# Gesture Controlled Wrist-Worn Display

Shi Chen, Feiyu Wang

## Introduction

For this project, what we wanted to do is a gesture detector that can be easily scaled to larger size and can be easily integrated with other components. The current version of the device can detect four gestures such as swiping up, down, right and left and direct the output to a LCD screen.

The final product is easy to scale, with some modifications of the code, we can achieve detection of hand movements that are getting closer or farther from the device, with this, we can improve the 2D detection we have right now into 3D detection. If we increase the amount of sensors and position it so that it points towards different angles, we can have a fully functional wide motion detector that can find out objects' position and shape in an area and map that to 3D space.

## Design diagram

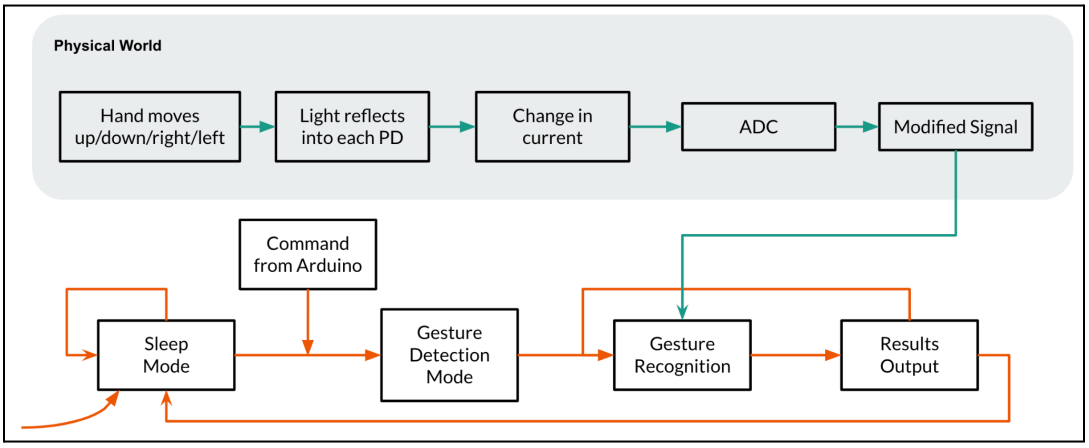


Fig 1. Block diagram of system

Our current gesture sensor system is composed of two subsystems which are the physical world subsystem and the digital world subsystem (Fig1). First, in the physical world subsystem, it starts from the hand motions that are made by users. For now, we only focus on the four basic hand motions that can be detected for this project which are upward, downward, leftward and rightward motion. Moving to the next stage of the physical world subsystem, each action or hand motion will reflect infrared(IR) light into four photodiodes (PDs) in different orders. The detailed working principle for PDs will be included in the hardware section. Then there will be a current change in the circuit and it will be detected using Arduino's analog reading functions. After acquiring the singles from analog pins, they will be converted to digital signals using the 16-bit analog-to-digital converter in Arduino Mega board. Now we have obtained the modified signals from the physical world subsystem and it is ready to be sent to the digital world subsystem for the actual gesture recognition.

Next, for the digital world subsystem, it is composed of multiple finite state machines (FSMs) which are designed to control the system to reach different states. It first starts at a sleep mode which shuts down all the powers that go into the components of the gesture sensor. These components include four PDs, one IR LED, and four op amps in the circuit. Then at some point in time, the user may require reading hand

gestures, a command will be sent through the Arduino digital port and activate the gesture detection mode. This is a non-deterministic action which means the user will input the command at any time or does not input at all.  In this mode, the digital pins that connect those components will be set to high which provide power to the circuit and warm up the LEDs. In the detection mode, the system will wait for a rise in the modified signal from the physical world sunsystem and determine the corresponding hand motions. Thai results will be sent to a TFT screen that is connected to the Arduino board and displays it to the user. Lastly, the user can decide to input another gesture or the system will automatically go to sleep mode after a fixed amount of time has passed.

## Specification and modeling

In Simulink, we created a complete model for our gesture sensor. We started the process with building the circuit model for the four PDs and IR LED. In Fig 2, The simulated gesture input is defined using four individual repeating sequence stairs to represent four reflected light sources that go into the PDs. For each PD, they all have the same power source which is a 5V DC voltage source and they are all grounded. Using current detectors, the output can be visualized when connecting the outputs of current meters to a scope.

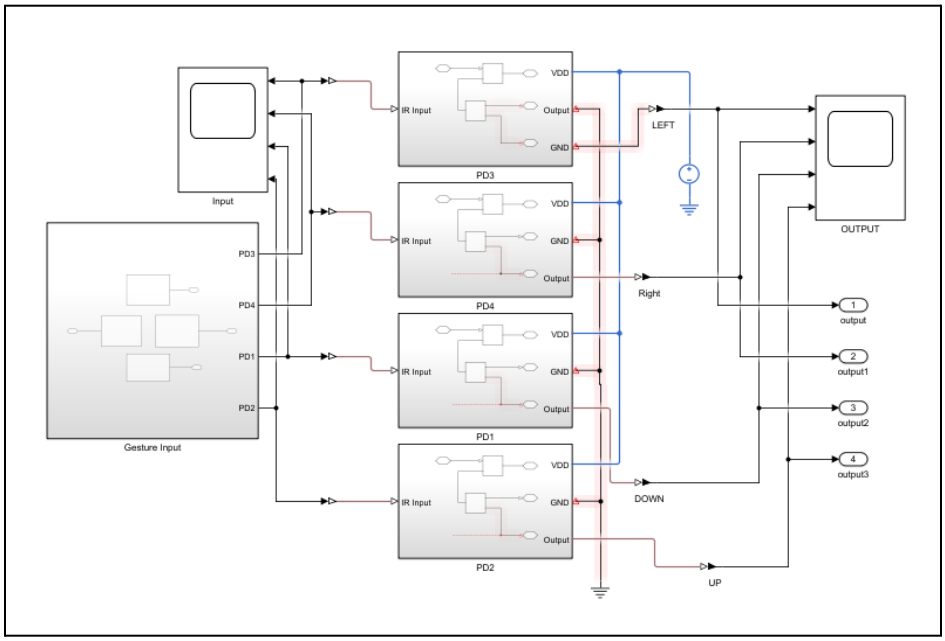


Fig 2. PDs and IR LED model diagram (Simulink)

Then, we built the FSMs using the stateflow feature in simulink to control different states in our system. In Fig 3, the two machines are a cascade composition of FSMs in which the output of the machine on the left is one of the inputs for the machine on the right. The system will first enter the *Sleep* state and *Idle* state of the two machines. If the input *Status* is false, which means the user has not requested the gesture recognition command, the system will be stuck in the *Sleep* state in a loop. If the user does not send a command (*Mode* = 1), the machine on the left will stay in *idle* state in a loop.

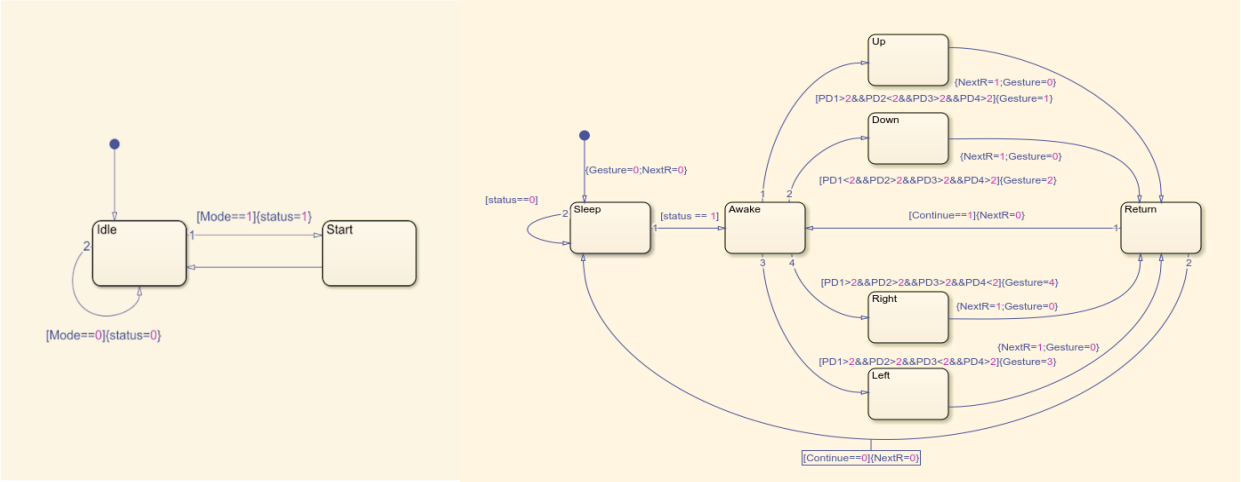


Fig 3. Finite state machines using stateflow

Next, at some point in time the user decides to request a reading for the hand motions, the input *Mode* will become 1 and output the status as high to the FSM on the right. Then the machine enters (*Start, Awake*) states, where the system starts to read the signal from the physical world. For now, the threshold is set to be 1 and if the signals pass the threshold, the machine moves to *Up/Down/Right/Left* state and generates an output *Gesture* to the scope for visualization. Lastly, it reaches the *Return* state and it can go back to sleep mode or return to the awake mode for more hand motion detections. The complete model for the gesture sensor is shown in the following figure (Fig 4):

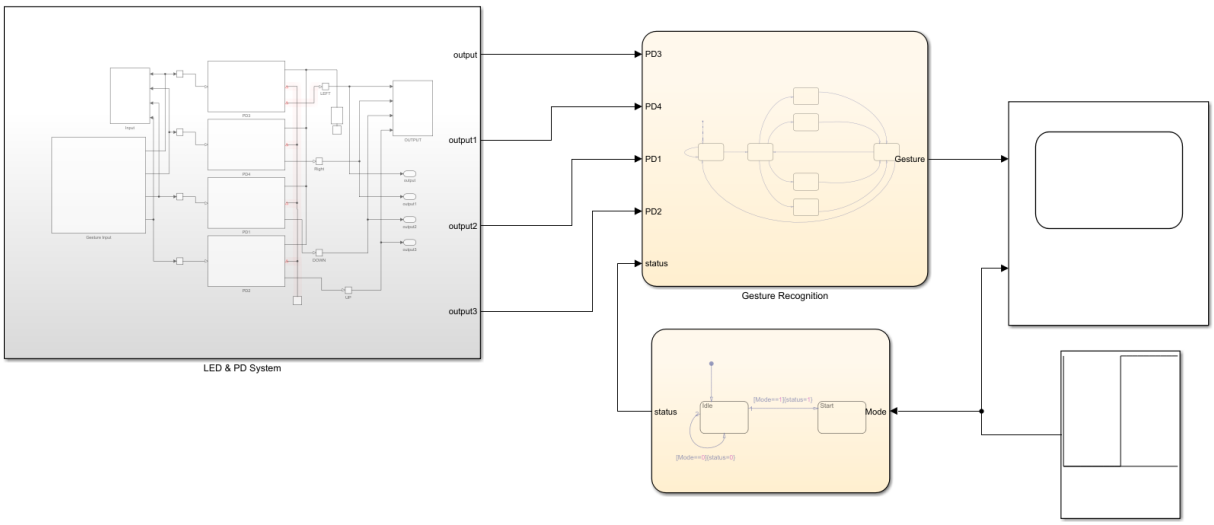


Fig 4. Complete model for gesture sensor in Simulink

## Hardware implementation

In order to build the physical world subsystem, we created a circuit diagram for our sensor and we soldered every element on a prototype PCB board. We connected the ports A0-A3 (Fig 5) to the analog pins A0-A3 on the Arduino Mega board for testing. We connected the board to the computer and used Arduino Analog Input blocks to obtain the signal detected on board.

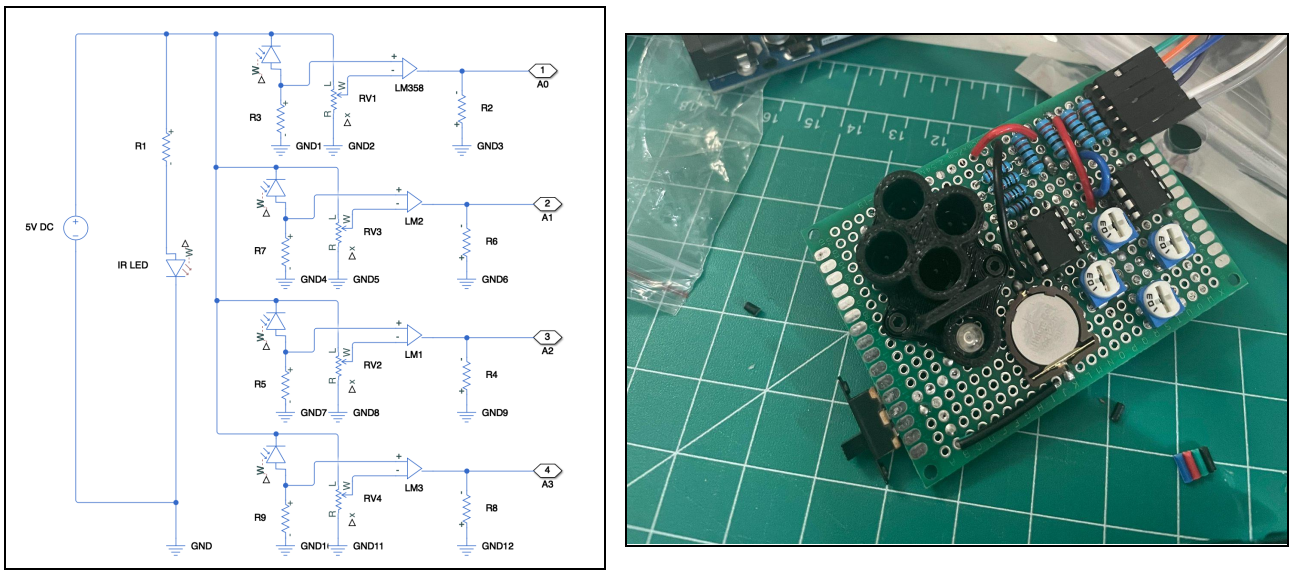


Fig 5. Circuit diagram and PCB board for gesture sensor

We also designed an optical barrier for the four PDs and the purpose is to extend the time differences between signals so that it is easy to measure. After obtaining the signal reading from the arduino board, the outputs for four different hand motions are shown in Fig 6. This has proved that the design of this sensor is able to detect gestures. For example, in the figure 6 which shows the result for upward motion,

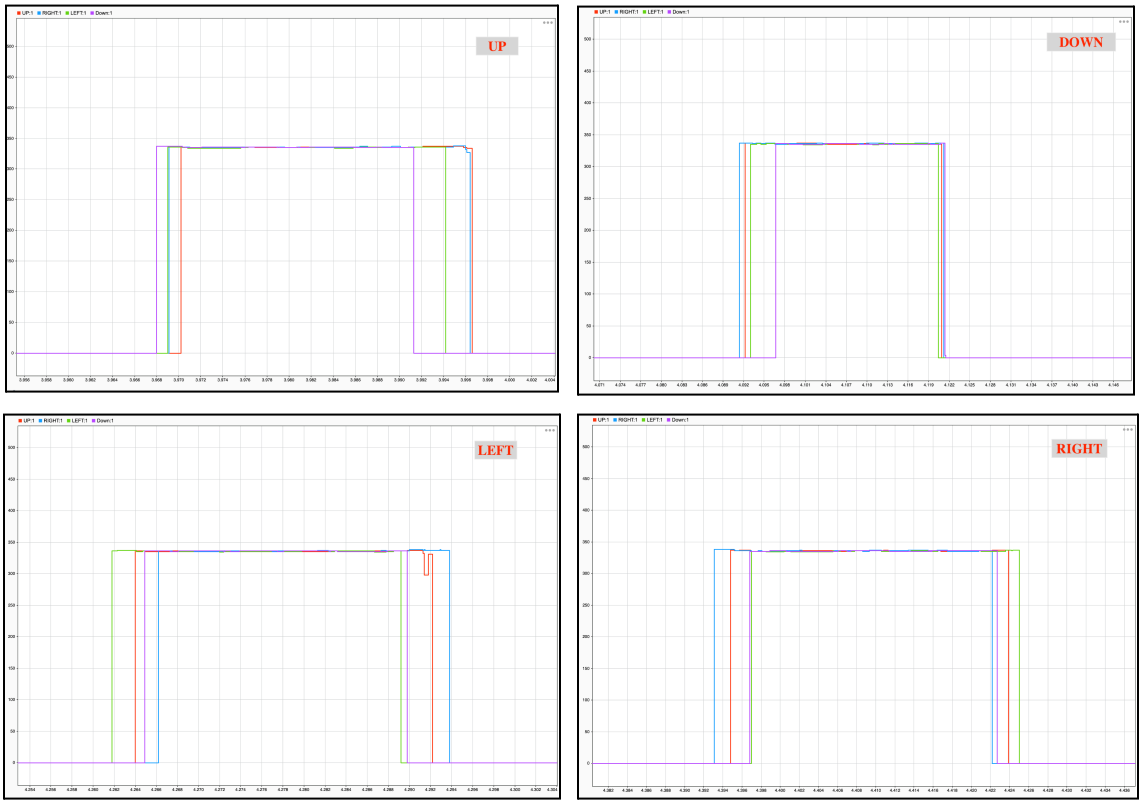


Fig 6. Test results for the analog reading from Arduino Mega

the PD at Down position will pick up a light input first. Then the PD at Up position will rise in voltage. This identification process is the same for the rest of motions. Since we have successfully detected all the signals caused by hand motions, now we can move to the next stage for our project which is the software implementation.

## Software Implementation

```
up = analogRead(Up);
down = analogRead(Down);
```

Reading from analog

```
Threshold = 550;
if(up <= Threshold && down >= Threshold)
  u = true;
if (up >= Threshold && down <= Threshold)
  Serial.println("UP");
```

Setting and using threshold to detect objects

Here is a very simplified version of the code that assumes if we only have two sensors and we are trying to detect gesture that is waving from top to bottom, we read the sensor information from the analog, and we say if the value of the sensor is larger than the threshold, which we already know the value by testing, in this case is 550, is might be different if we are using a different sensor. If this value is larger than the threshold value, then we know there is an object that has blocked the upper sensor, thus we know the hand is at the top of our device. However, at this point we don't know if it is waving or just staying put, so we add another check which checks if the object has moved away from the top sensor and the button sensor is still detecting the object, in that case. Is safe to say this object has moved from top to bottom, thus finishing the wave action. And this detection system will be running in a loop, keeps detecting gestures until it has been powered off or received a shutdown signal by the user to break the loop.
From this very basic idea, we made some adjustments to make it fits the purpose better, the first one being instead of having only one threshold, we made it two thresholds, a near-threshold and far-threshold, we only take any input in between those values, in this way, we can root out the influence from cloth that the person is wearing by the near-threshold and things that are very far from the user by the far-threshold, it still have some limitations which could be unintended could still be picked up by the sensor, but it did improve some aspect of this problem during real life situation. The sample code for this case would be if (up >= 100 && up <= 550 && down >= 550) u = true;.

## Analysis

Our product is able to detect hand gestures with a high accuracy in four directions, we have done 100 tests per direction, and were able to achieve a 91% accuracy on average. Most of the false results were detecting other directions to right swipe, which is detecting object movement from left to right.
We tried two things to narrow down the problem, one is to set the threshold to a smaller interval, in that case the chance of unwanted interference will be lower, however, this doesn't help us getting better results.

The second way we tried to find out the problem is to move the left sensor to top, and this time is getting a large amount of down swipe detection, we think that this could be caused by a sensor defection, or this sensor is overly sensitive, picking up false signals of an object even if there are no objects close by, we think we have find the problem, but with no extra sensors to replace and test, we are not certain that this is indeed whats causing us having this issue.
Given that out of 300 tests we did in other directions, about 70% of the false direction detection were detecting right swipe, we believe that with this problem fixed, a higher accuracy rate will be achieved.

### Division of work

For this project, Shi Chen did the FSM and circuit designs in simulink and Feiyu Wang focused on the coding for the sensor and arduino board. Then we combined our work and made the prototype as a team. We also tested the prototype together and performed analysis for our system.

## Conclusion And Future Work

In conclusion, we were able to build a gesture detector that is able to detect object movements and is moving direction with high accuracy. However, giving the size of the board and the sensors combined, is hard to fit the device onto the wrist.
In the future, we are going to try to make it smaller so it can fit on the wrist. Given that this device is very integrated with other devices, we are planning to add a radio-frequency module and a gyroscope to our system.
The gyroscope can help us finding out which direction the device is pointing towards, for example, if we are having a situation where the user is using it upside down, the top sensor should be bottom sensor, and the display should be flipped, this function is important for portable devices and with gyroscope, we can achieve those functionalities.
With the help of radio-frequency module, we can have wireless communication going between our display and gesture recognition device, this allows us to remove the computational limitations off the wrist worn device by having an other device like computer to do the computation heavy work and we only have to transmit the raw data to that computation device and transmit the result back to our device, in this way, we don’t need the large microcontroller, making everything easier to fit on wrist.
Overall, we think this project has a lot of potential. It might be a simple implementation of gesture detection at this moment, but it can be further expanded by going wide range detection with a sensor array or 3D detection and object mapping with code modification.

Appendix

Github link: https://github.com/FeiyuWang/Gesture-Controlled-Wrist-Worn-Display

References

*INFRARED GESTURE SENSING - silicon labs*. (n.d.). Retrieved December 16, 2022, from https://www.silabs.com/documents/public/application-notes/AN580.pdf

Molchanov, P., Gupta, S., Kim, K., & Pulli, K. (2015). Multi-sensor system for driver's hand-gesture recognition. *2015 11th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG)*. https://doi.org/10.1109/fg.2015.7163132

Rao, N. I. K. I. L. (2017). *a comparative performance evaluation of low-power gesture sensors: Semantic scholar*. A Comparative Performance Evaluation of Low-Power Gesture Sensors | Semantic Scholar. Retrieved November 16, 2022, from https://www.semanticscholar.org/paper/A-Comparative-Performance-Evaluation-of-Low-Power-Rao/41afaea923aafb3d37bfeabb318dd3ace870d36c