Capacitive Sensor Array for Object Classification

John Warwar, *MSEE, USC Department of Electrical Engineering*

Manoj Sunkara, *MSEE, USC Department of Electrical Engineering*

Rana Eltahir, *MSEE, USC Department of Electrical Engineering*

Sinthia Sivapalan, *MSEE, USC Department of Electrical Engineering*

Mohammed Arfan, *MSEE, USC Department of Electrical Engineering*

ABSTRACT : **Recently robotic arms, especially in applications of medicine, are growing rapidly. Research on object identification and letter classification using tactile sensing robotic arm has been useful. This paper suggests the development of a prototype for a Flexible Capacitive sensor array for object classification with a robotic arm which consists of a 2-layer capacitive sensor array interfaced with an analog front end which processes the readout from the sensor is developed. The 2 layered capacitive array provides 36-pixel values via two 8x1 multiplexers. A neural network is then used to classify the object based on their profiles sensed by the array.**

I. INTRODCTION

Biomedical machines, robots, and wearable devices have been emerging over the past few decades to better the lives of people in the world. Particularly focusing on wearable devices, flexible and stretchable wearables come under trending topics of research. Here we deal with an object recognition robotic arm where tactile sensing is seen to be a significant feature for identification. Tactile sensing is used for robotic manipulation, surgical robotics, medical imaging, consumer product recognition, security systems and automotive domains. And each of these applications use different techniques of tactile sensing. In the market tactile sensing has been significant in touch devices in the market such as mobile phones and uses techniques such as piezoresistive, piezo electronic and capacitance.

The known sensing used for Silicon based tactile sensing does not have enough flexibility to grab an object. Considering the case of object detection for a robotic hand, flexible material would come in handy to sense and detect an object. Robots require tactile sensing to detect touch and to control the minimum grasping force required to hold an object. A flexible substrate is required for such applications such as perylene, polyimide or Polydimethylsiloxane. This necessitates the development of flexible tactile sensors.

Flexible capacitive tactile sensing array with floating electrodes developed by M-Y Cheng et al [3] essentially works on the principle of air gap change between the sandwiched electrodes which thereby changes the capacitances when pressure is applied. In this design there are two sensing electrodes on a flexible substrate and one floating electrode on the top again on a flexible substrate. The electrical contacts are only made to the bottom layer. The floating electrodes are just to alter the total capacitance upon pressure application.

Most capacitance sensing array detect normal force or pressure applied with a sensing electrode on a flexible substrate and floating electrode with an air gap in between as depicted in Figure 1,

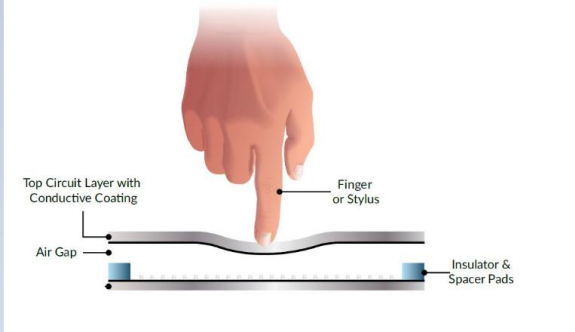


Figure 1: Capacitive sensor

When a pressure or force normal to the floating electrode is applied the gap between the electrodes reduces which increases the capacitance which helps in detecting the pressure or force applied. A PDMS layer is used to hold the floating electrode, as it allows a certain level of sturdiness to the floating electrode array layer which moves closer to the sensing electrode when force is applied. For slippage detection in robot hand, a polymer based capacitive sensing array with normal and shear force measurement could be used developed by Ming-Yuan Cheng et al [3]. This type of sensing uses four capacitive tactile sensing arranged in a 2x2 array with a pillar at the center for shear force sensing. This would be suitable match for the development of the robotic arm. Apart from this, some capacitive sensors look the fringing capacitance between two parallel plates are depicted in the Figure 2:

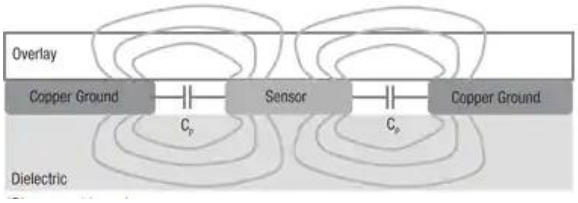


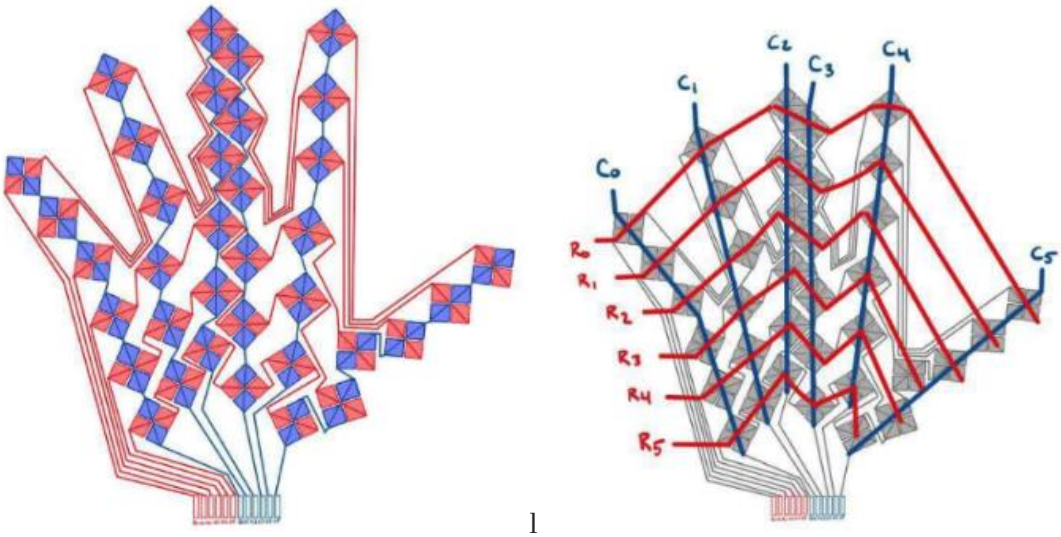
Figure 2: Fringing capacitance detection

II. MATERIALS

The capacitance array consists of 36 channels composed of six rows of six sensors. In Figure 3, the red row capacitor is the bottom plate, and the blue column capacitor is the top plate. These rows and columns will allow for both deformation and fringe induced capacitance changes. With each row and column, we have both Fringe field and multi layered capacitance to increase our sensitivity. For fringe field, it measures the capacitance two parallel plates. For multi layered capacitance, it's the capacitance between a row and a column. The columns will be connected to one mux and the rows will be connected to another, so we can access a specific element in the capacitance array. For example, if one wants a reading from the top capacitance on the thumb, it would be C05. Using the multiplexers, we will select ROW 0 and COLUMN 5 to the channel read out. The capacitive array was fabricated using a standard PCB etching method.

The Columns and rows of the 6x6 capacitance sensor array is connected to two Analog mux M1 and M2, respectively. This Analog mux are 8x1 74HC4051 from Texas instruments that help us switch between the columns and rows to select each pixel of the capacitance array. The mux is controlled by the digital signal on the 3 select pin. For example (LOW, LOW, HIGH) would select channel 1 and (LOW, HIGH, LOW) would select channel 2. These digital signals are provided by the Arduino BLE33 MCU which is an nRF52840 from Nordic Semiconductors, a 32-bit ARM® Cortex®-M4 CPU with Bluetooth low energy. Once the pixel has been selected by switching the two multiplexers. The capacitance is measured with a capacitance to digital converter Analog front end FDC2214EVM from Texas instrument. It is a high SNR and High-resolution touch sensing AFE with 4 channels of which only 1 is used here. The data from the AFE is fetched over I2C by the MCU. Following figure is the schematic for the above-described system and the Figure is the layout of the PCB of the system. Figures 4 and 5 show the block diagram of the system, the schematic and the PCB of the redout system.

.



l

Figure 3: Array with 36 pixels 6 Rows and 6 Columns

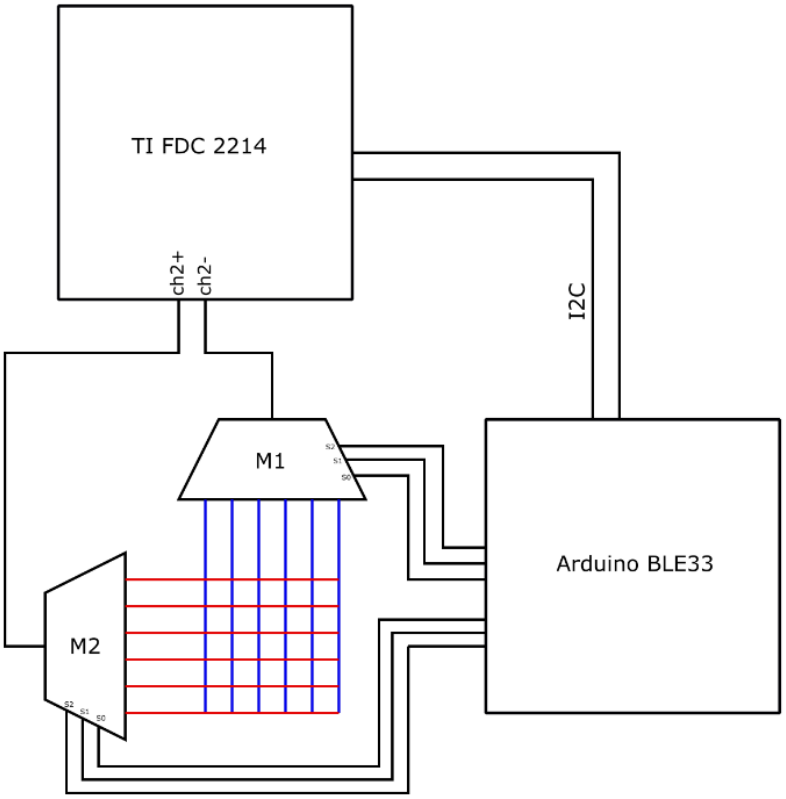


Figure 4: System for the measurement

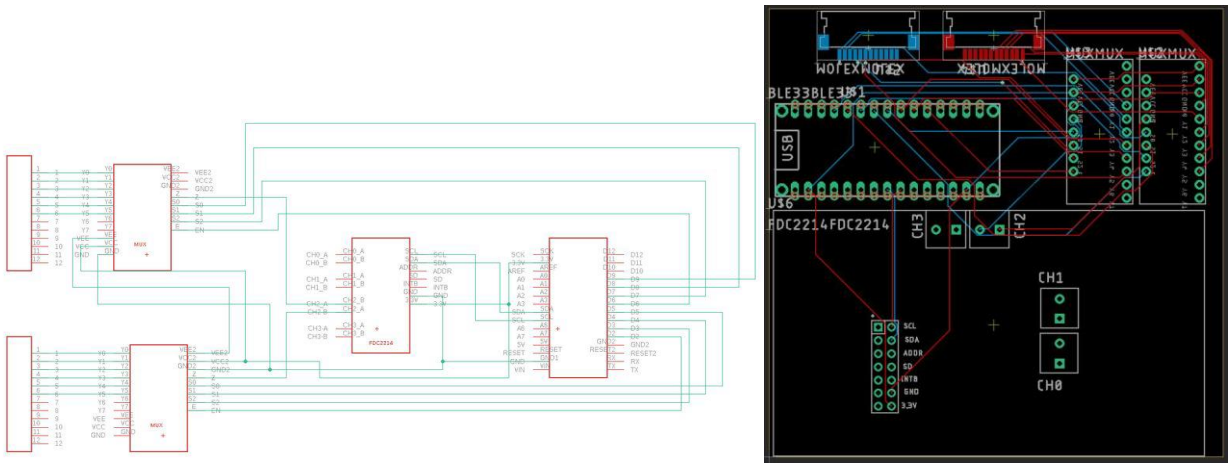


Figure 5**:** System Block diagram and PCB

III. DATA COLLECTION

For the data collection portion of the design, the Arduino BLE33 was used in conjunction with a Python script running on a PC to retrieve the data from the sensor which is processed into a usable format. In future designs, all the code could be implemented onto the microcontroller which would be more suitable for a real-world application of the sensor but would require significantly more convoluted C code as compared to Python. The same functionality that is being demonstrated could be achieved this way, but for the purpose of debugging and making significant progress on the design at this stage, we found this approach to be more practical. The Arduino microcontroller interfaces with the Python script using the UART protocol.

For the capacitive sensing, we use a multiplexing scheme to scan through the 36 pixels of the 6x6 array. The TI FDC 2214 capacitive-to-digital converter provides only four +/- channels, which was a bottleneck for our implementation. While we could have divided the 36 pixels up among the four input channels for a higher speed design, we would have needed more multiplexers as well as an additional processing step to filter out any offset between the input channels.

The actual reading of the capacitor value is continuously done in the Arduino main “loop”. In this loop, the “capsense.getreading28(2)” function call uses the C++ library provided by the IC manufacturer. This is followed by a delay statement which sets the minimum time per loop cycle at 50ms. We then read data from the PC serial port into a string which, in which we read 100 characters. With 100 characters in this time period, we end receiving about five capacitive readings from the microcontroller. To minimize potential error, the first and last readings are dropped in case of clipping on these values due to the asynchronous nature of the system. These three values are summed and then averaged based on the number of elements that were stored. It should be noted that if the delay statements were doubled from 50ms to 100ms, we would expect to get 10 readings instead of five, or 8 useable readings after removing the edge values. This averaging scheme is done for all 36 elements in the array which are successively stored into a two-dimensional 6x6 matrix.

In the Python script, there are three modes that allow the user to interact with the sensor.

1. Single Pixel
2. Calibrate
3. Scan

Pressing the ‘1’, ‘2’, or ‘3’ characters on the keyboard will change the mode in the script. In the single pixel mode, one pixel of the array is selected with the multiplexers and printed to the screen. Pixels can be changed by hitting the spacebar, followed by typing “m x y” where x and y denote the x and y locations of the individual pixel the user wants to move to. This is most helpful in the debugging stages of the design. In calibration mode, the 6x6 array is constructed using the scheme that is discussed above. The array holds the 36 readings from each pixel, which are non-normalized and 28-bit values. In the scan mode, the normalization of the readings takes place. The procedure on constructing the array follows the same steps as in the calibration mode, but additional processing steps are used. Once the array is formed with the raw values, it is normalized using the following scheme. The measured array, “measArr”, is subtracted element by element from the calibrated array “baseArr". This difference is then divided by “normVal” which is a hard coded integer value which represents the largest possible change in reading that we found the sensor capable of detecting using substantial force. This subtraction and division normalized the array so that when no object is present, each element of the array should be approximately 0. When the scan operation is done, the measured and normalized values are stored in a 6x6 array called normArr. As weights are added, the analog capacitance readings of each pixel will fluctuate between 0 and 1.

IV. MACHINE LEARNING

The system was used to capture the data and profile of 2 (a mouse and Air pods) objects and when no objects were places on the array. A total of 1050 data points were collected with 350 data points for each object. All the data points were for the objects placed in a specific place and orientation. Thus, the system is not capable of detecting these objects in different places or orientations. A relatively simple neural network with 36 inputs, 3 hidden layers and 3 outputs was used as shown in Figure 7. The hidden layers have 72,36 and 18 neurons each respectively. The model gave an accuracy of about 83%.

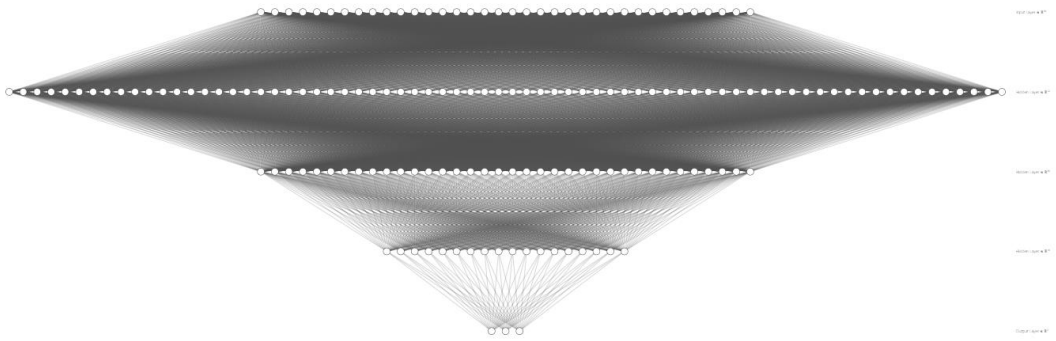


Figure 6: Neural network used for training

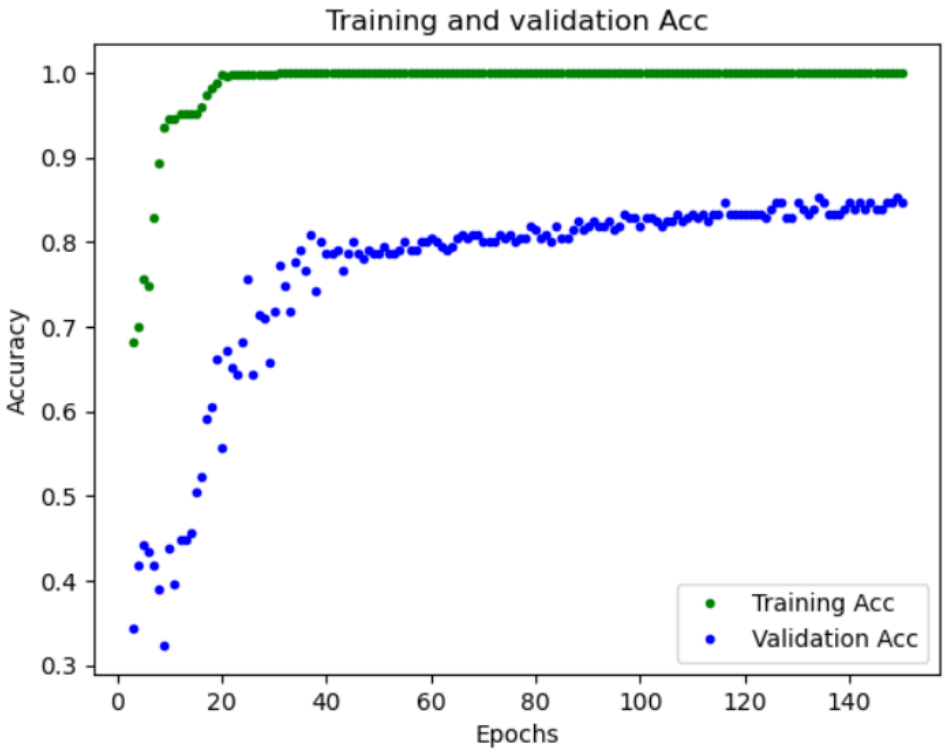


Figure 7 : Validation vs Training network accuracy

V. CONCLUSION AND FUTURE WORK

PCB was created to house all the components of the design. The Array primarily detects changes to fringing capacitance as the dielectric material used is not deformable. The effects of fringing on a single pixel of the array were potted in the figure. The multiplexers used add different levels of capacitances for different channels. These channel capacitances were obtained by measure the array readings 4 times and averaging the values. Due to this variable capacitance, the readings were not constant. However, the capacitance added by each channel and by extension, each pixel remained constant as depicted by the repeating wave show in Figure 8. This is confirmed over multiple measurements as depicted in Figure 9. Then, the values are normalized by subtracting these from the current reading. Each Scan takes 8 seconds.

As depicted in the figure, the spikes correspond to when a flat card was hovered 1cm and 2 cm above a pixel on the array. However, the baseline changes after the first operation. This drift in base line is observed sporadically throughout the testing process due to various reasons including improper contact of the z-axis tape as shown my Figure 9. To account for this change in baseline, the system was recalibrated after every scan. The Figure 10,11 and 12 below show the orientation of the objects and their corresponding results.

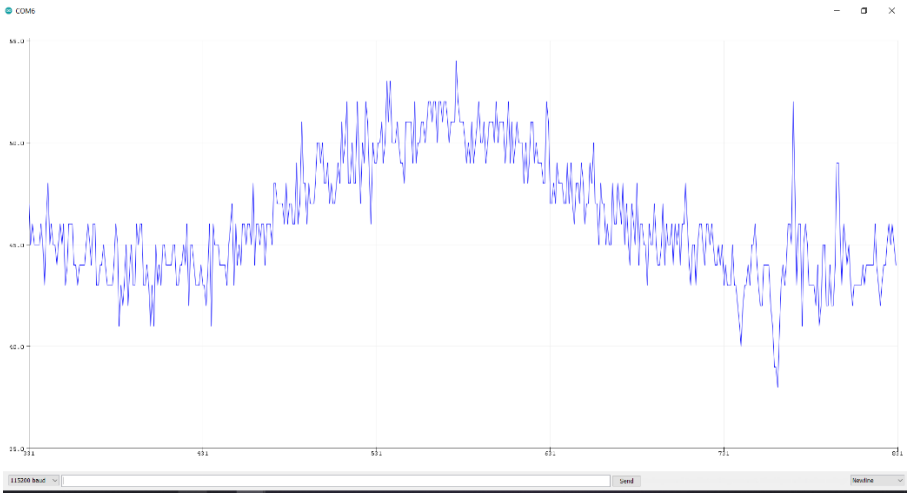


Figure 8: Capacitances added by the Multiplexers without normalisation.

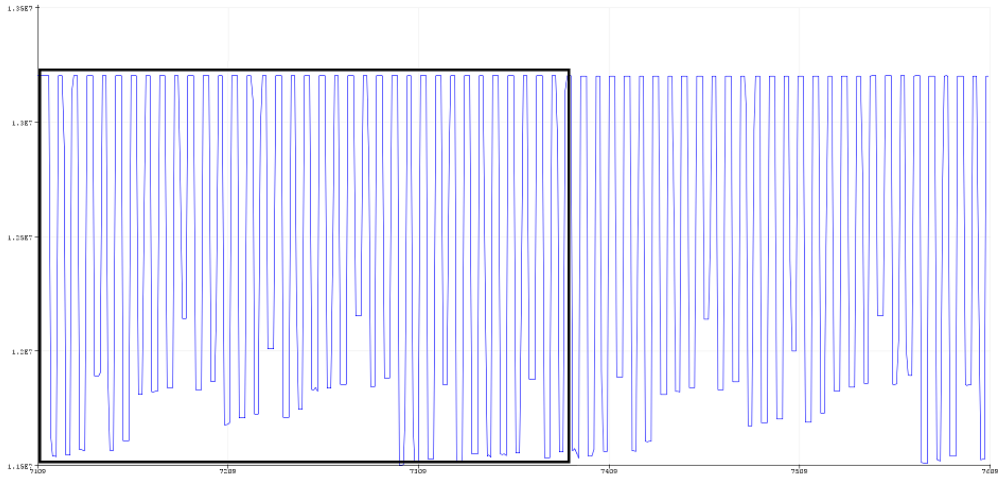


Figure 9: Capacitances added by the Multiplexers. The high point is when the multiplexers are disabled, and each spike captures the additional capacitance added by selection each pixel by the multiplexers

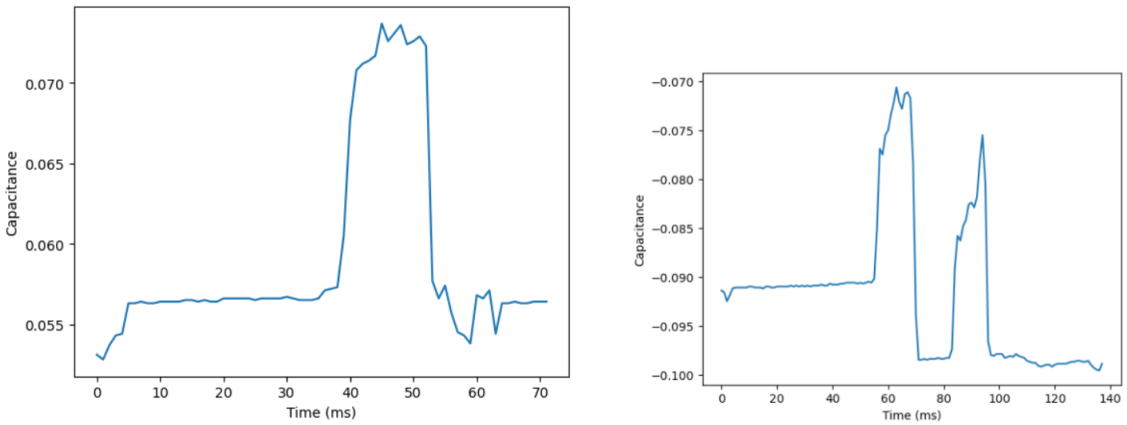
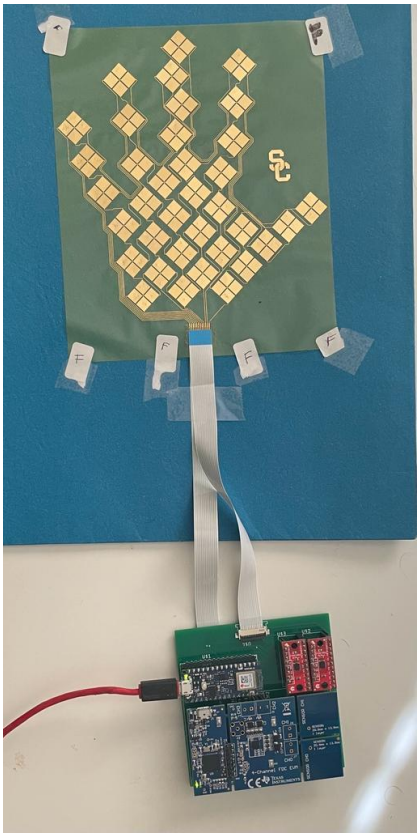
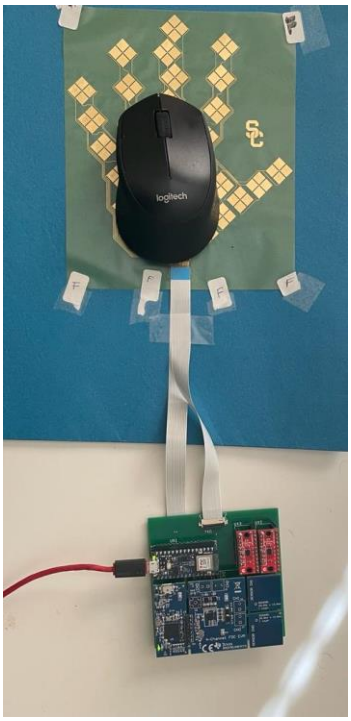


Figure 10: Detection of Fringing capacitance.



```
Running Calibration
 . . . . . .
[[12092981.3 12439687.  12092311.2 12126831.  12366497.3 12172574.5]
 [12078076.  12399659.  12058826.7 12097368.  12336478.  12143714. ]
 [12081129.5 12399681.  12042505.7 12089239.5 12332125.7 12140630.3]
 [12057907.2 12400619.7 12044810.3 12066692.3 12318311.2 12129177.7]
 [12025678.  12372426.  12027634.3 12045677.  12278907.2 12093391.3]
 [12017161.  12376418.3 12032556.5 12055799.  12282670.7 12077753.5]]
Scanning
 . . . . . .
**************************
scan 8
0
**************************
[[ 0.044   0.0001 -0.0007  0.0009  0.0005  0.0019]
 [ 0.055   0.0023  0.001   0.0009  0.      0.0002]
 [ 0.0457 -0.      0.0008  0.0013 -0.0002  0.0002]
 [ 0.0158 -0.0001 -0.0002  0.0006  0.0002  0.0008]
 [ 0.0063  0.0005  0.0008  0.001   0.0003  0.0004]
 [ 0.0038  0.0021  0.0022  0.0023  0.0023  0.0019]]
**************************
Detecting Object...
Empty
```

Figure 11 : Empty Array detection



```
Running Calibration
 . . . . . .
[[12055946.7 12080097.2 12090649.3 12121712.7 12365720.8 12172013. ]
 [12034355.7 12038818.  12058171.8 12092988.7 12336431.7 12144594.8]
 [12046067.3 12048103.  12042264.5 12084709.  12331524.3 12140630. ]
 [12041569.8 12056503.3 12044621.3 12062425.2 12317646.  12129293.7]
 [12015188.5 12045003.7 12025922.7 12040882.  12277562.7 12083097.3]
 [12010796.7 12029690.2 12029774.3 12044458.3 12029772.5 12066483.3]]
Scanning
 . . . . . .
**************************
scan 49
0
**************************
[[ 0.0455 -0.3079  0.0544 -0.0547  0.3534  0.1599]
 [ 0.0464 -0.1449  0.057   0.0782  0.3482  0.1562]
 [ 0.037  -0.0088  0.0567  0.0771  0.3478  0.1569]
 [ 0.0426 -0.3819  0.0563  0.0657  0.3486  0.157 ]
 [ 0.0367  0.024   0.0593  0.0552  0.3489  0.1446]
 [ 0.0109  0.0224  0.0608  0.0787  0.0653  0.1463]]
**************************
Detecting Object...
Mouse
```

Figure 12: Detection and Orientation of Mouse

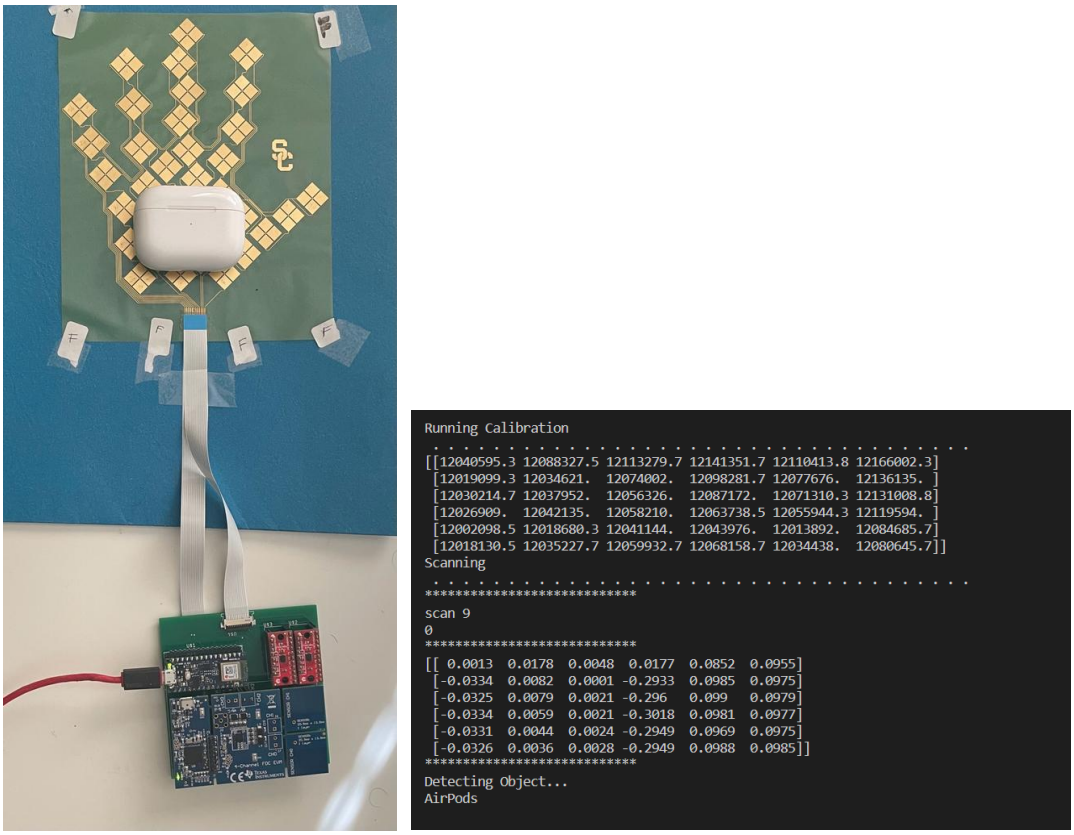


Figure 13: Detection and Orientation of AirPods

VI. CONCLUSION AND FUTURE WORK

The implementation proved that the System can be successfully used to detect objects using a capacitive array. Furthermore, It was able to detect and classify 3 different objects with relatively high accuracy. The FDC2214's ability to capture the capacitance in a 28-bit value provides us with a extremely system. However, the substrate used for the array in this project was not deformable. Thus, no change due to compression was detected. The scope of future work for this project is broad. A few improvements that can be suggested are:

- Switching to a 64-pixel model and there by using all 8 channels of the 8x1 Mux
- Using different Multiplexers that does not add variable capacitance when the channels are switched.
- Adjust the channel switching times to make quicker scans.
- Utilizing all 4 channels of the FDC2214 board to reduce scan time
- Utilize all 4 channels of FDC2214 to increase the density of the pixels.
- Making the machine learning models more reliable by getting more data with the objects in different places and orientations on the array
- Placing a deformable material as the substrate so changes in pressure can be detected.
    - This opens the possibility of getting the density, hardness of the materials as well

Apart from these, the array can be improved for various applications.

VII. REFERENCES

[1] Li, Jianke & Zhao, Yanxia & Li, Jianfeng & Hu, Aihua & Wei, Ruoyan & Zhang, Changshui. (2019). A Multi-Channel Data Acquisition System for Capacitive Sensing. Journal of Physics: Conference Series. 1213. 052102. 10.1088/1742-6596/1213/5/052102.

[2] Zuqing Yuan,"Transparent and Flexible Triboelectric Sensing Array for Touch Security Applications." in ACS Nano 2017 11 (8), 8364-8369.

[3] Cheng, M.-Y, "A Polymer-Based Capacitive Sensing Array for Normal and Shear Force Measurement. " Sensors,, 2010, 10, 10211-10225.

[4] H.B. Muhammad, "A capacitive tactile sensor array for surface texture discrimination," in Microelectronic Engineering, 2nd ed., Volume 88, Issue 8,2011,Pages 1811-1813.

[5] Cheng, Ming-Yuan et al. "A flexible capacitive tactile sensing array with floating electrodes." Journal of Micromechanics and Microengineering 19 (2009): 115001..

[6] Fukang Jiang, "A flexible micromachine-based shear-stress sensor array and its application to separation-point detection," Sensors and Actuators A: Physical., Volume 79, Issue 3,2000,Pages 194-203,.

[7] [6] Qize Zhong "Paper-Based Active Tactile Sensor Array," Advanced Materials.09 October 2015.

[8] Jonathan Engel, "Development of polyimide flexible tactile sensor skin," 2003 J. Micromech. Microeng. 13 359.

[9] Wang L, "A silicon-based shear force sensor: development and characterization" in Sensors Actuators

[10] Schmidt P A, Mael E and Wuertz R P 2006 A sensor for dynamic tactile information with applications in human–robot interaction and object exploration Robot. Auton. Syst.

[11] Hasegawa Y, Shikida M, Ogura D, Suzuki Y and Sato K 2008 Fabrication of a wearable fabric tactile sensor produced by artificial hollow fiber J. Micromech. Microeng.