

MIND x HAND: Immersive Multimodal Meditation

Ziyuan Zhu | Aikaterini Lamprou

How might we make the meditation experience more customized and personalized for amateur meditators? How can meditation lovers conduct meditation in a more mindful and insightful way? Through this project, we want to **explore the new possibility of combining gesture and voice control to enhance meditation experience and introduce a more fun and immersive experience to meditation beginners.**

Through observation of the daily meditation moment of ourselves and close friends, we saw that different gestures are used when meditators move into a different phase of meditation, which is known as “mudras” and contributes to the symbolic meaning of feelings in the meditation process. Considering the meaning of different hand mudras and the potential of combining it with multimodal interaction, we started to imagine a more adjustable meditation assistant which can help improve the personal meditation experience with meaningful gesture and voice control.

The outcome is a multimodal application which enables users to utilize voice and gesture to control the meditation process and customize their own experience. It is, also, an educational tool which introduces mudras to the one who wants to experience more origin and a fun meditation with the mudras. The interface is created using p5.js, and voice control driven by p5.speech.js, with which users can control the meditation interface with natural language to switch to different modes of the meditation. The hand detection function with a webcam, which enables users to pose mudras, triggers customized chime sounds which aim for a customized meditation experience.

The Github Link: <https://github.com/zy-zhu/6.835-final-project-immersive-meditation>

1 Introduction and Overview

Meditation is an inherently multimodal experience, where physical and mental sensations arise, and focus and insightfulness can be driven by visual stimuli or sounds. At the same time, the meditator's state is reflected in their face and body. While an abundance of meditation apps is available in the market, the motivation behind this project was to explore the potential of incorporating multiple modalities in a meditation assistant. While in an in-person meditation session, where a coach is providing live guidance, feedback is an important aspect of the overall experience, in the "automated" case of a phone app meditation needs to be interrupted due to lack of synchronicity with the practitioner's needs at a certain moment.

We designed a multimodal application which enables users (meditation beginner and amateur meditators) to utilize voice and gesture to control the meditation process and customize their own experience with the use of mudras, a gesture system used in the meditation process to enrich the meditators' experience. In particular, to cater the meditator's needs, the system:

- (a) supports a sitting-on-the-ground setup with voice commands and camera recognition of customized gestures;
- (b) offers meditation options which correspond to different visuals and sounds;
- (c) includes detailed learning instruction to introduce mudras to the meditation beginners or mudras beginners;
- (d) provides an instructions pop-up on the mudras which can be called and dismissed anytime through gesture;



Figure 1. Meditation with mudras

Our proposal incorporates solutions to improve the overall experience on multiple levels with voice control and mudras trigger. The application assumes a sitting posture during meditation on a chair or on the ground with the user standing a few feet away from the screen. First, UI navigation improvements are implemented, which allow for remote control through speech. Although a minor addition to conventional UI solutions, remote interactions such as speech synthesis and recognition can save the meditator from having to move their limbs outside of the meditation state. On another level, our proposal seeks to enhance **the meditation learning experience** of the **meditator through visuals and sounds**. Our adjustable meditation assistant takes advantage of different elements in meditation techniques. Meditation practitioners use different techniques, such as having somebody guiding the process, focusing on a particular object, using bowls, gongs, or other instruments to create sound vibrations or, simply, keeping quiet, to bring the mind and body to a more relaxed state. At the same time, different gestures, known as the mudras, are used to guide the flow of focus through the body. Our goal was to take advantage and combine these things to assist the experience. Each mudra has a symbolic meaning, and different hand mudras are thought to reflect changes in the process of meditation. Background sounds could be triggered by gesture recognition and respond to changes in meditation states. To this end, in our system four mudras are recognized (mudras library is expandable) and each one corresponds to a metal instrument sound. A fifth static, semaphoric gesture is used to signify the end of meditation. Interface colors and visuals aim to reflect the meditator's mood, and provide visual elements which can be used to enhance concentration, if required by the user.

2 System Description

The system is intended for both people who want to learn things about meditation and practice the mudras, as well as people who are already more familiar with meditation. It comprises a main functionality and an integrated learning module. The idea of meditation that the interface promotes is one that combines elements from different meditation traditions. At the same time, the concept of selecting different moods - translated into visuals and sounds in the interface - is thought as a powerful way to impact or enhance the current mood of the user. This is why the mood palette includes options for Focus, Joy, Zen and Sleep. These different moods are common in several other commercial concentration or meditation applications.

Two ways of using the app are considered noteworthy. Below we describe two use cases, one learning and one meditating session. The examples we bring, stand in the middle between the ideal use cases, and the ones we experienced by testing our application with non-experienced users. Below we describe two use cases, a learning and a meditation session. The examples we bring, stand in the middle between the ideal use cases, and the ones we experienced by testing our application with non-experienced users.

User A, a meditation learner

The user starts the app with the intention to practice the mudras. Upon loading the main page, the meditation assistant greets the user, and the Mudras Reminder pop-up window shows up on the left of the screen.

The pop-up has multiple purposes:

- (a) introduces the use of gestures to the user,
- (b) teaches them how to call/close the pop-up using gesture at any time,
- (c) supports learning, by showing images of four basic mudras to which the user can refer during meditation.

The first two are directly put in use when one first interacts with the application. To close the pop-up, the user needs to read and understand how it is controlled. Shortly, they are already familiar with using gestures in front of the camera for interaction. For more details on the use of the interface, an Instructions page is available, which can be called from the Main page.

From the Main page the user can pick a mood that suits their preferred meditation experience through speech, or by just clicking on the respective circle on the Mood Palette. They are then directed to the Meditation page, where the background sound and graphics suggest the selected mood. There, the different mudras can be detected and, upon detection, respective chimes come up. To the mudra learner, the metallic sounds operate as rewards for showing correct, recognizable gestures. If they forget a gesture, they can call the Mudras Reminder window on-screen. They can enjoy the experience, or practice the mudras as long as they want. It is possible to end the session by saying intuitive things such as "Nice" or "Namaste" to go to the end page, where they are saluted by the app assistant.

User B, an amateur meditation practitioner

To a person who is aware how to use the mudras, the application finds a different use. The Mudras Reminder, again, introduces the use of gesture controls, and informs the user on the mudras which are recognizable by the application. The user can go to the Instructions page to explore the application capabilities, among which

the possibility to add new mudras to be recognized by the application, and link them to their preferred sound effects¹. All the recognizable voice commands are mentioned in the Instructions. When the experienced user picks a mood for Meditation, they can start meditating directly. There are multiple options for meditation:

- (a) The sound can be kept on or turned off. The volume is adjustable through a slider on the window².
- (b) The user can keep their eyes open or closed. If open, the visual object in the middle of the screen can be used for concentration.
- (c) If mudras are practiced, chimes peel upon changes, signifying and supporting at the same time a deeper state of meditation.

When finished, the meditator can say so, and exit the meditation mode³. It is possible to either exit the app, or return to the Main page, and select a different Meditation mood.

3 System Functionality

3.1 Basic Architecture

The basic architecture of the application consists of voice and recognition, as well as the meditation learning & experience interface. The following diagram shows the system architecture.

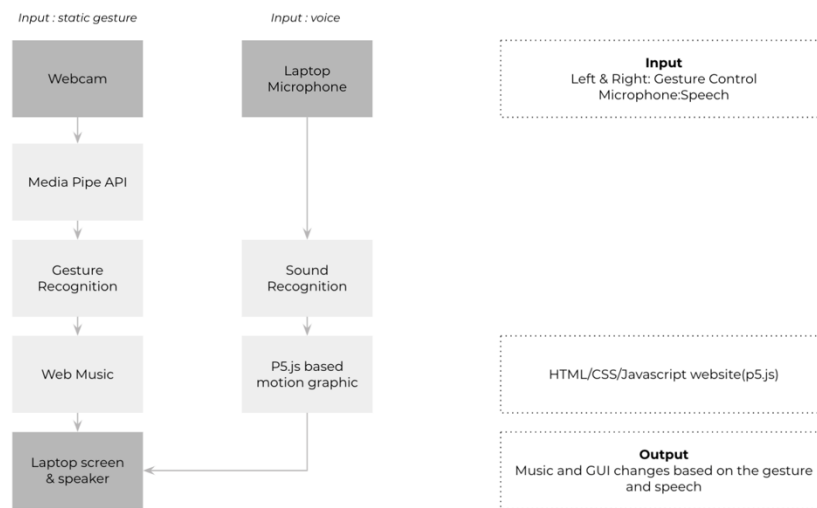


Figure 3. System Architecture of our project

In the following two sections, we will describe the speech recognition and the use of hand gesture recognition in this project.

¹ Unimplemented feature. The final recognition implementation allows to do this easily, although the UI elements which could allow this are not yet implemented. For now, it is part of the concept, suggested as future work.

² Ideally, sound control should, also, be possible remotely.

³ Exit should be possible with the Anjali Mudra.

3.2 Voice Recognition and Interface Design

Several things were taken into account for the user experience design. When the user starts the app, they come across the main page where they are welcomed by the meditation assistant. The user is able to pick a meditation mode based on their mood. Upon selecting a mood, the system shifts to meditation mode. In meditation mode, static gesture recognition is possible. Control of the interface in all modes is possible through speech, keyboard and mouse. An instructions page is provided informing the user on all available controls and interactions. The different components of the interface and their functionality are presented below in more detail.

(1) Main Page

This is the initial screen which shows up upon starting the app (Fig. 3). The user is welcomed by the system, and is asked to choose among the four different modes. Alternative interaction options were implemented for UI navigation. All paths are accessed through speech. At the same time, all on-screen elements are clickable. A button linking to the Instructions is available on the top right.

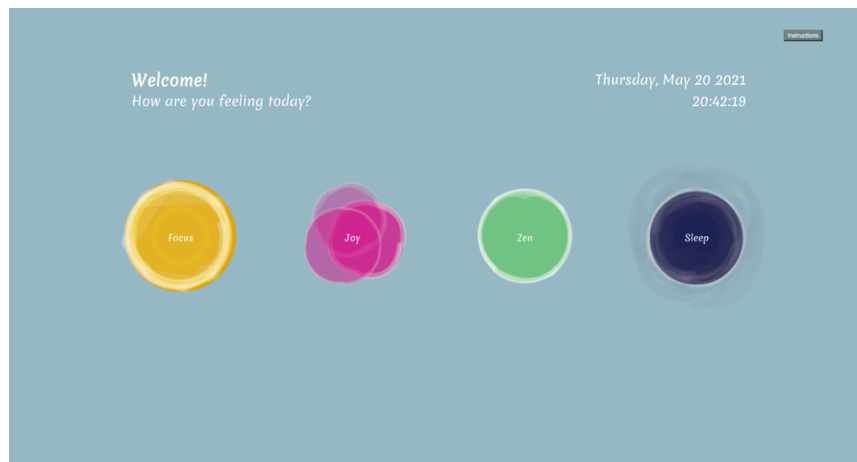


Figure 4. Main page. Users can select the respective meditation mode via speech.

Some extra details regarding the interface design are the following. Date and time information are provided on the screen. The colors of the different moods reflect the main color of each mood-canvas on the respective meditation page. Since the proposed setup for using the application assumes that the user stands a few feet from the screen, the pages are thought of as canvas rather than typical web-pages. The size of on-screen elements is bigger than in typical web-pages, for legibility.

(2) Instructions Page

The user can say “Instructions” or click on the Instructions button to access the page. All the possible interactions and controls of the interface are explained here. Instructions specify the following.

1. The proposed physical setup and where the camera should be placed so that recognition is possible.
2. All the keywords that can be used for interface navigation.
3. Possible keyboard interactions.

4. Meditation Instructions. The names and images of the mudras which can be recognized are shown, proposing the ideal way of conducting the gesture so that it is easier to be recognized by the system. The “stop” gesture, used to exit meditation mode in silence.

All speech interactions with the interface which are specified in the instruction page are illustrated in Tables 1 and 2. Keyboard interactions shown in Table 3 are, also, mentioned in the Instructions. Gestural interactions during meditation are further explained in the following section.

Table 1. Speech Recognition for Interface Navigation.

<i>(From Page)</i>	<i>(To Page)</i>	<i>(Keywords)</i>
Main	Instructions	instructions
Main	Meditation // Focus	focus, focused
Main	Meditation // Happiness	joy, happy, cheerful
Main	Meditation // Zen	Zen, calm
Main	Meditation // Sleep	sleep, relax
Main	End Screen	exit
Meditation (any)	Main	back, go back, return
Meditation (any)	End Screen	nice, done, end, Namaste
End Screen	Main	back, go back, return
Instructions	Main	back, go back, return

Table 2. Speech Synthesis for the Meditation Assistant.

<i>(Upon Loading Page)</i>	<i>(Speech)</i>
Main	<i>Welcome (name). How can I help you today?</i>
End Screen	<i>Namaste</i>

(4) End Page

The user can use words which indicate the end of meditation or use the keyboard to access the end screen. The meditation assistant thanks the user by saying “Namaste”.

(5) Mudras Reminder Pop-Up

A pop-up window was created for learning purposes. It shows up upon starting the app and indicates the gesture to control it. It shows pictures of all mudras which are detectable by the system.

3.2 Gesture Recognition

An important functionality of the interface is related to gesture recognition which empower users: 1) to **learn about mudras used in the meditation process**, which enrich the meditation experience by adding another layer of feelings; 2) to **control the sound which reflects the mudras' meaning** in the meditation process. To enable the accuracy of detecting the static gesture, we choose to utilize MediaPipe Hand Detection Library to detect the customized gesture. This library enables us to define and categorize mudras without using Tensorflow or self-trained models.

With MediaPipe Hands, employing ML to infer 21 3D landmarks of a hand from just a single frame, we are able to categorize customized mudras gesture detection based on each finger's status(open or closed). Also, the right and left hand can be detected. Fig. 4 shows the landmarks detected in MediaPipe. By calculating the figure status of each finger, for example, e.g. for index figure, the y of landmark 8 is smaller than landmark 7 and 6 when this finger is closed, we can detect if the finger is open. Then we use a different combination of the finger status to categorize the customized gesture.

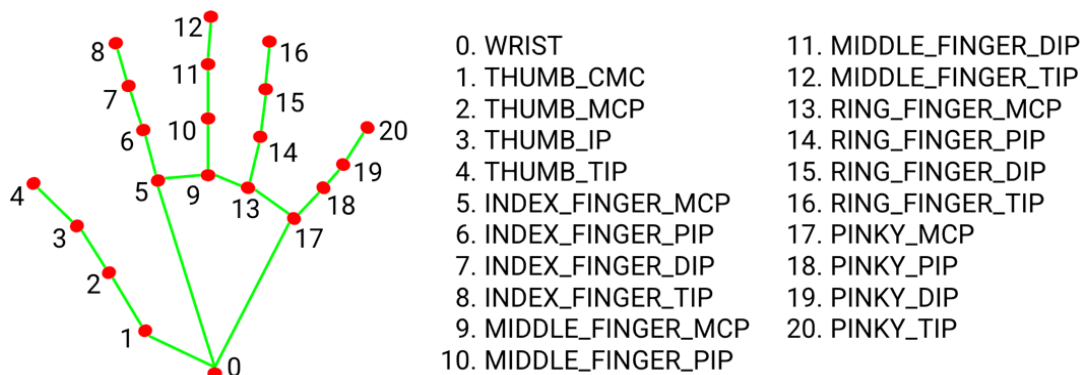


Figure 5. Hand Landmarks (source: Google, mediapipe)

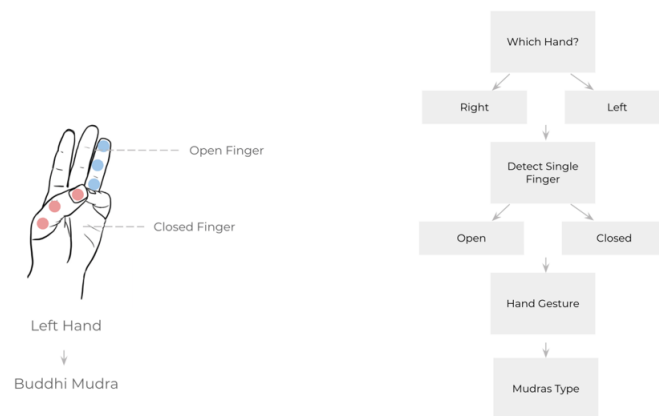

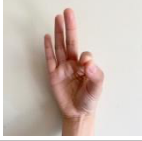
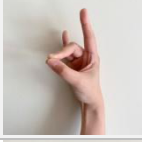

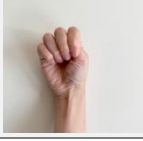


Figure 6. Mudras categorizing methods

In our project, we made 5 customized mudras based on these methods. The following chart and picture shows the figure status of each mudra:

Table 3. Gesture recognition based on finger status

MUDRAS	THUMB	INDEX FINGER	MID FINGER	RING FINGER	PINKY FINGER
	close	open	open	close	close
	close	close	open	open	open
	close	open	close	open	close
	close	open	close	close	close
	close	close	close	close	close

3.3 Successful Elements and Interesting Failure

In this project, several things can be considered as successful, both with respect to learning, and achieving our goal of getting the application to work as intended.

- 1) We can successfully detect 5 static gestures and apply them to different interface controls with different meanings (mudras, and open-close the pop-up window). At the same time, the detection of gestures is customizable.

To get there, we gained valuable experience and learned different things by experimenting with different tools and methodologies. We experimented with Leap sensor and computer vision, we tried multiple libraries and frameworks (OpenCV, handtracking.js, Leap SDK, ml.js) and we, finally, chose Mediapipe library to customize gesture recognition. Tracking 21 hand landmarks enabled us to evaluate the status of each individual finger to define and, then, identify the gestures. While at the beginning 60% of the gestures were correctly recognized with OpenCV-based recognition (due to the small amount of training data), we figured that with a pre-trained

model it is relatively easier to handle the hand detection and further extend the customized hand gesture library based on this feature.

- 2) We implemented speech recognition for Interface navigation using an intuitive interaction vocabulary. In all cases, users did not have any difficulty picking up and using the speech commands.
- 3) We proposed, designed, and implemented a novel meditation experience that is based on multimodal interaction which only requires basic equipment. We managed to develop an idea/concept from scratch to a working and visually pleasing prototype. Different components (gesture, speech, sound, graphics) are integrated to provide a simple and enjoyable user experience. We believe that our proposed idea, despite the changes and fixes that are yet to be made, is an interesting contribution to the existing digital tools to support meditation.
- 4) We managed to conduct user studies, remote and in person with four users other than ourselves, whose feedback allowed us to troubleshoot our intentions and implementations during project development.

As for the interesting failure, it is actually the function of our application and reminder page. The function of pop-up page is supposed to remind users the forgotten mudras during the mediation process, however, due to the fact that it is hard to control, users normally forget to user it. The fun thing is, from our user feedback, users mentioned that through the instruction and the mudras instruction, they started knowing about mudras and get interested in using the mudras. The following pictures shows the reminder window, supposed to be a reminder but finally became a learning tool.

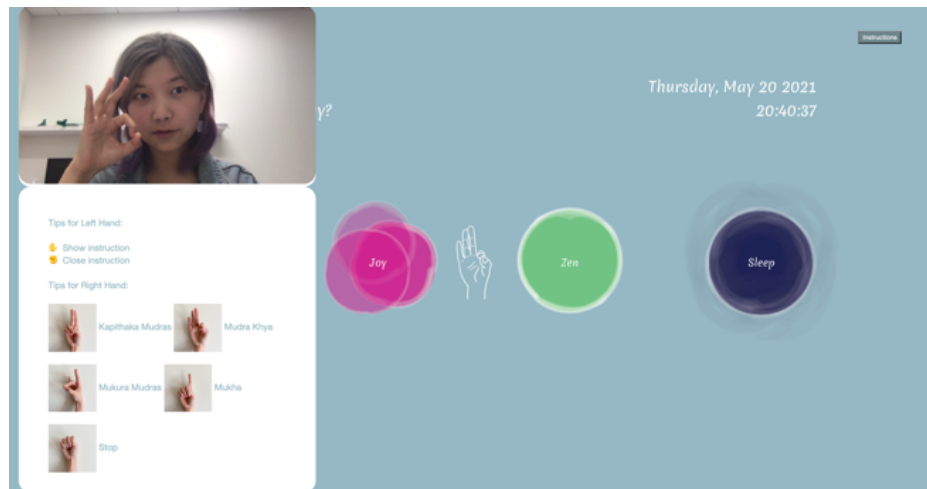


Figure 7. Mudras Pop-up Reminder

3.4 Previous Experiments

Before settling for the final version of the application, different options for multimodal interaction were considered and tested. Our initial intention, presented in the Design Studio, was to implement gestural controls for the application, such as volume control with up-down gestures, together with the mudras' recognition, using the leap motion controller. In this proposal, we had included the idea of incorporating eye-tracking to determine whether the

(Main screen)

Good morning my friend
Tell me, how can I help you today?

Tuesday, 11th May 2021
08:12:58

focus happiness calm relaxation

(Meditation modes)

User can say: "focus", "happy" or "happiness", "calm", "sleep" or "relax" to select a meditation mode

Focus track plays on the background

Calm track plays on the background

Happy track plays on the background

Sleepy track plays on the background

User can say: "Back" to go to the main menu // "Nice", "I am done", "Namaste" to go to the end screen

(Instructions)

Instructions

You are designed for different purposes with your voice.
Select between the different meditation modes by voice.

How to use:
"focus"
"happy" or "happiness"
"calm"
"sleep" or "relax"

It works in meditation mode.
When in meditation mode, say "Go Back" to return to the main menu.

Say "Nice", "I am done" or "Namaste" to go to the end screen and stop the application.

During meditation you can press (HOME) to return to the main page.
Press (X) to enter or exit meditation mode.

Enjoy!

(End Screen)

Until next time...

Namaste

We developed two versions of the interface, and tested each version. In the first version, the main components of the interface were developed: the main screens, as well as different navigation options with keyboard and touch. We implemented in meditation mode for volume control and gesture recognition using the finger tips for gesture recognition. The implementation of the interface corresponded to gestures of pre-trained models. We presented the interface to different users.

⁴ A detailed documentation of the work on recognition using the Leap sensor is provided in the Appendix.

For the Implementation Studio, we focused on gesture detection with a camera. A tracking skeleton data was preferred over fingertips to achieve a higher accuracy. For this iteration, the speech interactions were further developed to include a richer interaction vocabulary and controls over more navigation tasks. An extra mood was included to the Mood Palette, and a preliminary version of the Instructions page was added.

For the final, recognition is based on a hand model with landmarks indicating the open or closed status of each finger. This solution not only allows better classification of each gesture, but also makes the system a lot more versatile, as it can easily be extended to incorporate any number of gestures which can be defined based on the above method.

Table 4. Prototype Iterations Comparison.

<i>Iteration</i>	<i>Gestures</i>	<i>Interface Navigation</i>	<i>What is good</i>	<i>Required Improvements</i>
Design Studio	Leap motion control was proposed for gesture interactions	Speech recognition Mouse / Keyboard	Camera recognition proved more intuitive to use for meditation	- Recognition accuracy - Interface functionalities (navigation, Instructions) - Sound selection
Prototype Studio	1. Leap Motion Sensor (Leap SDK, ml.js) 2. Camera (Handtrack.js) - finger tips used for recognition	Speech recognition Speech Synthesis Mouse / Keyboard	Interface Navigation with speech intuitive/easy to grasp Gestures are recognized	- Recognition accuracy - Speech vocabulary should be enriched
Implementation Studio	Skeleton data used for recognition	Speech recognition Speech Synthesis Mouse / Keyboard Buttons and Sliders	Higher gesture recognition accuracy	- Gesture customization is required
Final	MediaPipe Hands	(same as before, refined)	Customizable gesture, high recognition accuracy	-Instruction page -Pop up learning window -left/right hand recognition

4 User Study

Designed for meditation beginners and meditators who want to learn about using mudras in the process of meditation, we have conducted user interviews before the project and user tests during the project development. In the following two sections, we document the results of our user studies.

4.1 Initial User Interviews

To prove the concept and get a deeper understanding of the meditation process, we conducted interviews with one amateur meditator, who is identified as our targeted user, and one product designer, who is treated as an expert. Within the conversation, we talked about the learning process of meditation and different types of meditation techniques. We listed the following insights:

1. The meaning of mudras has not been well accepted and known by meditation groups, the current meditation mainly focuses on close-eye meditating and sound-guide instructions.
2. The meditation interface is a good way to know about meditation feeling, but for close-eye meditators, they may not want to stare at their laptop for a long time. You could use the interface to learn.

3. During the meditation, some assistant tools could be used, such as small drums to indicate the changing of mind and status.
4. Meditators prefer hand-free meditating experience.

Following the insights from the interview, we adjusted the functions and purpose of the project during the process, including:

1. utilize the interface as a guide tool to help people learn about mudras and experience meditation.
2. Speech will be used to change mode, and mudras will control the chime and drum sound as an indicator of meditation status change.
3. Add more detailed instruction and treat users as a first-time meditator.

4.2 User Testing

We followed different testing methods throughout our project development. During the initial stages of the application, we gave specific instructions to the users on how to interact with the interface. After integrating the Instructions in the application, we were able to test how clearly information related to usability was communicated and, also, how each user spontaneously reacted to our design. In this section, we focus on the experiments conducted using the final interaction of the application.

Experiment 1

Part 1 (guided)

A guided experiment was conducted where a user who is not a meditation practitioner and had no prior interaction with the interface. The experiment comprised two parts, and for each one the user was given specific tasks to perform.

The purpose of the first part was to see how easy it is for a person who is, in general, familiar with technologies, to understand and use gestural interface controls. Additionally, we intended to test the ease of use and the necessity of the pop-up window as a reference for the mudras learner.

Tasks:

- a. Open and close the Mudras Reminder with gestures
- b. Learn the five mudras indicated in the instructions

Results:

- a. With no prior knowledge of gesture controls, it took the user less than 2 minutes to understand how to control the pop-up window.
- b. After several iterations of opening and closing the pop-up window to remember the gestures, the user was able to memorize and perform the five basic mudras in about 3 minutes.

The results of this experiment show that, for a new user, familiar with technologies, it is fairly easy to understand how to use gestures to control an interface. The process of learning the control gestures was nice and quick, and soon he was able to use them without realizing. With respect to learning the mudras, the pop-up proved really helpful. It was very easy to use, and provided direct reference. Since recognition worked simultaneously, the user was able to see and do the mudras, then close the window and test if he could repeat them by heart.

Part 2 (semi-guided)

The purpose of the second part was to give a general instruction which urged the user to interact with the interface and figure out which visual and sound elements were more appealing, in which order they would choose to navigate the interface. At the same time, we wanted to figure how the overall experience would change based on the prevalent way of interaction (speech vs. conventional keyboard/mouse controls).

Tasks:

Explore the moods -a- with speech only, and -b- in silence. Then, compare and report on the two experiences.

Results:

The user started by using speech to navigate the interface. Some problems with speech recognition came up with the selection of two of the 4 moods, so he had to use the mouse to overcome this problem. Speech recognition failures were related to a speech library issue which paused speech recognition after 60-90 seconds, and with non-native pronunciation of words. Interface navigation with mouse and keyboard did not indicate any errors. The user reported that mudras recognition, although fairly accurate when performing a gesture, requires further improvement as chimes come up when no mudras are shown. Finally, comparing the two experiences, the user said that failures related to sound were far more disturbing than the silent failures such as not being able to bring up the pop-up window or close it with the first try.

By observing the user, we pointed out that the graphics that were more intense in terms of color, contrast and motion, were more attractive to the user (such as Joy and Sleep). The user indicated his preference for the moods in the following order: Joy, Sleep, Focus, Zen. To evaluate this result, the inexperience of the user with meditation should be taken into account. Ideally, the two navigations versions could have been tested with more people, of different ages and backgrounds, the results could have been a lot more informative about the impact on different user groups.

Experiment 2 (unguided)

The purpose of this experiment was to test the usability of the interface and the functionality of the entire user flow. In this case, the user was asked to test the interface themselves and conduct mediation process without any verbal instruction from us.

Results:

At the beginning, it took user some time to figure out the left and right hand controls, finally the user learned the gesture to open and close the instruction as well as pose the mudras to trigger sound changes.

There are several learning outcomes of this test: 1) Enhance the readability of the instruction; the user mentioned that since the font is small in instruction and normally meditator will not start at screen to read the instruction, it is better to have voice over instruction rather than only readable version; 2) the instruction pop-up window is useful for learning and calibration; user said that the pop-up window is useful to learn about mudras and mimic the hand gesture. 3) sound volume adjustment need to be improved; the users would like to adjust the volume since sometimes the chime sound is too loud. From the. Unguided user test, we got the problem in the user flow side, and further improvement is needed to enhance the smoothness of overall experience.

5 Performance Evaluation and Insights

There are several inefficiencies of the system, on a technical and design level, which we started to realize and partially solve while developing the project.

- Imaginary vs. Real use

By comparing the leap sensor and camera versions, we realized that our expectation of the user response to a certain technology differed from reality. The fact that the sensor limited the scope of gesture recognition made the user extremely aware of the device, which counteracted our intention to make the user develop focus and insight while using the application. This showed us that achieving a technical task does not guarantee a successful user experience. Also, due to the placement limitation of leap sensor, we finally chose to use camera with a more flexible placement option.

- Usability Tests

A crucial takeaway of this project is the importance of user testing. While we have been able to test the functionality of our designs on a technical level, every user test we have conducted has revealed a series of inefficiencies which needed to be tackled in the following iteration. Our particular inability to find and test the interface with frequent meditation practitioners has given us less insight on aspects related to the actual meditation experience, on which we could have focused more. For this reason, the Mood Palette is less developed towards this direction. At the same time, testing with non-experienced or amateur meditators gave us a lot more intuition and feedback on the learning aspects of the application, which we decided to integrate along the way.

- Customization

An important improvement for our system is to make it fully customizable in terms of gesture and mood selection from the user. For now, the system is limited to proposing specific gestures to the user, and providing them only four options in the Mood Palette. As meditation is a unique experience for each person, we believe that, in order for this tool to be made really useful, it is important that it is highly customizable on this level.

- Learning

Since the interface is a learning tool, another level towards which the design could be expanded is related to adding more context on the meaning and the use of the mudras. While one aspect is learning the gesture itself, learning the way that the mudra is used is a whole different level of learning about meditation.

6 Tools and Packages

Our project includes three parts, hand gesture control, speech control and interface design. The following chart shows the tools and libraries we used in this project:

Package/Tool/Library	Where is it available? (eg url)	What does it do?	How well did it work?	Did it work out of the box? If not, what did you have to do to use it?
MediaPipe	https://mediapipe.dev/	In this project, we used it to detect customized static hand gesture	it is pretty accurate, and you can customize the gesture with simple calculation.	You can download the link and learn it from the docs. It is pretty easy to handle
Handtrack.js	https://victordibia.com/handtrack.js/#/	In the first version of the project, we use it to detect hand gestures.	Since it has several pre-defined gestures, and it cannot customize your own gesture, we choose another library at the end of the project. However, if you just want to detect hands and simple gestures, it is the easiest choice (for webcam).	Follow the docs, it is pretty easy to handle
Leap Motion SDK	https://developer.leapmotion.com/sdk-leap-motion-controller	https://blog.leapmotion.com/getting-started-leap-motion-sdk/	It provides a big set of methods that give access to points on the hand model (x,y,z at each frame), has inbuilt methods for moving gesture recognition such as grab, pinch, swipe up/down, circle ect. which are very useful and tunable for different applications.	The swipe up/down gesture was used to adjust the sound volume. The speed of the gesture needs to be adjusted for better performance. Data from each frame (fingertip positions) for each gesture were used for gesture classification with the kNN. The data require normalization for better results.
ml.js, kNN	https://github.com/abhisheksoni27/machine-learning-with-js and, specifically https://github.com/abhisheksoni27/machine-learning-with-js/tree/master/knn	It can be used to accurately recognize hand gestures/patterns based on similarity. Similarity is calculated based on Euclidean distance. https://hackernoon.com/machine-learning-with-javascript-part-2-da994c17d483	In general, kNN performs very well in a wide variety of applications.	Add the library in your code. Train/test datasets need to be created and loaded for the kNN to work.
p5.js.	https://p5js.org/download/	This is an excellent library for interface prototyping, since it has a great set of functions for visuals/graphics. See also: https://p5js.org/reference/	It is perfect for prototyping creative and highly customized user interfaces. It is not ideal for computationally intensive applications.	Yes. It has a draw() loop which updates a canvas on a given framerate. The canvas can be adjusted to fit the browser window, or can be integrated with any other DOM elements.
p5.sound.js	https://github.com/processing/p5.js-sound	Sound, very easy to use add on for p5.js https://p5js.org/reference/#/libraries/p5.sound	It works very well with p5.js. Just load the sounds in a preload function. Has many methods to control sound and it is very simple and well documented.	Just download, add to libs folder, call in html
p5.speech.js	https://idmnyu.github.io/p5.js-speech/	Speech Synthesis/Recognition, very easy to use add on for p5.js.	It works very well, apart from a fix that is required, as recognition may stop after 60-90secs.	Just download, add to libs folder, call in html

8 Collaboration

Ziyuan Zhu | initial proposal and research, hand gesture recognition, learning page design, user test(50%), video making (50%)

Aikaterini Lamprou | idea development, interface design, speech recognition/synthesis, user test(50%), video making (50%), leap motion detection (initial version)

Other parts are all collaborative work.

Appendix | Leap Motions Sensor for Moving and Static Gesture Recognition

GitHub: https://github.com/babel-kat/InteractiveMultimodalMeditationInterface/tree/leap_version

Leap sensor data were used in two cases: first, to implement the swipe up/down gesture for sound volume control, and, second, to recognize five different mudras.

The Leap SDK was used to implement a moving gesture control of the interface: turning sound volume up and down while in meditation mode. The inbuilt swipe up and down methods were used to recognize the two gestures.

To recognize the different Mudras, we decided to use a kNN classifier for the following reasons. First, the classifier takes into account similarity between gesture patterns. Second, it can easily incorporate multiple features, such as the fingertip positions. Third, it is easy to tune since it has only one hyper-parameter which is the number of neighbors (k).

The procedure we followed is the following. A JavaScript program was created, which allowed us to collect gesture data (frames) from the Leap Motion Controller. Data points from a single hand were collected for each of the five static gestures (approximately 70dp/gesture for training and 15dp/gesture for testing). The main reason for collecting data from only one hand was because this way only it was possible for a single person to capture the frame data (the other hand needed to press a SPACE on the keyboard!). Apart from the five mudras classes, an extra class with random gestures was also created with more data points, in order to classify random hand movements which did not belong to the set of five gestures that needed to be recognized. Data were collected from 2 users, one for training and one for testing. A JavaScript kNN classifier from ml.js was used. Multiple tests were run and it was determined that training was optimized for $k = 3$. The classifier was trained using the train data with $k = 3$. Although the data were not normalized, and were taken only from a single person due to time limitations, the overall test accuracy was close to ~60%.

Although the initial results of implementing the above functions with Leap sensor were quite satisfactory, the idea of using it for the app was discarded in favor of trying out remote controls and recognition using the webcam. Testing the Leap version with users indicated several things about the use of the sensor:

- (1) the user was easily confused about where they should perform the gestures so that the sensor could capture them. This, created an significant mental overload which counteracted the idea of meditation,
- (2) the user unwillingly mistook the sensor for a microphone. They took their hands off the sensor to talk towards the direction of it in order to try speech interaction,
- (3) limitations applied to the possible physical setup and the preferred distance of the user from the computer screen due to the need to plug the sensor on the computer,
- (4) limitations applied due to the inaccessibility of the Leap Motion sensor in most cases of potential users.