in figure 6) over steppers because servos provide high speed and high torque. The input for the servos is served by the microprocessor, which used filtering function to smooth out the sensor readings. We premeasured the output for relaxed state and bending state of the flex sensors and mapped the output to the turning degree of the servo. This means that the output of the relaxed state will map to 0 degrees and the output of the fully bent state will map to 180 degrees.

We utilized a total of five servos to actuate control signals from the microprocessor. One servo was used per finger to achieve our target goal of controlling the fingers independently from each other. The servos were attached to the metal strings that run through the robotic fingers and directly control their movement and they were mounted on a laser-cut MDF board.

Figure 6. Servo components

Our initial design involved using a single string tied to the servo to control the movement of the finger. However, due to some reasons stated below in the robotic hand section, we changed to two string mechanism. In our initial design stage, we encountered some problems with the servos. The

servos behave erratically in response to sensors. It turned out to be the microcontroller's problem, and we fixed that as explained in the previous section. Generally speaking, the servos work well through each stage of this project. The performance is evaluated by how precise the servo can rotate in response to the control signals and if it can provide enough torque to pull the string that connected to the finger. We first tested how many degrees for each servo must turn to completely bend a finger. And then we mapped the flex sensor readings to that many degrees so that the servo can provide different degrees of rotation representing different degrees of finger bending. It worked pretty well for our final prototype, all fingers can reach their desired positions by the pulling of servos.

Robotic Hand

The last functional block is the robotic hand. Our goal for developing the robotic hand was to emulate a human-like hand visually as well as functionally by resembling the natural movement of the human hand.

In our final design, each finger used a total of three strings with two strings running through the top and one string through the bottom of the finger. The three strings were attached to a single servo that pulls and releases appropriately to clench or loosen the robotic fingers. All the palm, fingers, and external arm frame components were 3D printed using the rigid PLA material.

Figure 7. Robotic hand final design

Our initial design involved making the robotic finger out of a rubber-like elastic material so that the robotic finger can be controlled by using only a single string. We initially settled with single string mechanism because it is both easier to implement and requires less number of strings to deal with. As a result, we actively looked for the appropriate elastic material that we can use to build the fingers out of. One of the suitable materials we found online was TPU, which is short for Thermoplastic Polyurethane and has great elastic property, but we found that it was too expensive and required a long delivery time. Also, the single string mechanism was material-dependent, which meant that even the slightest changes in material property from overuse or misuse could be detrimental to the robotic finger's operability. Therefore, we eventually decided to pivot away from a single string mechanism to settle on the double string mechanism because of its reliability and effectiveness.

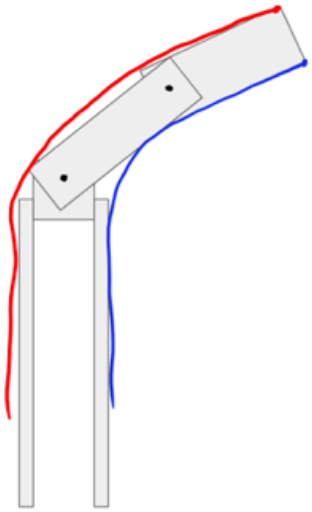

Figure 8. Double-string mechanism

Implementing the double string mechanism allowed us to control the movement with only strings that run through the back and front of the robotic finger. And later in our final design, we found that strings can be loosened after several pulling, which could make the finger fall apart. To solve this problem, we decided to insert a rubber band into each finger. The rubber band is not mechanically driving the fingers in any way, but the elasticity of the rubber band allowed finger components to stay intact and provided more stable structure for the string to drive the movements. Inserting a rubber band also provides extra safety and back-up for our robotic hand system. If either the back string or the rubber band failed, the whole system can still work normally.

Another challenge was designing the fingers in a way that permitted strings to easily move inside the finger and with appropriately sized joints that resemble a natural movement of the finger. After a series of testing, we found that the slanted design of the joints would be the most effective in achieving this goal. After long hours of research and testing as described in the prototyping process for third and fourth functional blocks, we were able to make the robotic hand function well in coordination the servos in the end. Each

robotic finger resembled the natural movements of the human finger and was controlled independently to allow wide range of hand gestures.

Integration

The integration of each functional block was another big challenge. We wanted our design to be compact, good-looking and easy to debug, which required our connections to be simple and the circuit to be as compact and efficient as possible.

For the first connection from the sensor glove to the microprocessor, each sensor had two wires and each sensor glove used five sensors. We bounded the wires altogether as an input from the sensor glove to the microprocessors. All the circuits and microprocessors were sitting on four combined breadboards, which were glued on the laser-cut MDF board. On the top half part of the laser-cut board was five mounted servos. The servos connected the microprocessors by wires and connected the robotic fingers by strings. The 3D-printed robotic palm was glued on the very top of the laser-cut board. Except for the sensor glove that will be put on by users, all other components were attached to a single 13.5-inch by 5-inch laser-cut board, where we can easily see the connections between each functional block. This design makes it easy for us to debug and also ensure all parts were close to each other. To make our robotic hand better looking, we covered the laser-cut board with a 3D-printed arm frame. The frame was designed to have a slider in it, which makes it easy to put on or remove. The final prototype is shown in Figure 9.

Figure 9. The final prototype of the robotic hand

FUTURE WORK

Although our final project accomplished many of our target design goals initially defined and discussed in the introductory parts of this report, we envision the following extensions that can be added to further improve the usability of our project in the context of video conferencing. One of the extensions to add is incorporating wireless connection over WiFi between the sensor gloves and robotic hand to allow the operator to remotely control it. This is perhaps the most important extension to add because without this capability, our design cannot be applied to a long-distance videoconferencing context.

Another extension that can be included is to add additional joints below the palm and elbow to increase mechanical functionalities and movement range of the robotic hand. Our current final design can only move the fingers, which can make the interaction limiting with the robotic hand limiting. By adding this extension, the potential for interactivity increases significantly because the robotic hand can perform actions such as high-fiving the partner or grabbing an object nearby.

Lastly, the third extension that can be added is to integrate natural language processing to activate the movement of the robotic hand via verbal and text input. Our current design only lets the operator to control the robotic hand by wearing the sensor glove. By adding this extension, it will not only make it easier for the operator to control the robotic hand via verbal or text signals, but it also increases interactivity by adding a motion component to emojis that you send to express your feelings to your partner.

REFERENCES

1. Adafruit. 2019. Short Flex Sensors. Retrieved October 18, 2019 from https://www.adafruit.com/product/1070
2. IPHACOGOODS. 2019. 2.2/4.5 inch Bend Flex Sensor Robotic Arm Power Glove etc. US. Retrieved October 18, 2019 from https://www.iphacogoods.com/index.php?main_page=product_info&products_id=622961
3. Matthew Burris. 2019. Choosing Between Stepper Motors or Servo Motors. Retrieved October 18, 2019 from https://www.lifewire.com/stepper-motor-vs-servo-motors-selecting-a-motor-818841
4. Pucci Marco. 2014. Arduino Project-5: Robotic Hand. Retrieved October 18, 2019 from https://celebratelife24x7.wordpress.com/2014/11/17/arduino-project-5robotic-hand/