Hand Gesture Shortcuts

Jose Cadenas, Christopher Gomez-Selio, Zachary Lebrilla, Jonathan Srinivasan

Project Description and Motivation

The production of our Hand Gesture Shortcuts program was initially inspired to help decrease computer navigation times for able-bodied and disabled users alike. At first we focussed on the benefit towards computer users affected by either Rheumatoid Arthritis or Osteoarthritis: those who may find it more difficult than the average able-bodied user to repeat the many mundane hand strokes required to navigate to, and while on, a website or application. The group that we believed would receive the second most benefit from this application, and others alike, are the average user, as a common issue with humans is their belief in being able to multitask. By using Hand Gesture Shortcuts the user would ideally be cutting out all of the middle steps of navigation and turning the entire experience into a two step, “wave then browse”, function.

To begin our application the user must open Chrome, in order to receive the permissions to use the computer’s webcam. Within the program the user is able to specify multiple paths for the camera to track and match. Our method completely gets rid of the need for a keyboard during the navigation process to a website, starting from when the camera permissions are given. Our application was also created to be an alternative to voice commands. Rather than needing speech and accent recognition software, that requires a training period with new users, Hand Gestures Shortcuts is ready to track and open websites the moment you download. The only constraints on the quality of the user’s experience are on the quality of the webcam being used and the lighting of the area during usage.

Prototype Design

After deciding to use the library, handtracking.js, to build our program, we first built a simple Air Guitar application. The application essentially displayed letters on your computer screen, and when the camera catches your hand swipe across the letter, a corresponding sound would play. The purpose of creating this Air Guitar application was to gain familiarity with handtrack.js and getting used to tracking the coordinates and motion of our hands.

We then started implementing our hand gesture application. Our first version had only one way of motion implemented. A horizontal swipe of a hand from left to right would open up a website. After our first version, we started implementing all sorts of directions: vertical, horizontal, and diagonal. In total, we could swipe in eight different directions. For our final version of the application, we added a hand icon and a “GO” icon for a better user experience. The hand icon will appear at the start of the application to show the user where to place their hand. Once their hand is over the hand icon, a “GO” icon will pop, and then the user can swipe in any direction to navigate up their intended website.

When creating our final design, we did run into an unique situation. One might compare it to the Greek story of The Midas Touch. The story is about how King Midas wished that everything he touched would turn to gold, and his wish was granted. He enjoyed his ability to turn any object into gold, but he soon realized the downfall of this ability. Whenever he touched food, it would turn to gold, so consequently, he almost starved himself to death. In our case, our computer would immediately start detecting the user’s hand while the user is trying to position his/her hand. Due to this, while positioning his/her hand, the seer would open up web pages he/she did not intend on opening. To solve this issue, we added a 350ms, giving the user enough time to position his/her hand without accidentally opening a website.

Implementation Details

The most important part with implementing hand gesture shortcuts was which library we used to track hand movement. On our first iteration of the design we were looking for one that could track fingers, worked quickly, and could be used offline. We soon discovered that in order to get finger tracking to track hand gestures, reasonably well, to map shortcuts, the camera angle had to be very specific. Many of the first libraries we went through could track finger movement but had a hard time telling what side of the hand it was tracking and whether the fingers were moving towards or away from the camera whilst in a claw shape. Ultimately we decided to scrap finger movement and pick a library without it. The second issue was with offline usability. Most of the libraries we found used a media pipeline to send the video input to another server and have them processed there before returning the hand tracking. Offline usability was a big component when the desire was to make desktop shortcuts but ultimately useless when the switch to browser based shortcuts was made.

Ultimately the library we decided on using was handtrack.js. As the file extension implies it is a JavaScript library and its intended usage was for web applications. The library takes an input of an image or video feed and returns a bounding box and confidence interval around the hand. It's worth noting that this library does not have finger tracking or offline usability but because of these factors, it is a lot faster than other similar libraries. While it can still be taxing on the CPU if multiple windows are running we found that it was the best for our given use case.

In order to implement shortcuts using any webcam available, we take the box positional value and we track it over an interval of 350 milliseconds. This process can be scalable to different hardware requirements. We track the x and y value of the hand bounding box returned by the library. Using these positional data points we build a velocity vector between them keeping both the direction and speed of the hand movement. We then use this velocity value to call on different shortcuts to open websites. For visual sake, there is a hand and green “GO” sign that operates on a different interval tracker in the center of the screen. Every second it checks purely positional data as to whether the hand is in the middle of the screen and gives the "GO" to begin gestures for shortcuts.

Removed Features

Our project's original motivation was to help those with disabilities, such as arthritis, open desktop applications with a simple or complex gesture, rather than navigating through the desktop's menus. Due to the group's collaboration constraints set on us by COVID-19 we found it difficult to test and develop an application that would work across multiple operating systems (OS) thus we decided to cut this feature and prove the concept's success with a web application that opened URLs. Another factor in that decision was the use case, for instance opening our app to open another is contradictory to the entire motivation. Moreover, this gesture control would ideally be best if it was built into the OS to increase security and computational power consumption. The following feature to be removed was the ability to operate offline. Offline functionality made sense for launching desktop applications but does not for opening URLs on the internet, therefore it's reasonable to lose this functionality. The final major feature that we removed was finger tracking, which proved to be difficult to do universally. To track finger motion, we needed very specific camera orientations which did not make sense since we wanted to reach a wider number of users. Thus, our use of full hand gestures makes that the ideal use case for everyday users.

How to use our Hand Gesture program

To use our program, open Chrome*. Once the page has loaded up handtrack.js will automatically start detecting one hand and tracking its position on screen. We have encoded a starting position signified by the hand overlay in the middle of the screen, use the webcam footage to position one of your hands over that hand overlay. Our program will verify that your hand is in the center of the screen every 350 milliseconds (ms). If the condition has been met then a green go icon will appear signifying your ability to proceed with a gesture, recognition will be enabled for one second. After that second the green go icon will be replaced with the hand overlay signifying the starting position is needed again to detect a gesture.

Simplified Instructions:

1) Open Chrome*

2) Use the live webcam footage to place hand over the hand overlay

3) The green go icon signifies that you may proceed with a gesture

*Initial setup; open Chrome and set our website as your default startup page and allow webcam usage for this site. This is the ideal use case to save the user time.

Study Design

According to our research, there are three fonts that are preferred by people living with Dyslexia: Helvetica, Verdana, and Arial. We ended up using Arial for our study, but for no particular reason over the other fonts. The app features a green GO to let the user know when they are able to start their hand gesture. The GO is created using a PNG file and due to time constraints attributed to the current COVID-19 pandemic we were unable to digitally alter the photo to a more suitable color for colorblind users.

We recorded the navigation times for these following 4 ways of navigating to websites:

	Description
Case 1	Subject is only allowed to use one hand to use their keyboard to navigate to a website
Case 2	Subject is able to use both hands to use their keyboard to navigate to a website
Case 3	Subject is able to use both hands and a bookmark along with their keyboard to navigate to a website
Case 4	Subject uses Hand Gesture Shortcuts to navigate to a website.

For the trials that require a keyboard the subject is to have their mouse in the center of the screen and the timer starts after a countdown at which the subject is to navigate to their preferred browser and open the selected webpage. Time stops when the webpage begins to load. For the Hand Gesture Shortcuts trials, the subject is shown a green “GO” sign to know when to move for the gesture. Time starts at the GO and ends when the webpage opens. Each test is done three times and the average time for each test is what is ultimately being compared. Due to instrumental constraints we were only able to time navigation times to the 100ths of a second.

Results

Before conducting our research, we believed case 1 would take the most time, case 2 would take the second most time, case 3 would take the third most time, and case 4 would have the fastest time. Due to COVID-19, the participants had to be individuals that we reside with, meaning housemates and/or siblings. Therefore, the participants were around the ages of 18 - 23.

People around this age tend to be very familiar with technology so our participants were already fluent in the actions of our research. After every trial for each case, they seemed to get faster and faster because of the repetitiveness.

After conducting our research, our results differed slightly from our expected results. Here are the average times to took for each cases:

Case #	Average Times
Case 1	6.46 seconds
Case 2	4.52 seconds
Case 3	2.96 seconds
Case 4	3.12 seconds

As you can see, our predictions for case 1 and case 2 taking the longest times compared the other two cases were true. However, the average time to use bookmarks was faster than using the Hand Gesture Shortcuts. The time it took for participants to set up their hand and swipe to open a website outweighed the time it took for them to click on a bookmark. We also recognized that once our participants knew where to look for the bookmark, the times for trials 2 and 3 were significantly faster. Even though case 3 was faster than case 4, case 3 was only faster by 5.13%.

Conclusion (Limitations, Sources of Error, Room for Improvement)

Even though our research was conducted in a structured manner, there were still some limitations and sources of error in our research. Our first limitation was our participants. Our intended audience for this application is not only able-bodied individuals, but also disabled individuals such as those with Rheumatoid Arthritis or Osteoarthritis. We would have liked feedback and results from these audiences, but due to COVID-19, we were limited to the participants that we live with. Another limitation caused by this pandemic was that we were not able to test our participants on one computer. We were not able to achieve the independent variable of a singular machine, because everyone's computer works in different ways so the results will obviously be a little inconsistent.

There are also some sources of error in our experiment. The first obvious one is the human error of timing the navigation time. Since we are timing our participants using our own reaction time of starting and stopping the stopwatch, there is going to be some unavoidable human error. Another source of error is the placement of the cursor before every trial. The position of the cursor matters because every second counts in our experiment, and if the cursor is even slightly closer to its destination, it can affect the results. However there is no way to start the cursor at the same position before every trial, so we have to place it in the estimated vicinity everytime.

After finishing our research project, we looked back and reflected on what this project accomplished, and also what we could have done better. There is definitely room for improvement and if we had more time or had the chance to do this research over again, I believe we would have improved our application. An area of improvement is the reliability of our application because it sometimes takes a repeated swipe to open the desired website. The FPS and user interaction could also be improved because it can be very laggy at times.

Overall, Hand Gestures Shortcuts and its research did achieve a lot. It provided a solution for people with disabilities to navigate to websites. Those with Rheumatoid Arthritis or Osteoarthritis have a method to navigate to a website pain free by just using their hands. Through research, we found that it is faster than manually inputting the website URL and that is already a huge accomplishment. Although it is slightly slower than using bookmarks, it still provides a fast alternative to opening websites. Ultimately, Hand Gestures Shortcuts took a huge step towards decreasing navigation times while providing itself with a lot of potential to improve.