Creating Novel Interactions with EIT-Based Devices through a Mobile Enabled API

by

Joshua Verdejo

S.B. Electrical Engineering and Computer Science
Massachusetts Institute of Technology, 2021

Submitted to the Department of Electrical Engineering and Computer Science

in Partial Fulfillment of the Requirements for the Degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2021

Author . . . . . .
Department of Electrical Engineering and Computer Science
May 18, 2021

Certified by . . . . . .
Stefanie Mueller, Professor of Computer Science and Engineering
Thesis Supervisor

Accepted by . . . . . .
Katrina LaCurts, Chair, Masters of Engineering Thesis Committee

Creating Novel Interactions with EIT-Based Devices through a Mobile Enabled API

by

Joshua Verdejo

Submitted to the Department of Electrical Engineering and Computer Science
On May 18, 2021, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

Traditionally, electronic devices have some type of physical input. Whether those are the buttons, triggers, and joysticks of a video game controller, or the flat, multi touch screens of mobile devices we have become used to, technology has pushed the limits on these physical and visible inputs. While developments continue to be made, such as pressure sensitive displays and flexible touch surfaces, there is an entirely different aspect of technology that has not been as utilized: the invisible input. Electrical Impedance Tomography (EIT) devices measure electrical conductivity and impedance of a part of the body using noninvasive surface electrodes, e.g., a wristband, and form a tomography image of that part, which can be used to measure internal muscular changes. In this way, users are able to imagine pressing a button, or simply make a gesture in open space, and have that gesture correspond to some command. Moreover, these gestures do not have to be sophisticated at all -- the simple act of flexing and relaxing a muscle would be enough to generate a signal to respond to. EIT technologies can create new interaction interfaces and make older interactions more accessible, because the movements required to interact with an EIT based system would be much less intricate. This research looks to create a novel method of interacting with EIT based devices, moving the interaction to a mobile medium in through a mobile API. By using a mobile device and focusing on interactive applications, more specific features can be implemented and are explored in this paper.

Table of Contents

Table of Figures

1 Introduction

Electronics have made our lives easier since their inception. However, as the time went on, one of the primary motivations in technological design has become more emphasized: user intuition. Making experiences that are not intentionally new, but are more adaptive to user intuition, are what many would consider the mark of a great design. Being able to pick up a device, and understand how to use it before reading any documentation, tutorials, or manuals is the goal in many technological designs. From mobile devices to appliances, we have grown accustomed to interactions that make sense (touch screens, voice assistants who you speak to using common words, etc.). Technological experiences are made to be as intuitive as possible, so as to integrate seamlessly with everyday life through natural physical interactions. However, there are many technologies that have not taken full advantage of the physical interactions we can perform. One such technology is seen in Electrical Impedance Tomography. Electrical Impedance Tomography (EIT) measures electrical conductivity and impedance of a part of the body, and reflects its internal changes due to biochemical activities, like muscle movements. EIT devices are traditionally used for medical imaging. However, by transitioning to applications in human computer interaction, EIT can be reimagined [20]. It can shift from being utilized as a tool primarily for imaging, to a form of "invisible input" where users do not have to click a button or flip a switch, but simply make a gesture or flex a muscle to trigger a command.

While this idea of an invisible input has the potential to change the way these EIT devices are used, there are very few people who would be able to take advantage of these advancements, due to the lack of support in regards to developing an EIT system. Some open source solutions have been attempted [13], however, finding enough supporting documentation to understand the software alone can be extremely difficult. Because of this, the project is designed to be easily

compatible with an existing infrastructure [6] that helps users model and plan out electrical components to build and develop on their own EIT systems. By giving users flexibility in choosing constraints for the system, a variety of different EIT bands can be modeled and much more easily fabricated. EIT Kit Mobile removes some of the obscurities that comes with developing similar systems, and decreases time needed to prototype new systems when used in conjunction with EIT Kit [6].

Because the latest developments in open source EIT visualizations are not accessible for mobile devices, it was important to focus on a new medium to process, visualize, and interact with these signals. The API creates a framework for understanding and developing EIT based applications, and lowers the barrier of entry for future users. The API also creates new methods of connection EIT band interactions with different types of technology. The motivation behind this research is to make interactive technologies more intuitive, and make interactions between users and their digital environments more straightforward. Finally, an element of modularity was very important, so that these new methods of interacting with EIT systems could be further developed, improved, and refined over time.

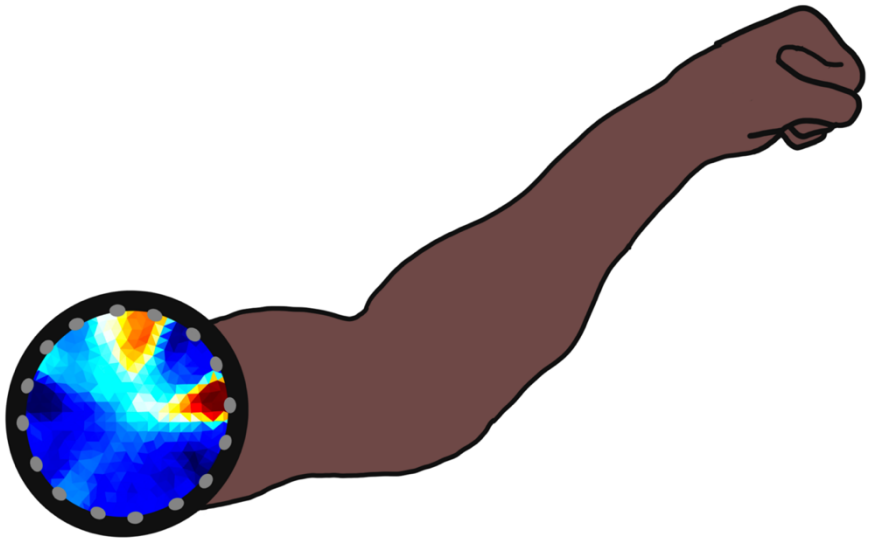


Figure 1 Visualization example: Cross sectional area frame of user making a "Fist" gesture.

2 Background

This work builds on EIT Technology, mobile app development, and integration of hardware types to improve upon a new human computer interaction. This section discusses the history of EIT Technology, and how it informed and motivated this research.

2.1 EIT Overview and History

Electrical Impedance Tomography (EIT) reads the conductivity and permittivity of the human body, reading electrical impulses generated as a response to one's physical behaviors and movements. Traditionally, this information has been used specifically for understanding the body of a patient and providing visualizations of the inner workings of the human body [4]. Examples include understanding brain activity [5], diagnosing cancerous regions in the body [23], and understanding lung function [10]. However, with the advent of improvements in cost efficiency, as well as more generally accessible types of technology, use cases for EIT based systems have greatly expanded beyond the multi thousand dollar setups used exclusively for medical rehabilitation and diagnosis. One of the most popular applications of this technology as it pertains to human computer interactions involves the reading of electrical impulses of the wrist, by using a wristband containing a number of electrodes that are used in order to understand hand gestures and motions [20]. The basic idea of using an EIT wristband consisting of electrodes mounted on the wrist has been done multiple times, by multiple sources. Some focus on hardware, creating the optimal electrode layout pattern of an EIT system to increase detection accuracy and reduce detection time [11], or looking into creating similar systems that use conductive thread to bypass direct electrode usage [7]. Others focus on creating an overall system that works to be compact and modular, fitting on a smart watch [20].

In more recent times, there have been steps towards making EIT more accessible. Image reconstruction libraries such as pyEIT [9] and EIDORS [1] have worked to create a method for EIT visualizations, making it easier for users to understand EIT devices. However, these systems are all computer based. In an increasingly mobile world, our primary interface for interacting with technology is through mobile devices. Our API will take advantage of this, working primarily on mobile devices in order to allow users to develop specific applications using EIT technology.

2.2 Past Implementations of EIT Software

EIT sensing software has existed in an open source capacity since at least 2002 [1]. There are two popular open-source iterations of EIT Sensing software: EIDORS and pyEIT. EIDORS is the solution that has been most commonly used in open source application, especially in regards to data visualization, based on the fact that it was written in MATLAB, and has been operated and maintained for two decades. PyEIT is much newer, created in 2016, and focuses more on modern approaches to EIT visualization, as it is written in Python. This has led to expansions based on the original project [13] and has an active community of developers working to continually add features. The API developed in this research, EIT Kit Mobile, pulls aspects from both of these implementations, taking parts of the visualization from EIDORS, and processing logic in similar ways as pyEIT.

2.3 Use of EIT Sensing Bands

The research expands EIT applications in two facets. Firstly, the API looks to utilize the readings generated as a result of using multiple EIT bands on an appendage. Using multiple bands [8], or conductive materials arranged in a plane instead of a singular band [22], will allow

for a much higher degree of freedom when using an EIT device, as more information about the user will be able to be decoded. Users can potentially use their own body surface area as an interactive touch screen, where presses and touches are mapped through the EIT device, or even developing a limited version of motion capture that does not rely on a full suit to work efficiently.

Secondly, our implementation will study the impact of using multiple EIT bands. There are many types of implementations that have been created using one EIT band, especially in HCI. Between gesture recognition for smartwatches [20] comparing performance between sensing methods [21], and optimization of EIT band hardware [15], single band readings are used constantly and without much consideration of alternatives. A relatively limited number of previous works have looked at using 2 bands, with a different method of data processing [17] and a lack of real time input tracking. This research also looked to improve upon speed of data acquisition and study the effects of multiple EIT bands (more than two) being used simultaneously. By increasing the amount of information gathered, more minute details can be expressed, with less effort on the end of the user, besides wearing multiple wristbands.

Besides only working on computers, other visualization software such as pyEIT and EIDORS focus primarily on single band interfaces. Our API will work to expand this perspective, working with as many bands as the user needs, and potentially introducing new ways to visualize the bands as well.

2.4 Expansion of EIT Interface Areas

This research looks towards the implementation of EIT bands in more areas than just the wrist. In the past, EIT bands on other areas such as around the torso have been studied [4]. Other

have been able to create EIT interface devices, by moving the electrodes off of the body [16]. However, many of these studies have not focused on the potential for simple human computer interactions. In this study, the effect of using EIT to adaptively change an electronic device state based on input generated from the body of the user will be focused on much more heavily. Since EIT can consistently work on various body parts [10], relying on changing cross sections of conductivity, and this conductivity can be read based on human motion, most parts of the body can be read and understood more through the use of an EIT device.

3 EIT Kit Mobile API Overview

EIT Kit Mobile is a tool, in the form of a Swift Package, that allows users to develop apps for iOS and PadOS that utilize EIT technology for human computer interaction, instead of the more traditional medical imaging purpose. By understanding what the system does and how EIT Kit works, users should be able to manipulate the visualizations to their content, and implement a variety of visualizations as needed.

Installing and using the EIT Kit Mobile API is a very straightforward process. Upon importing the package, users can view example view controllers to see how the process is implemented. If manual setup is required, users can simply set up the mesh by calling mesh_setup, and defining parameters as needed, including shape of the mesh, number of electrodes, and number of terminals. After calling mesh_setup, users will have access to the three objects needed to get the mesh visualized: an EIT object which acts as a solver, reading the input conductivities and solving for the visualization output, and the objects that make up the mesh; one that defines the points in the mesh, and another that explains how to connect those points to

form the triangular structure. By calling mapSingle, the solved EIT information is paired with the mesh information to “fill in” the mesh, and the API returns an object that can be displayed on the app.

At this point, the user has a 2D, single band visualization. However, with a few more built in functions, even more information can be read. Users can call mapMulti3D in order to visualize the information in a 3D manner, however, even here, there is choice. Users are returned a three dimensional object that is able to be visualized either entirely on the phone, or in the “real world” through AR. By doing this, users can have insight into the cross sectional volume behaviors of an object wrapped in two EIT bands, in real time. Finally. By looking at the example view controller, users can find a simple method for implementing a basic machine learning algorithm that can be trained to recognize specific types of processed data. Users can save images generated by the app (also included in the example), and use the saved images to train a machine learning model. In this research, training was performed through createML, an Xcode extension that allows for users to simply drag and drop images to train an image classifier. After training the model in createML, users can download the model, drag it into their project, and after appropriately naming it, should be able to see the category and confidence of predictions made by the machine learning model.

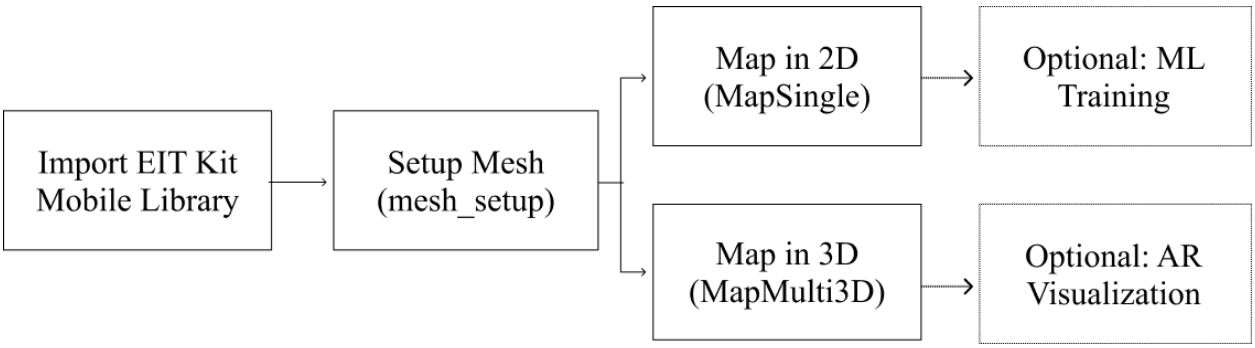


Figure 2 Basic Implementation Flowchart

4 Example Application Integration

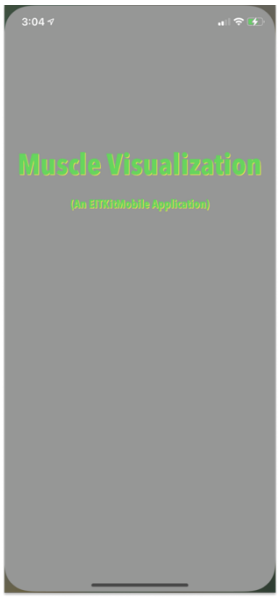
In this section, an overview of five apps that implement various aspects of the EIT Kit Mobile package is shown.

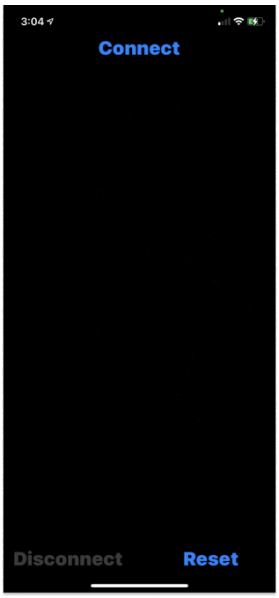
4.1 Muscle Visualization

This app uses the multi band approach to create a 3 dimensional, AR ready visualization. Using the EIT Kit Mobile Library, the mesh is generated on app open, and users have one interface to connect to the EIT Kit Device, or disconnect from it. Once connected, the EIT Kit Mobile Library multi band visualization tools are utilized, in order to create the 3D visualization of the bands, and overlay the model of choice for a user, in order to show activation areas within the model.

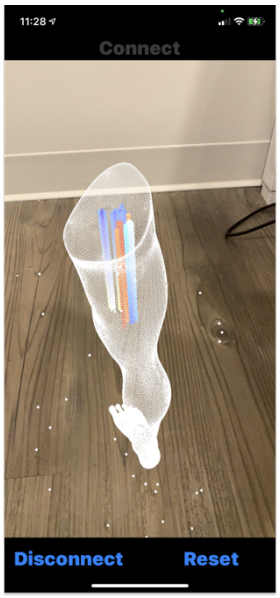
This app works with the EIT Kit Mobile API by utilizing the 3D visualization section of the API. Users wear two EIT bands, sending information from two cross sections of the thigh. The API's 3D visualization tool works to show relevant parts of the thigh, instead of the whole circle. It does that by calculating the average level of activity in the thigh, summing the collective change in conductivity, and dividing it by the number of zones of conductivity in the visualization. Once that is done, users can define a threshold over which information will be considered statistically relevant, and a low threshold based on the average as well, in order to gather information about the lowest and highest areas of activity.

Finally, in this example, the visualization is contained within a wireframe model of a leg. This is done in order to simulate what the bands on the leg of the user are recording. Users can see the changes in conductivity in their leg, in real time. The steps for how to use this app, along with example photos, are shown below.





Step 1: User points the camera to where they want the visualization to be set, and hits connect.



Step 2: User sees a 3D visualization of their leg, based on the bands connected to their leg.

Figure 3 How to set up the Muscle Visualizer app

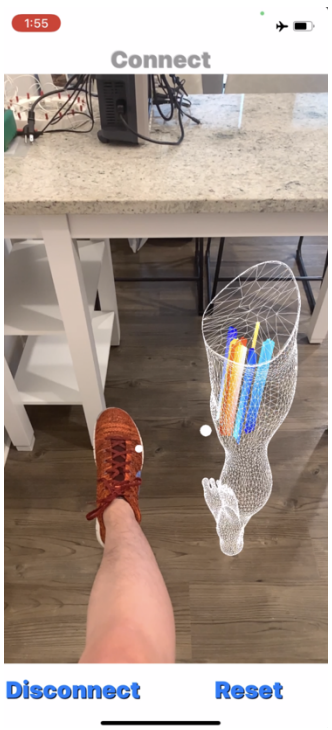
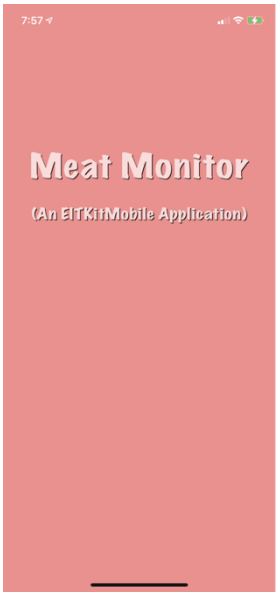


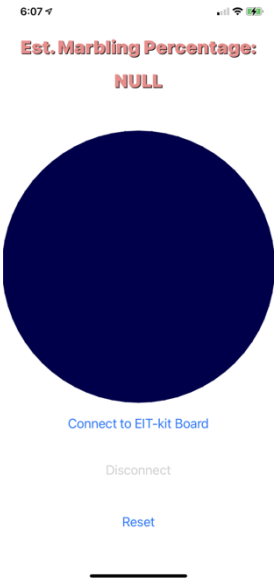
Figure 4 Example use case of Muscle Visualizer app

4.2 Meat Monitor

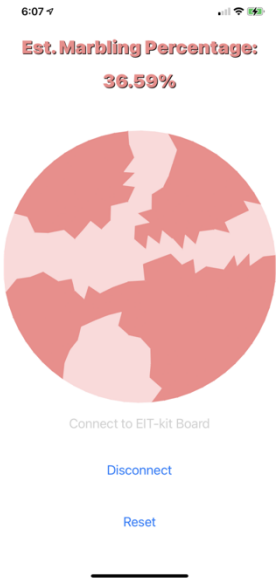
This app utilizes the EIT Kit Mobile Library in a very unconventional way. To begin, the mesh is setup as expected, primarily using the mesh setup function. Then, a modified version of one of the functions defined in the API (mapSingle) is used, where, instead of applying colors based on a gradient of values for a continuous appearance, a discrete version of the method is implemented. Numbers are snapped to one of two colors based on their conductivity. If the conductivity falls below a certain threshold, it is colored as fat (a light pink), otherwise, the meat is seen as lean (a darker pink).

Finally, while rendering each frame, each triangle in the mesh is counted, accounting for if it falls above or below the fat threshold. If the color range of a given mesh falls under the category of “fatty” the number of fat regions in the meat is incremented by one (since the triangles are all roughly the same area, this method is relatively unbiased with regard to orientation of the electrodes), until finally, the render is shown on the screen, along with an estimated marbling percentage, which is calculated based on the proportion of the area of the fatty sections of the meat compared to the entire cross sectional area of the meat. The steps that go into these interactions, as well as in app screenshots, are shown below.



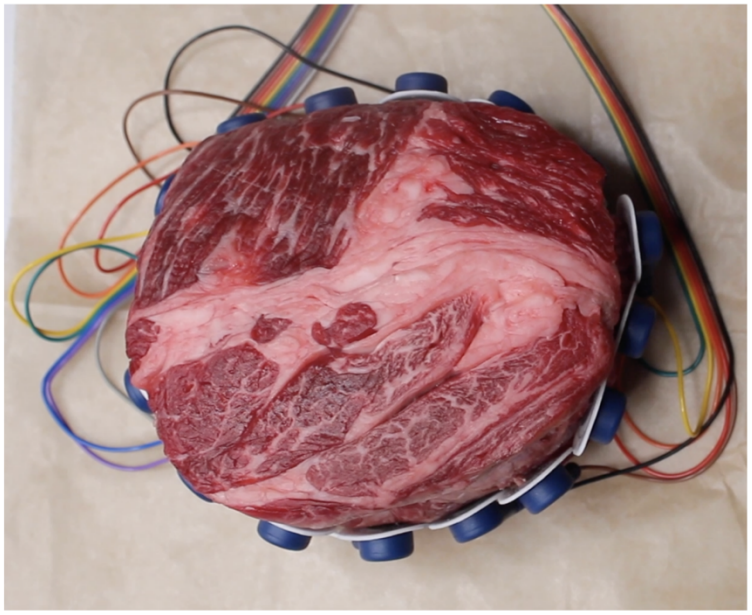


Step 1: User opens the app, and hits connect to connect to the EIT Kit Board



Step 2: User sees a live, 2D cross sectional area of the meat, with an estimated marbling percentage.

Figure 5 How to set up the Meat Monitor app



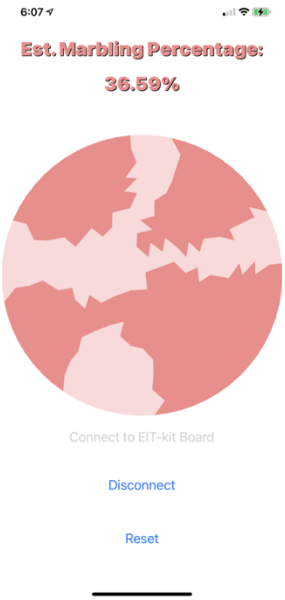


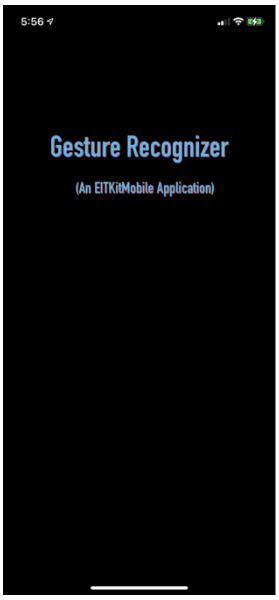
Figure 6 Example use case of Meat Monitor app

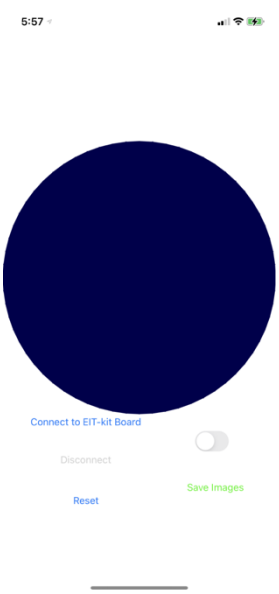
4.3 Gesture Recognizer

This app is a much more traditional use of the EIT Kit Mobile Library. Users have a single interface for training and running a machine learning option. Upon opening the app, the mesh is created, and on connecting to the EIT-Kit Device, the mesh fills with the information on conductivities read by the system.

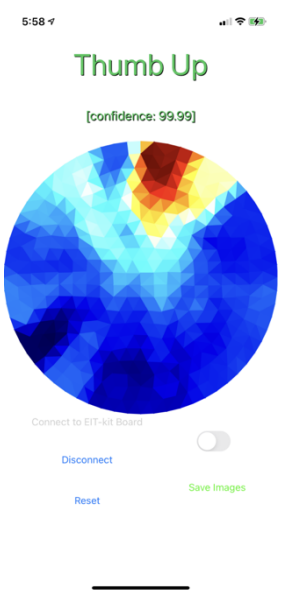
Where this app differs is in how it uses these visualizations. While the previous examples focus on allowing the user to see the data, and drawing basic conclusions from it, or extrapolating the information in order to create a more advanced, three-dimensional version of the visualization, this app process the data to a much more significant extent. This is done by allowing the user to save each frame as it is displayed, saving the images directly to the Photos app on the iPhone. From there, users can upload these images to the CreateML app, that comes built in on MacOS, and train their own machine learning model simply by using drag and drop commands.

Once the model is done, all the user has to do is download it, drag it back into the Xcode project, and re-run the project. Once the user re-runs the project, the app will use the machine learning model in order to predict which gesture the user is making, based on the model the user just created. An example of a six-gesture classifier, along with in app screenshots, are shown in the following figures.





Step 1: User opens the app, and hits connect to connect to the EIT Kit Board.



Step 2: User sees a 2D cross sectional visualization of the muscles in their arm, and can use ML to recognize gestures.

Figure 7 How to set up the Gesture Recognizer app

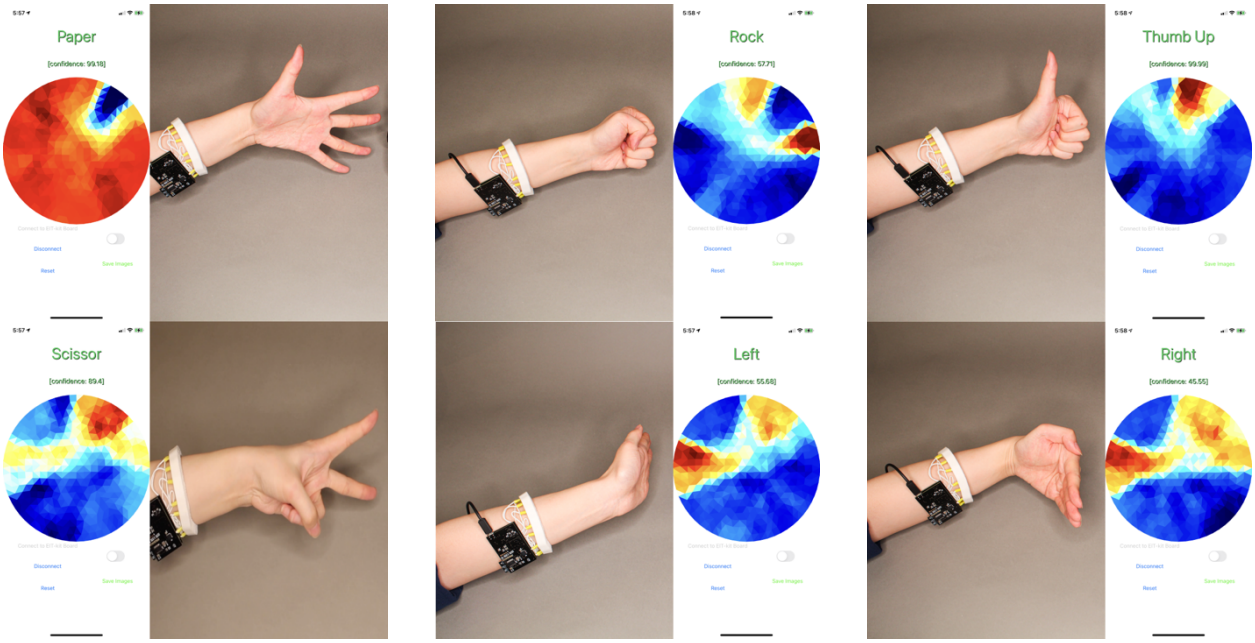


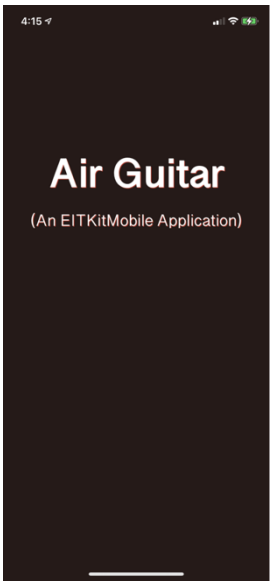
Figure 8 Example use case of Gesture Recognizer app

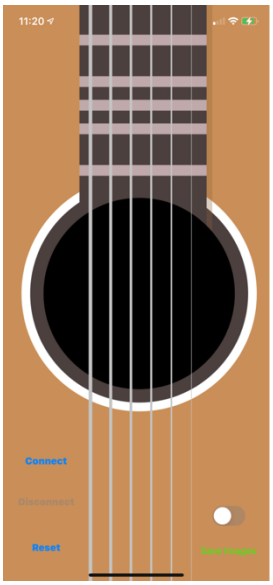
4.4 Air Guitar

This app uses a similar premise as the gesture recognizer, with a bit of a fun twist. Users are able to run the methodology, train the ML model, and upload it in the same way as they do with the Gesture Recognizer app. However, when the machine learning algorithm is uploaded, users now have another way to interact with the app: through sound files. In this iteration, users can match sounds to the gestures they make. For example, a “Fist” gesture can be interpreted as a G chord, and an “Open Hand” gesture can correspond to a C chord.

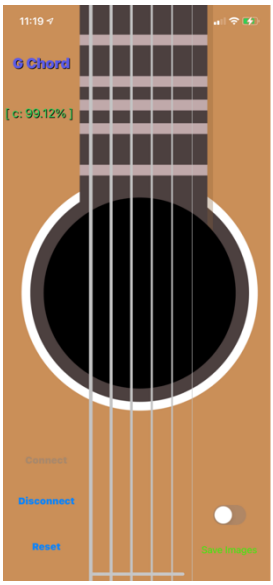
Once users train the ML model and upload it, the only steps the users need to follow in order to play the chord are as follows: The user makes the gesture, and “strums” the string on the guitar, by swiping their thumb across the screen. The intention for the app is recreational, but showcases the various ways that the EIT Kit band can be used for accessibility. The ease of use of the system, combined with the fact that multiple gestures can be loaded into the system and triggered with a simple screen motion, highlight the most promising part of the potential of the library.

The ability to quickly and effortlessly change commands as long as users are able to make large, discernable differences in hand motions, takes away from the smaller, more discrete taps and swipes that users usually have to perform, which can be extremely difficult for certain users, especially those who struggle with fine motor control.



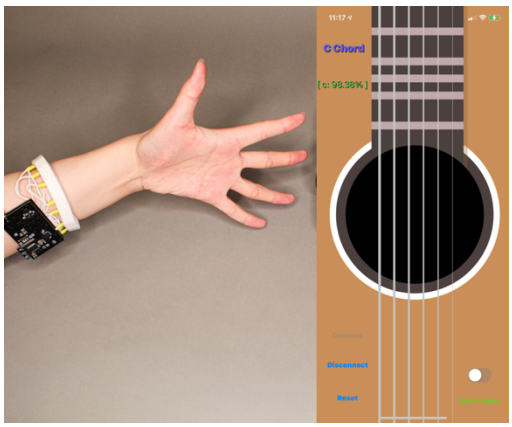


Step 1: User opens the app, and hits connect to connect to the EIT Kit Board. Here, they can toggle the Save Images feature to train an ML model.



Step 2: User makes gestures to launch different chords, and strums the strings by swiping across the screen on the visualization to play the chords.

Figure 9 How to set up the Air Guitar app



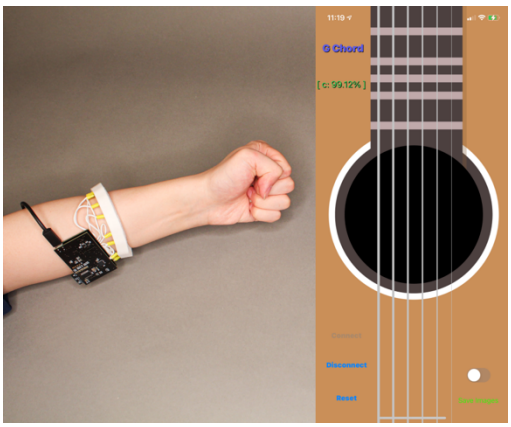
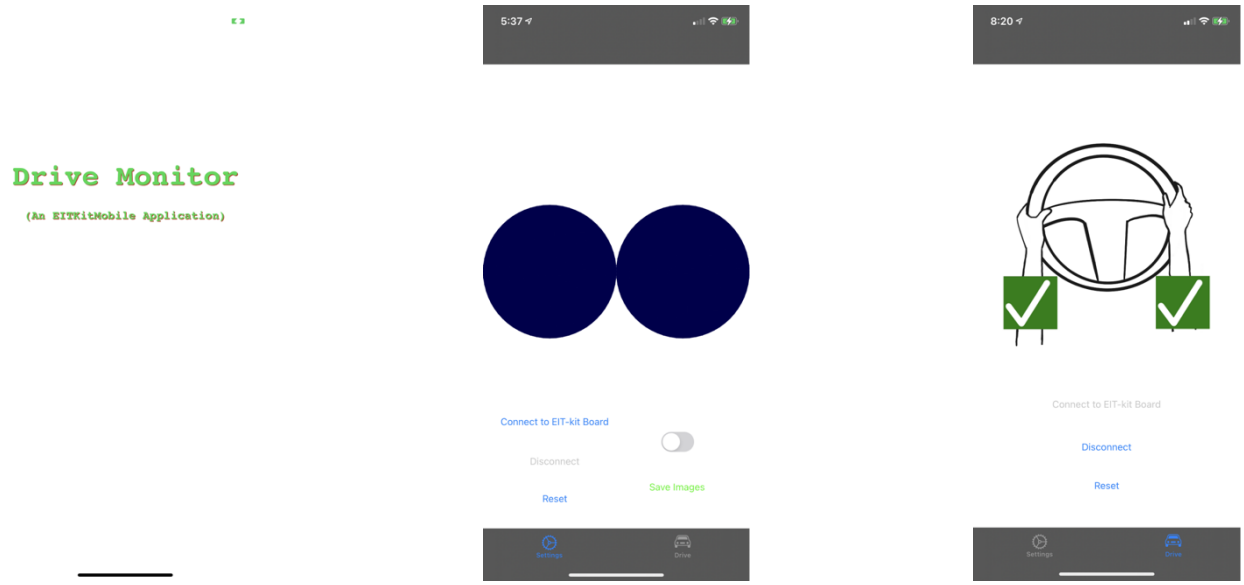


Figure 10 Example use case of Air Guitar app

4.5 Drive Monitor

This app implements the multiband EIT approach in an unconventional way: by having the user wear the bands on different arms. Users can train a machine learning model for each arm, or an individual machine learning model for both, in the “Settings” section of the app, and apply that model to the “Drive” section of the app. Once the visualization is trained and the user is in the “Drive” section of the app, there are four possible visualizations, representative of if the machine recognizes the gestures of having both hands on the wheel, one hand, or neither.

Because both arms are being read at the same time, the multi band visualization functions of the EIT Kit Mobile Library are being utilized for the training visualization, and the gesture recognition as well. However, the multiband hardware currently alternates the sending of data (starting with the left at runtime), so the app does the same, only updating one side of a reading at a time. By increasing the sampling rate, it may be possible to get the visualization to a point where it seems as though the updates are happening at nearly the same time. Alternatively, the user could modify the code such that it connects to two separate EIT Kits, receiving, parsing, and display the data for both at the same time. An example using the single EIT Kit approach is shown below.



Step 1: User opens the app, and hits connect to connect to the EIT Kit Board.

Step 2: User moves to Drive tab, connects to the EIT Board, and runs the ML model they generated, getting an indication of hands on the wheel.

Figure 11 How to set up the Drive Monitor app

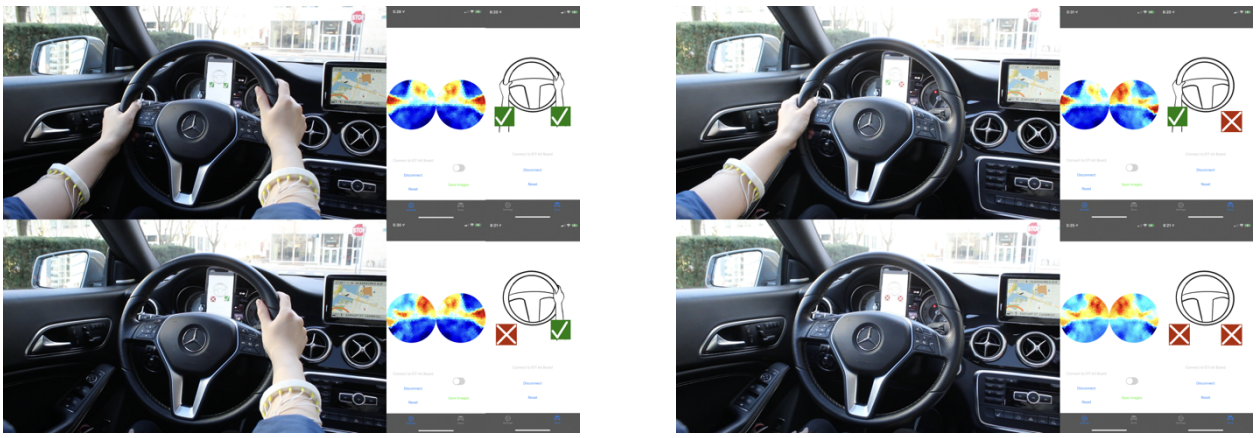


Figure 12 Example use case of Drive Monitor app

5 Technical Implementation

EIT interfaces work by outputting a small current between two electrodes attached to a line of electrodes, (often a power of two) on a band, attached at the wrist, thigh, or anywhere where a user would want to have these measurements read. The rest of the electrodes, that is, those that are not outputting current reading the reported voltage drop between every other combination of electrodes in the system. In this way, we are able to receive large numbers of readings between electrode pairs per frame of information, which allows for a relatively high degree of specificity in reading these electronic signals. These signals in turn correspond to muscular activations in whatever region of the body the band is on. In this section, the process of moving from input data (raw readings from the electrodes) to a visualization is described.

5.1 Inputting the Data

The EIT Kit Mobile API works first by receiving inputs. There are three methods of generating or receiving these inputs that are built into EIT Kit Mobile. First, users are able to send inputs from a reading directly to the application. Because the EIT Kit needs a frame of reference, data is expected in the following format: one reading per line, with clear demarcations indicating the origin frame, intermediate frames, and the end of the file. These demarcations should simply be the words “origin frame”, "frame”, and “end of file” respectively. EIT Kit will work to parse this information, treating it as live readings, and allowing the user to visualize each frame as slowly or quickly as the user may like.

Second, users can create new data to be parsed using EIT Kit Mobile directly. Similar to how the system works in pyEIT, users are able to define permeation objects, strength of permeation, and their location. Essentially, users can simulate putting simple geometric objects

into a “test field”, and view the simulated visualization as though the data were live. Mathematically, these visualizations are created through a FEM Forward solver [19]. While EIT Kit mobile currently supports static images in this method, users should be able to animate these motions as well, simply by feeding in different permeation locations, generating a number of frames, and running them one after another.

The previous implementations focused on simulated data and prerecorded data. These ideal conditions are not always the case, however, and the tool would not be complete without being able to create visualizations of real time data. For this, EIT Kit Mobile currently uses Bluetooth, since it was the fastest, most reliable wireless form of communicating with the EIT Kit band. Users can define Bluetooth peripheral and characteristic IDs, connect to the hardware, and parse information as it comes in. In order to ensure maximum throughput from the Arduino to the phone, certain assumptions are made. All information is assumed to be 6 characters long (typically one digit followed by a decimal point, and 4 significant figures after the decimal). Also, information is expected to come in on one line, so as to not use more characters than necessary. Finally, the divisions between frames were also expected to be 6 characters. “origin” marks the beginning of the origin frame, “framei” (frame intermediate) represents the beginning of an intermediate frame, and, if necessary, “framef” (frame final), represents the end of the transmission from the Arduino. While more information could be sent by decreasing the number of significant figures and the length of demarcations, the system as it stands was able to send approximately 10 frames per second consistently, using an iPhone XR.

5.2 Creating the Mesh

After receiving the data, functions in the package work to process it. First, users must define the number of electrodes that make up the EIT Kit band, so that the mesh created for the

information can be filled in properly. The mesh can have varying degrees of specificity, based on the number of electrodes on the band. Next, users must define the shape of the mesh. Users can currently choose between two mesh shapes: Circular or Elliptical , based on whichever fits the electrode arrangement best. Finally, users can choose between two or four terminal data collection, which determines the number of electrodes measuring the conductivity changes, versus the number of electrodes that are producing current to be measured. The mesh is generated through a modified version of a similar procedure, originally implemented in MATLAB [14] now translated to Swift. The mesh consists of two parts, a list of points in the region of the mesh, and a list of 3 element arrays, where each array defines the three vertices of a triangle. These triangles in turn are what are primarily displayed to the user

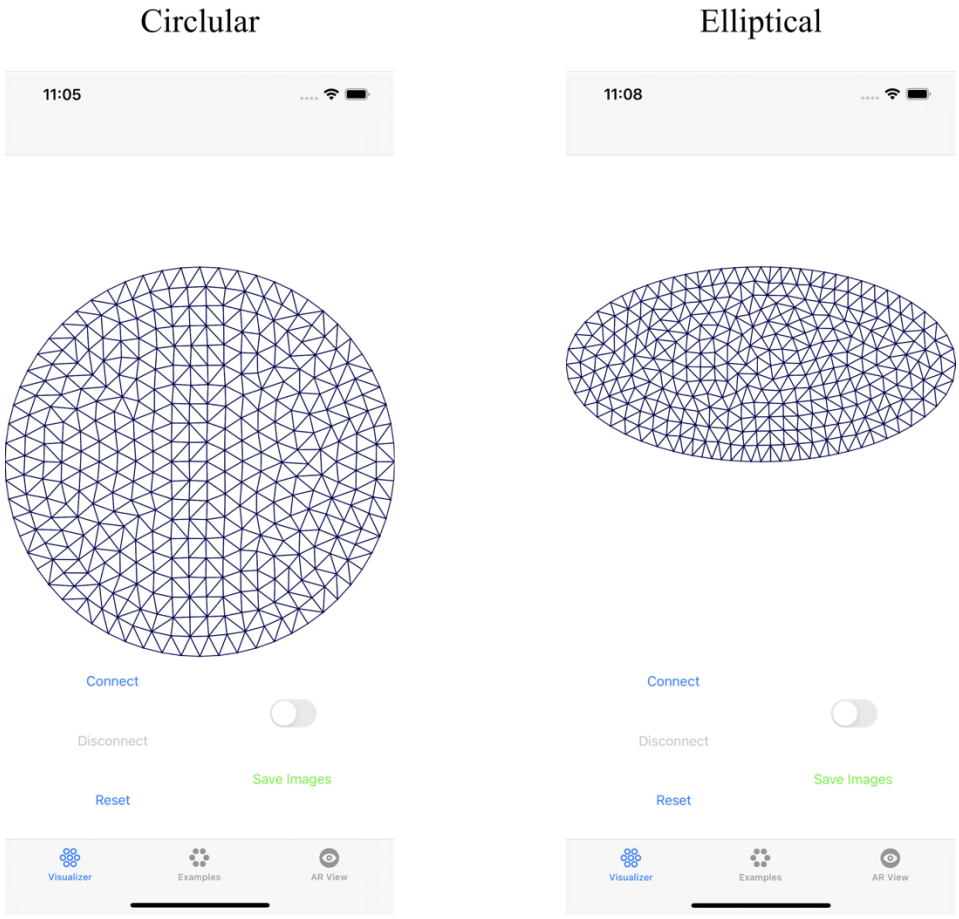


Figure 13 Currently available shapes for mesh generator

5.3 Filling the Mesh

Finally, the mesh is filled, using the generated information after being passed through the EIT solving function. Using an optimized version of back projection [18], the system solves for conductivity at every triangular vertex of the mesh, and the visualization fills each triangle with the average conductivity values of each vertex in the triangle.

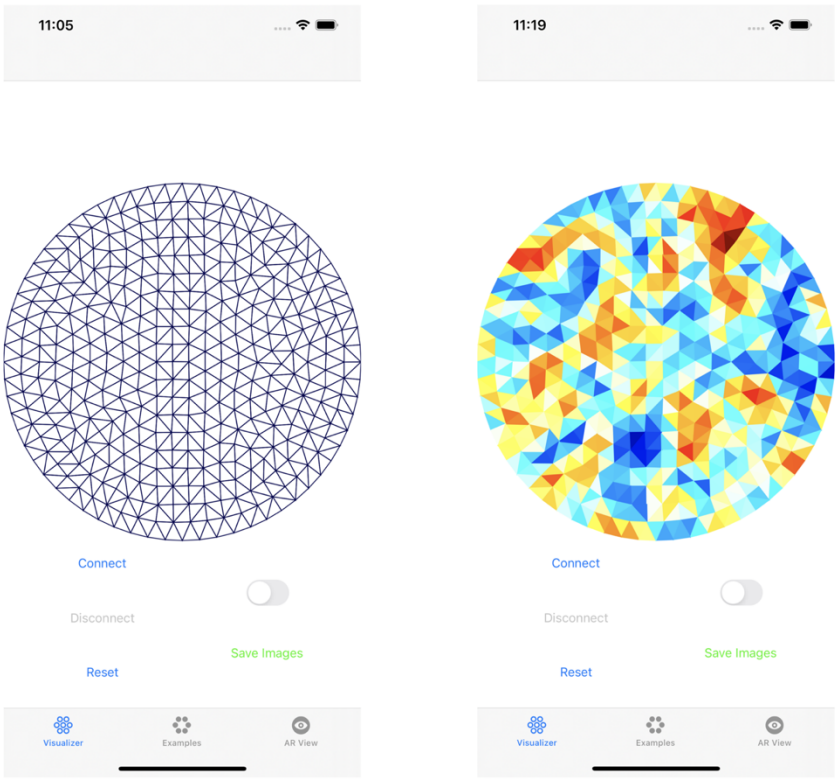


Figure 14 Example of mesh generation, followed by filling in mesh with example data.

5.4 Using Multiple Bands

The visualizations shown up to this point have all been single band and two dimensional. However, EIT Kit Mobile also supports three-dimensional, multi-band visualizations. These visualizations are accomplished by reading the information from two bands, processing each separately as a 2D visualization, and cross referencing each visualization to find areas of high

and low activity, determined by conductivity thresholds set by the user. Because a 3D visualization with a full cylinder would not be as helpful to the user (as only the outer ring of conductivities could be seen from the side of the cylinder), the high and low areas are given full opacity, while areas of little to no deviation from the average conductivity are made transparent. Users are able to dynamically adjust this value, simply by raising or lowering the threshold that determines how far from the average conductivity a value has to be to be considered relevant. Users can choose to visualize this information completely on the phone, or through AR, simply by choosing the proper View Controller.

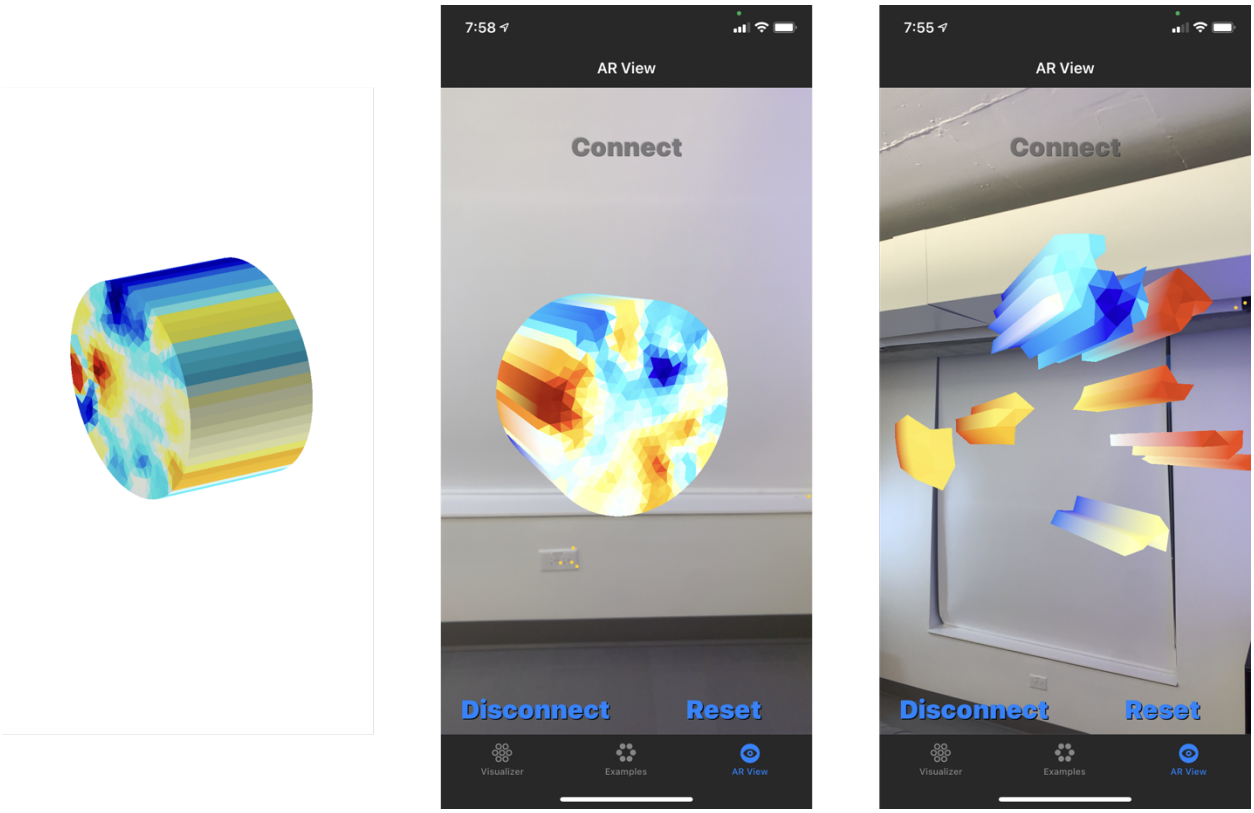


Figure 15 3D Model of Visualization, 3D AR View, and Filtered 3D AR Visualizations

5.5 Implementing Gesture Recognition

Finally, users have the option of including a machine learning component in their visualization. By using the example view controller script included in the library, users can save images of their visualization frames, send those images to their computer, and train a Machine Learning model to recognize these images. This way, users have the ability to recognize the gestures being performed as they are being performed in real time (an example of this is shown in the Gesture Recognizer app, in the Example Application Integration section below).

6 Evaluation

This project has successfully demonstrated the ability to package the EIT interface in a much smaller package. The condition of success was determined by inspection. Evaluations were performed visually, by placing plastic objects in a saline solution to create nonconductive areas in a conductive pool, wrapping an EIT band around the pool and running visualizations in EIDORS and using the EIT Kit Mobile API in order to compare the results (Comparison image on next page).

Upon visualizing the same inputs, it was clear that the implementation of the solvers was similar but not the same, which was expected, as methods of solving were slightly different (different solvers were implemented based on approaches), visualizations did not use exactly the same scaling for colors (as the gradients that were used for EIDORS and pyEIT are not readily available for use in Swift), and libraries used due to different languages led to different results.

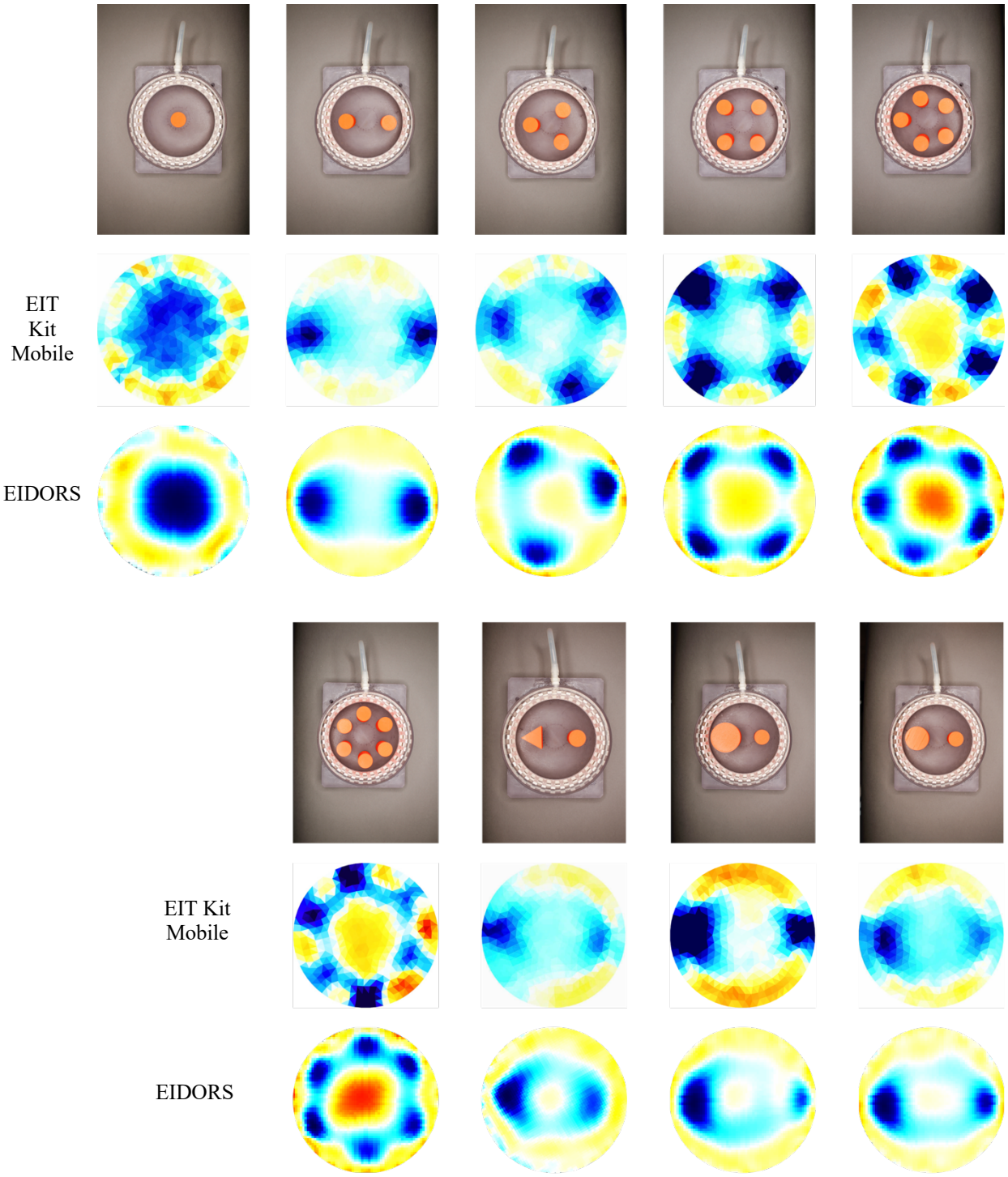


Figure 16 Real world data simulated in EIT Kit and EIDORS

7 Discussion

In this section, an overview of future plans and other possible application mediums of EIT Kit Mobile are discussed.

7.1 Compatible Devices

The initial implementation of EIT Kit Mobile was developed while testing specifically in iOS. However, as the final implementation had the form of a Swift package, there is a very strong likelihood that with minimal work, these functions could be repurposed for PadOS and macOS. As a proof of concept, a simple visualization was created on PadOS, simply by changing the device being used to simulate an existing example application, and the only change in code needed was one to fix the formatting of the display. In the future, developing for watchOS could be very convenient, as having the visualization on a wrist would be informative and feel intuitive, and building an EIT Kit band into an Apple Watch band would be a multipurpose, powerful band similar to the Tomo band [20].

7.2 Future Work

Aside from the implementation of the EIT Kit Mobile package on watchOS, future work also involves including different types of solvers, such as DBAR [12], and CEM [2]. Implementing different solvers will allow the information to be visualized in different ways, and there may be a more optimal way to calculate a basic form of the visualization on the iPhone. Also, in line with giving users more unique ways to solve an visualize their EIT problems, work can be done in implementing new boundary types, beyond the circle and ellipse. Currently, implementing shapes takes the form of defining distance functions, and the code could be

modified to accept functions that define more abstract shapes, or even user generated shapes. Finally, there are limitations that could have potential solutions in future software optimizations. Due to the complexity of the calculations, as well as the processing capabilities of the iPhone, the speed of processing may be able to be optimized to run at a higher framerate, and the Bluetooth communication protocols may be able to be improved in a similar manner.

8 Conclusion

This research explored the novel implementation of EIT as a Human Computer Interface system by implementing EIT Kit Mobile, a mobile enabled API to allow users to develop and interact with EIT based devices from mobile devices. Then, example applications were created using the API to explore different purposes, use cases, and modification techniques for the API. Finally, the success of the implementation was evaluated through visual inspection of previously implemented EIT visualization software.

References

[1] Adler, A., and W. R. Lionheart. “Uses and Abuses of EIDORS: an Extensible Software Base for EIT.” *Physiological Measurement*, vol. 27, no. 5, 2006, doi:10.1088/0967-3334/27/5/s03.

[2] Bahrani, N., and A. Adler. “2.5D Finite Element Method for Electrical Impedance Tomography Considering the Complete Electrode Model.” *2012 25th IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*, 2012, doi:10.1109/ccece.2012.6334971.

[3] Borrego-Carazo, J., et al. “Capacitive-Sensing Module with Dynamic Gesture Recognition for Automotive Applications.” *2020 23rd International Symposium on Design and Diagnostics of Electronic Circuits & Systems (DDECS)*, 2020, doi:10.1109/ddecs50862.2020.9095748.

[4] Cheney, M., et al. “Electrical Impedance Tomography.” *SIAM Review*, vol. 41, no. 1, 1999, pp. 85–101., doi:10.1137/s0036144598333613.

[5] Holder, D. S. “Electrical Impedance Tomography (EIT) of Brain Function.” *Brain Topography*, vol. 5, no. 2, 1992, pp. 87–93., doi:10.1007/bf01129035.

[6] Junyi Zhu, et al. “EIT-kit: An Electrical Impedance Tomography Toolkit for Health and Motion Sensing” *2021 Proceedings of the 34th Annual ACM Symposium on User Interface Software and Technology (UIST ’21). ACM*, 2021. (Under Review)

[7] Ku, P., et al. “ThreadSense: Locating Touch on an Extremely Thin Interactive Thread.” *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, 2020, doi:10.1145/3313831.3376779.

[8] Lim, C. G., et al. “MuscleSense.” *Proceedings of the Fourteenth International Conference on Tangible, Embedded, and Embodied Interaction*, 2020, doi:10.1145/3374920.3374943.

[9] Liu, B., et al. “PyEIT: A Python Based Framework for Electrical Impedance Tomography.” *SoftwareX*, vol. 7, 2018, pp. 304–308., doi:10.1016/j.softx.2018.09.005.

[10] Lymperopoulos, G., et al. “Applications for Electrical Impedance Tomography (EIT) and Electrical Properties of the Human Body.” *Advances in Experimental Medicine and Biology*, 2017, pp. 109–117., doi:10.1007/978-3-319-57348-9_9.

[11] Ma, G., et al. “An Optimal Electrical Impedance Tomography Drive Pattern for Human-Computer Interaction Applications.” *IEEE Transactions on Biomedical Circuits and Systems*, 2020, pp. 1–1., doi:10.1109/tbcas.2020.2967785.

[12] Mueller, J. L., and S Siltanen. “The D-Bar Method for Electrical Impedance Tomography—Demystified.” *Inverse Problems*, vol. 36, no. 9, 2020, p. 093001., doi:10.1088/1361-6420/aba2f5.

[13] OpenEIT. “OpenEIT/EIT_Firmware.” *GitHub*, github.com/OpenEIT/EIT_Firmware.

[14] Persson, P., and G. Strang. “A Simple Mesh Generator in MATLAB.” *SIAM Review*, vol. 46, no. 2, 2004, pp. 329–345., doi:10.1137/s0036144503429121.

[15] Russo, S., et al. “Towards the Development of an EIT-Based Stretchable Sensor for Multi-Touch Industrial Human-Computer Interaction Systems.” *Cross-Cultural Design*, 2016, pp. 563–573., doi:10.1007/978-3-319-40093-8_55.

[16] Sato, M., et al. “Zensei.” *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, 2017, doi:10.1145/3025453.3025536.

[17] Suzuki, Y., et al. “Touch Sensing on the Forearm Using the Electrical Impedance Method.” *2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, 2019, doi:10.1109/percomw.2019.8730739.

[18] Wang, H., et al. “An Implementation of Generalized Back Projection Algorithm for the 2-D Anisotropic EIT Problem.” *IEEE Transactions on Magnetics*, vol. 51, no. 3, 2015, pp. 1–4., doi:10.1109/tmag.2014.2361839.

[19] Woo, E. J., et al. “Finite-Element Method in Electrical Impedance Tomography.” *Medical & Biological Engineering & Computing*, vol. 32, no. 5, 1994, pp. 530–536., doi:10.1007/bf02515311.

[20] Zhang, Y., and C. Harrison. “Tomo.” *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*, 2015, doi:10.1145/2807442.2807480.

[21] Zhang, Y., et al. “Advancing Hand Gesture Recognition with High Resolution Electrical Impedance Tomography.” *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, 2016, doi:10.1145/2984511.2984574.

[22] Zhang, Y., et al. “Electrick: Low-Cost Touch Sensing Using Electric Field Tomography.” *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, 2017, doi:10.1145/3025453.3025842.

[23] Zou, Y., and Z. Guo. “A Review of Electrical Impedance Techniques for Breast Cancer Detection.” *Medical Engineering & Physics*, vol. 25, no. 2, 2003, pp. 79–90., doi:10.1016/s1350-4533(02)00194-7.