ENS 491-492 – Graduation Project

Final Report

Project Title:

Zoom for Big Data Analytics: User Interaction Mechanism for Multimodal Interfaces in Collaborative Analytical Sessions

Group Members:

İrem Gürak - 25442

Sena Yapsu - 24874

Zeynep Tandoğan - 25200

Supervisor(s):

Selim Balcısoy

Ekberjan Derman

Date: 29.05.2022

Table of Contents

1. EXECUTIVE SUMMARY

Online meetings have gained a lot of importance recently. There are many online platforms that supply audio and video sharing, however, there is still a huge gap between a physical meeting and online meeting. Understanding each other while working on a screen is especially a problem since it does not provide any gestural communication. In this project, it is aimed to solve this issue by developing an interactive yet still online meeting environment which consists of a table, a Kinect camera -which faces the desk- and a projection machine. Some environmental and social obstacles have been faced during implementation and due to this reason, the methods and the plan of the project were always subject to change. In this paper, different methodologies that are used and different plans that have been followed are discussed in detail. There are also three different approaches implemented to solve the problem that was addressed before, but since they include some environmental constraints that could not be overcome, a fourth approach is implemented. This last approach seems to be the most promising one and this implementation reached success in basic hand gestures. So far, the aim is achieved in a different setup that uses a Kinect camera which faces the user, but since another aim of the project was to use the table that has the visual data projected on it and make the interactions more straight forward, another implementation with a table facing Kinect was needed. Additionally, this system does not need to be only addressed as a meeting environment, it can also work in art shows, museums, so that it needs to be developed in a way that it can be used in different places.

2. PROBLEM STATEMENT

For the last few years, online meetings have become a very important part of business life. There are many successful platforms that are being used for online meetings. However, online meetings do not make participants feel as if they are together and understand each other. This project aimed to enable more realistic and efficient communication in a situation where both parties are physically in meeting rooms but are in online communication from afar with each other. While there is a need to solve a problem or a different need of visualizing data, people need to show things to each other using hands and body gestures. As a solution to these problems and improvement for online meeting platforms; gesture recognition was studied.

In the literature, there are many gesture recognition implementations for Kinect cameras using many available languages. However, most of them require seeing all the skeleton of the human body and can only detect gestures when standing in front of the camera, which was not the case with this project. Apart from developments in applications such as zoom, an application that provides an active online meeting environment using a similar setup as ours has not been found yet. The aim of this study was to increase the efficiency in online meetings by using and developing image recognition algorithms with the Kinect camera which only captures hand movements from above.

The only similar study is "Facilitating Decision Making with Multimodal Interfaces in Collaborative Analytical Sessions". In this study tracking of hand motion was applied, and the gesture is recognized through the difference between two consecutive frames. This study was enlightening in terms of gesture implementation since it was a prior study done in the same system with this project. However, since most methods that have been used are outdated and not working as of this moment, another implementation became necessary.

The system that is being implemented in the scope of this project will be used for "Atlas of Opportunities". Atlas of Opportunities serves necessary information for the investment opportunities such as mobility patterns, publicly available socioeconomic sources, GPS location traces, ATM and bank transactions and call data records for the decision-makers, policymakers and investors. The main goal of Atlas is to give the right data analysis to bring the right resource to the right location at the right time and make these decisions easier and more consistent for the investors, policymakers and decision-makers. It is implemented for South Australia and is being developed for Brazil. South Australia implementation has 2 options for the map: Small Business Support and Business POIs(Point of Interests). Small business support enables users to freely compare a wide range of social, economic and demographic data about South Australia to produce insights for people, business owners, and authorities to help them make informed decisions. Business point of interest part has some marked business locations especially restaurants and cafes, which are the places where social bridges between communities are created. These locations are chosen by the algorithm implemented in accordance with the individuals' aggregate data provided from a variety of legal organizations. These data can help to

design cities and make future plans for the communities to find strong social bridges between communities and found businesses in the profitable locations (“Atlas of Opportunities”, n.d.).

1.1. Objectives/Tasks

· Progress with the most suitable set of methods for the project by trying different approaches

· Implementing gesture recognition by considering the gestures that are suitable for the system and can be put into practice

1.2. Realistic Constraints

- **Social:** This project may have restrictions in terms of privacy, required permissions should be taken.

  - To perform gesture and emotion recognition, camera permission should be taken from all users.

- **Environmental:** There may be environmental factors preventing the full usability of the system.

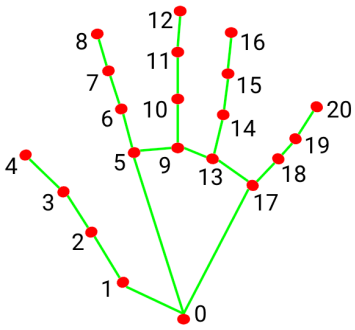
  - Full recognition may not be achieved due to the effects of light, position etc. The impact of these factors is being tried to be minimized during implementation.
  - Cross-platform usage can cause problems.

3. METHODOLOGY

In this section, all the implementation versions and the reasons why they are good or bad will be represented. At the beginning of the project, the main missions are to get coordinate information from the hand and to obtain frames from the Kinect camera. The main reason why Kinect is desired to be used in this project is to obtain depth information while performing recognition activities. To use Kinect with Python, there are several libraries such as libfreenect and PyKinect2. Firstly, libfreenect is tried; however, after long time trials, it is understood that it

is an outdated library. We published some posts in the websites about the problems that we experienced and it is seen that there are other people that can not overcome that issue. After that, we moved to the PyKinect2 library. PyKinect2 library enables writing Kinect applications and games. With pykinect2, all possible kinds of frame inputs such as depth and infrared can be obtained.

In the first implementation, Mediapipe library is used to detect the hand and obtain the coordinates of the fingertips. Mediapipe hands is a hand and finger tracking solution, it finds hands with 21 3D landmarks of a hand from just a single frame ("Hands", n.d.).



0. WRIST
1. THUMB_CMC
2. THUMB_MCP
3. THUMB_IP
4. THUMB_TIP
5. INDEX_FINGER_MCP
6. INDEX_FINGER_PIP
7. INDEX_FINGER_DIP
8. INDEX_FINGER_TIP
9. MIDDLE_FINGER_MCP
10. MIDDLE_FINGER_PIP
11. MIDDLE_FINGER_DIP
12. MIDDLE_FINGER_TIP
13. RING_FINGER_MCP
14. RING_FINGER_PIP
15. RING_FINGER_DIP
16. RING_FINGER_TIP
17. PINKY_MCP
18. PINKY_PIP
19. PINKY_DIP
20. PINKY_TIP

Figure 1 : Mediapipe Hands Solution

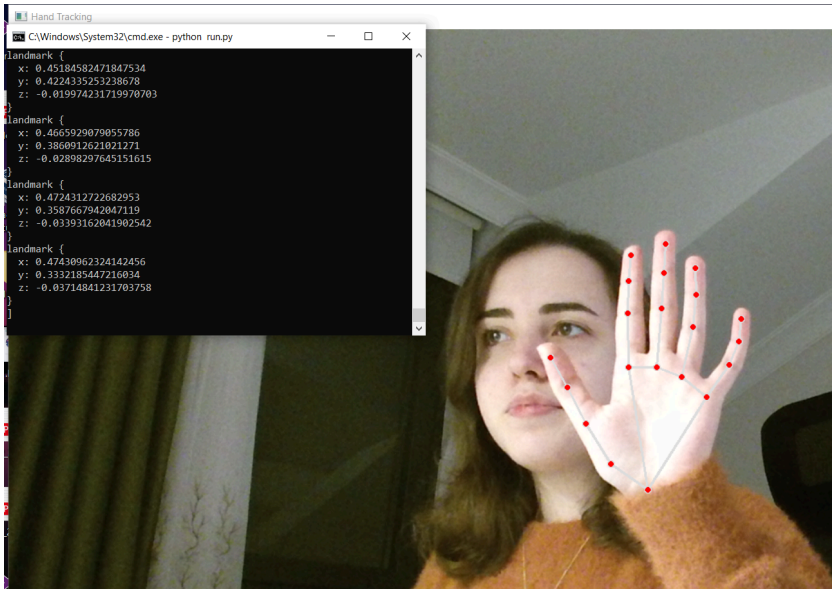


Figure 2 : Obtaining coordinates through mediapipe

The maximum number of hands that will be detected is given in the code for Mediapipe hands. OpenCV, which is a real-time optimized computer vision library, is used to process the frames that are required from Kinect and to draw the circles and boxes to the image (“OpenCV”, n.d.).

All the gestures are implemented by using mediapipe hands. At this stage, we have not moved to the setup in the lab. All of these are implemented for the case where the camera stays front, however, it is tested that it can detect the hand from upwards.

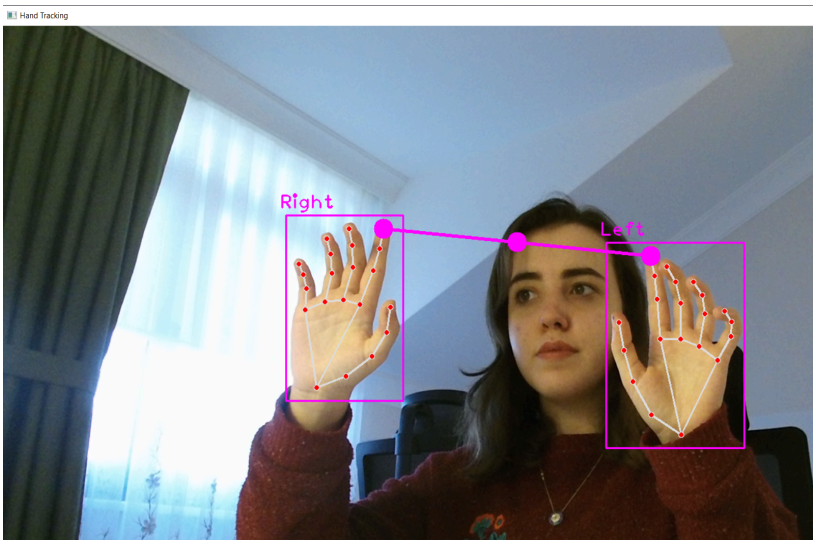


Figure 3 : the example of how mediapipe is used – Zoom In/Out

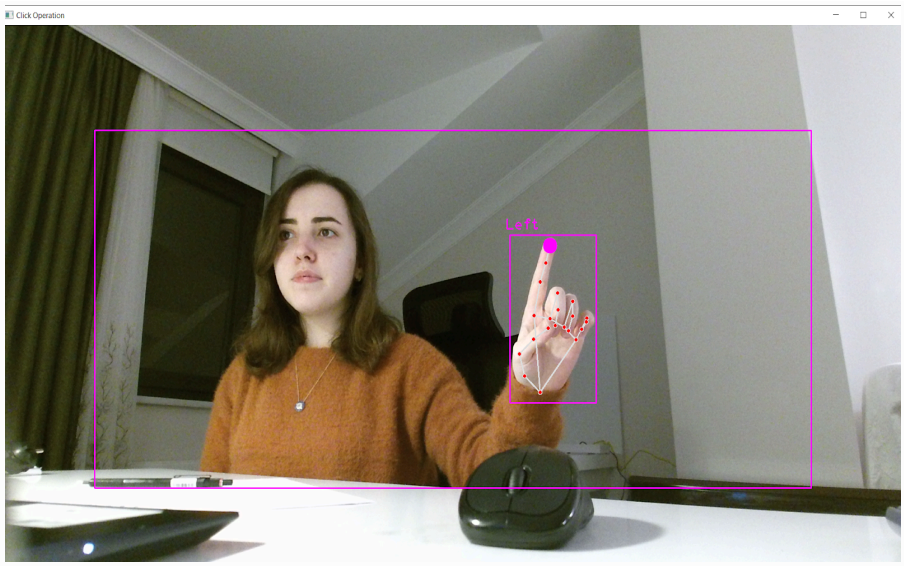


Figure 4 : The example of how mediapipe is used - mouse movement

Below figures, the demo pictures are presented from the mediapipe implementation.

Figure 5 : Clicking – first implementation

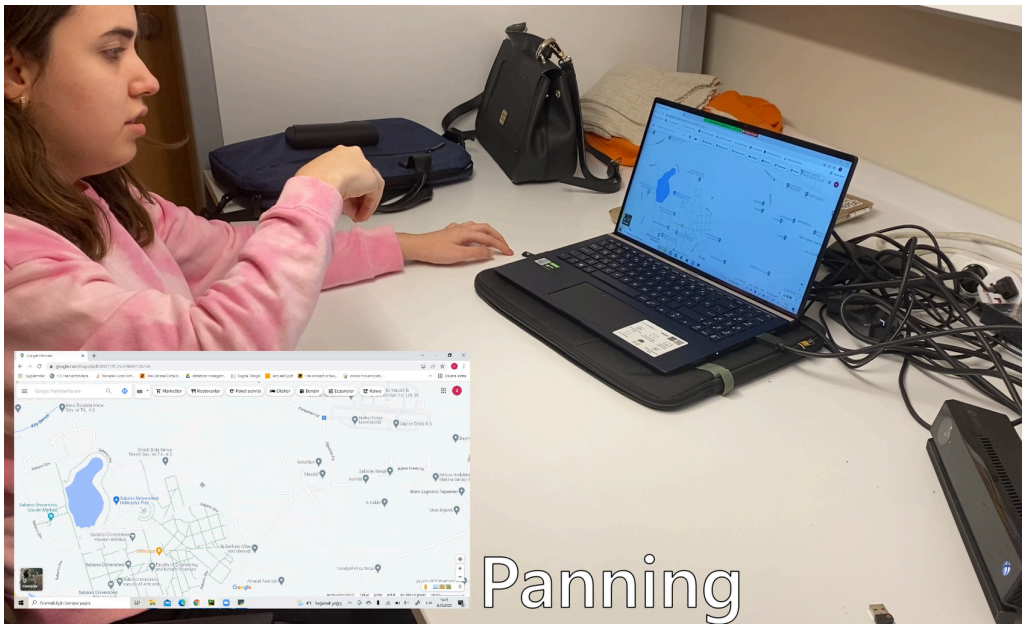


Figure 6 : Panning - first implementation

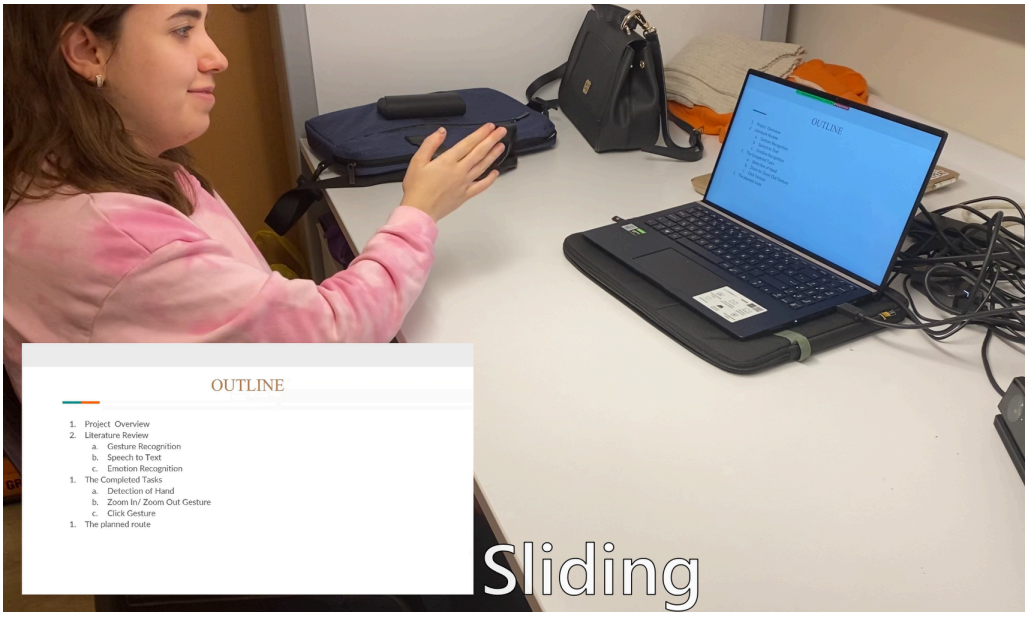


Figure 7 : Sliding – first implementation

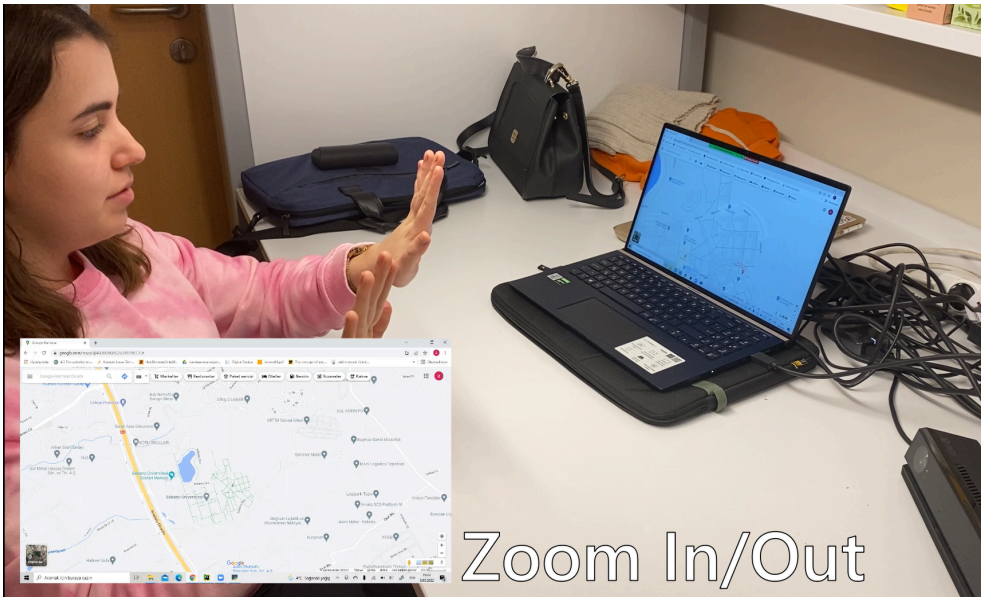


Figure 8 : Zoom In/Out - first implementation

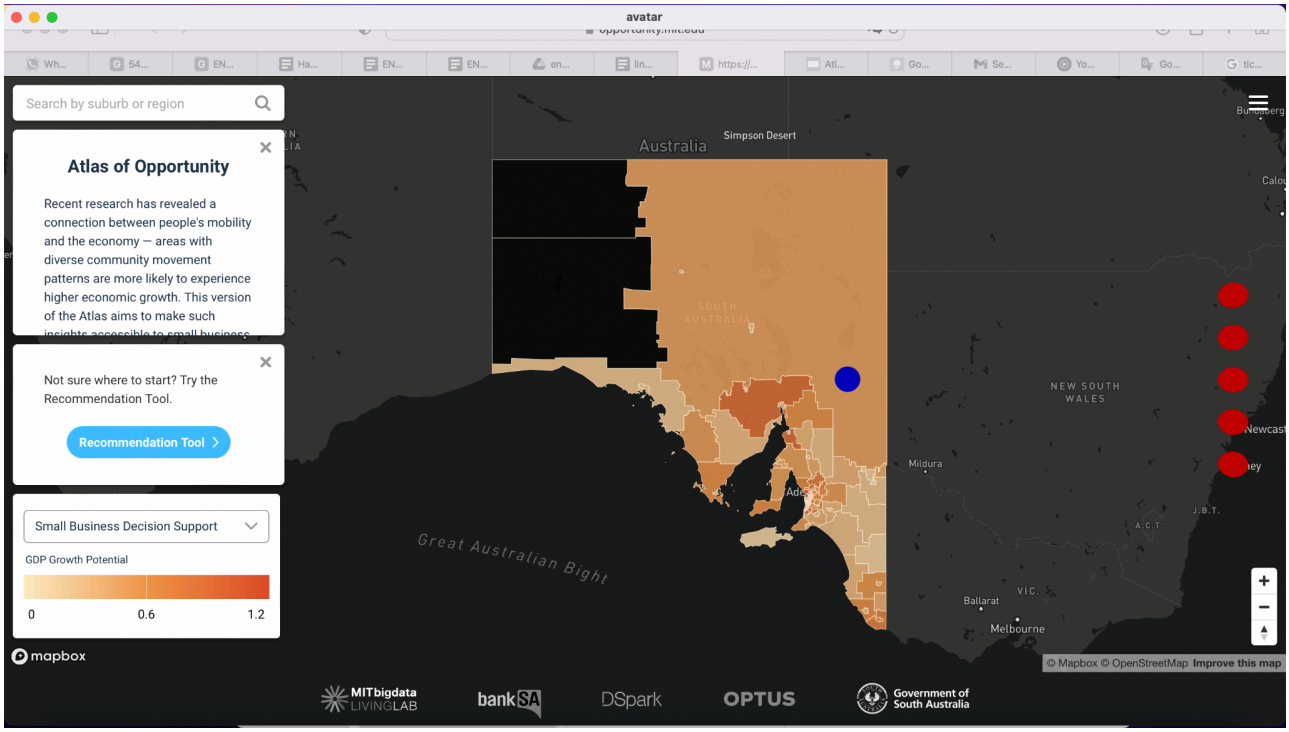
In this implementation; Python is used as a programming language, OpenCV, pyKinect2, mediapipe and pyautogui libraries are used. Server-client implementation is performed with gRPC. gRPC is a remote procedure call framework that can be implemented in any language ("gRPC", n.d.). It works consistently between two devices. However, there are factors that come up when we switch to the lab setup that are not desired. The first of these was that it could

sometimes find the hand and sometimes not find it, due to the camera distance being almost 3 meters. The other factor was that with the addition of the projection light, the hand could not be found at all.

In this case, it was tried to get the result by giving the infrared and binary image to the mediapipe first, but the mediapipe was quite unsuccessful in terms of working under those conditions. This idea is taken from the previous study like this one, they worked on depth and infrared frames from the Kinect (Kaya et al., 2020). Thinking that this is probably due to the distance, we tried to detect the person from the camera looking from above and cut the part where we found him from the image and give it to the mediapipe. For this, yolov5 is trained with the dataset that labeled people from the above camera. In addition to these, we were implementing the avatar part at the same time. An avatar for each user was created using the OpenGL 3D environment. 3D sphere objects are created for this purpose. Avatar implementation is connected with Atlas map in two different ways:

- Created a framebuffer in the OpenGL 3D environment and put the screenshot of the Atlas page as an image. Atlas page screenshot was taken with opencv in ubuntu. However, windows window manager API is the screen capturing method in Windows. This method has not been implemented yet.
- Made OpenGL 3D environment's background transparent and both Atlas map and avatars are visible in this method. Figure below shows the screenshot of the implementation.

After these improvements in the avatar part, this section is given to someone that is not in the team; in order to make us more focused on gesture implementation.



Detection is performed relatively successfully however, cutting and giving the crop to the mediapipe does not work. After this point, it is understood that this implementation is not applicable for our case and another implementation method is tried. Besides these factors, there is a delay in the movement of the mouse cursor in that implementation, for this reason at that point, it is decided to change the language to C++. Since we have a demo at that week, we tried to implement the best in that short time. In c++, there is a kinect library that finds joints when the camera is at front. In addition to that, kinect studio gesture builder is used to recognize panning and zoom in/out gestures. This implementation works successfully when the camera is front. If the scenario of the project was to project the screen to the wall, it can be applicable.

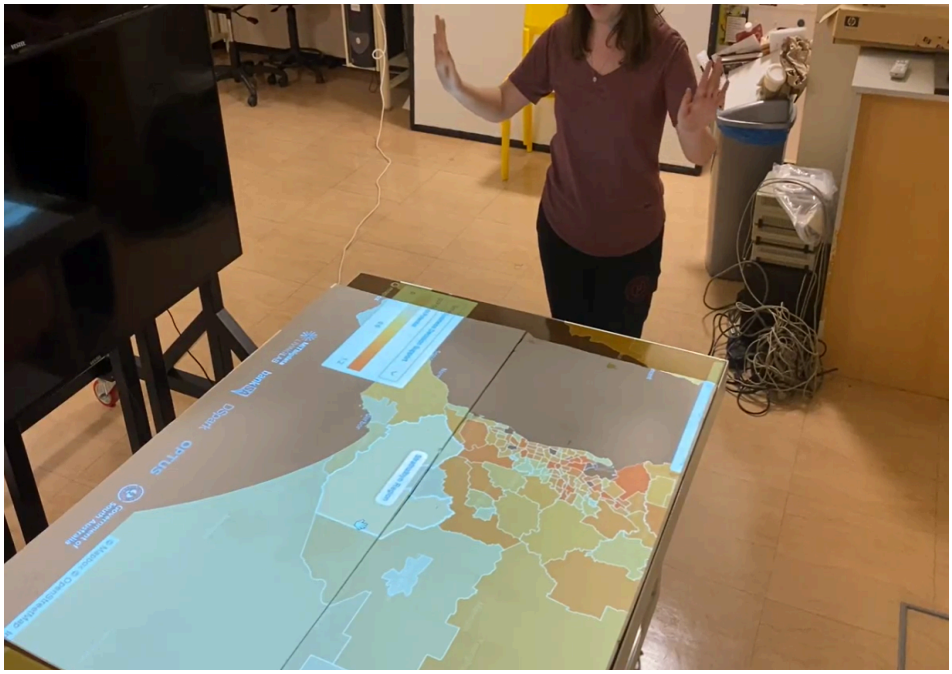


Figure 9 : Zoom In - second implementation

Figure 10 : Zoom Out - second implementation

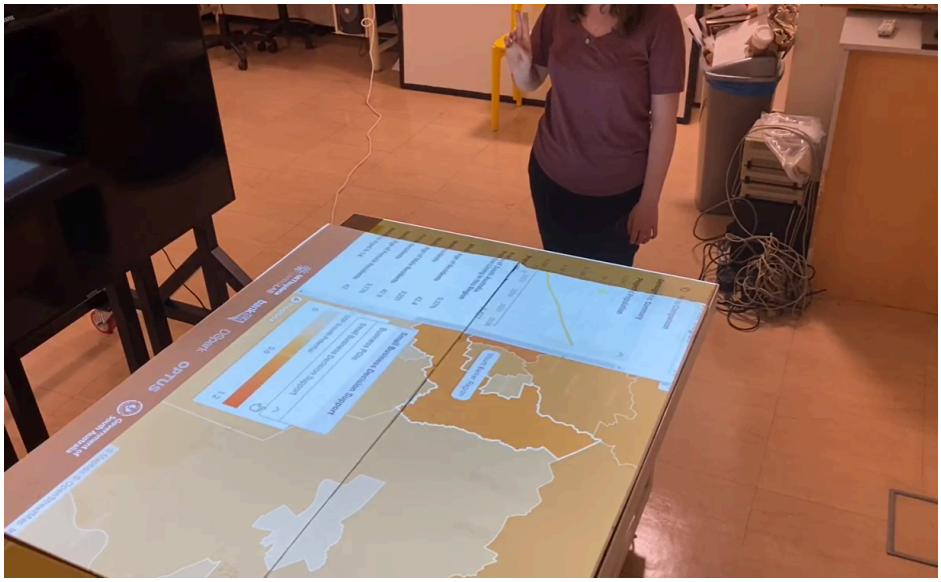


Figure 11 : Click - second implementation

However, since it is desired to interact directly with the table, this implementation is not appropriate for the case; but while doing this, in a short time, it greatly increased our knowledge in terms of implementation. After these, it is tried to implement mouse movement and click by using depth information. By analyzing depth changes in the obtained frame, the part that hand is located tried to be obtained; but at this point, it is seen that depth measurement of the Kinect is not sufficient. The arm and the hand are seen in the same depth; but they are not. For the demo, the first point that is deepest is directing the mouse cursor; if the hand is located in the same point in 30 frames it performs a click. It does not perform well actually, but in the demo it seems sufficient.

After the demo, we changed our implementation techniques for the last time. It is seen that the coordinate information of fingertips cannot be obtained by using a library. Because while some of them are not capable of detecting under a projeksion light, others cannot get coordinates when the body is not seen, which is our case.

For this reason, it is implemented such that the coordinates are obtained by following the contours. In order not to be affected by the projection light, an infrared image is used and then it is binarized such that the bigger portion in the image is made white and the rest remains black.

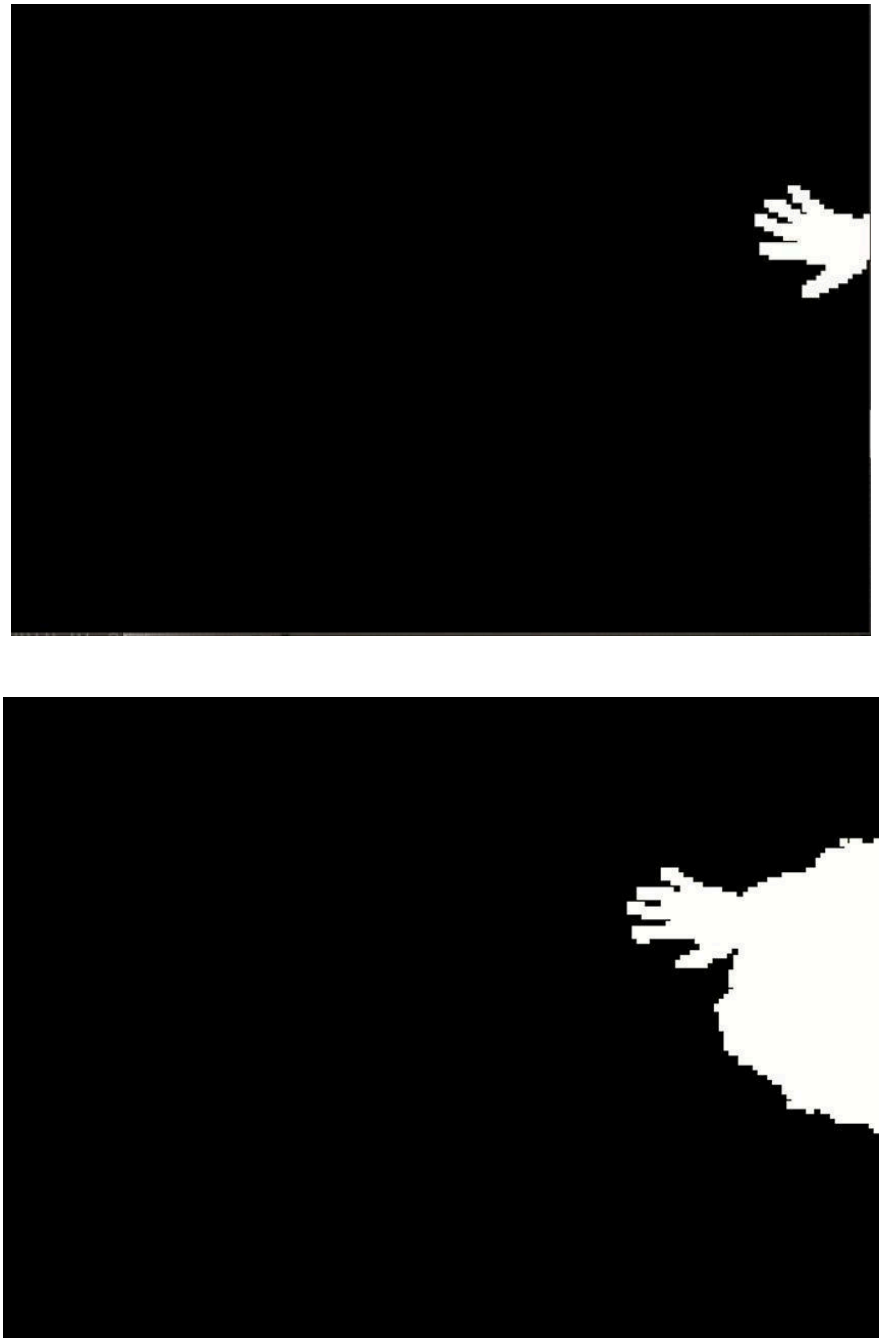


Figure 12 : Binarized image from infrared - body not included vs body included

In these pictures, firstly a convex hull is found and then convexity defects of this hull are obtained. Convexity defects are the points farthest from the convex points. By using these points and measuring the angle two types of points are obtained: far points and peak points. While peak points are in places like fingertips,far points are located near to the palm. The corresponding points to the binary images can be seen in the following:

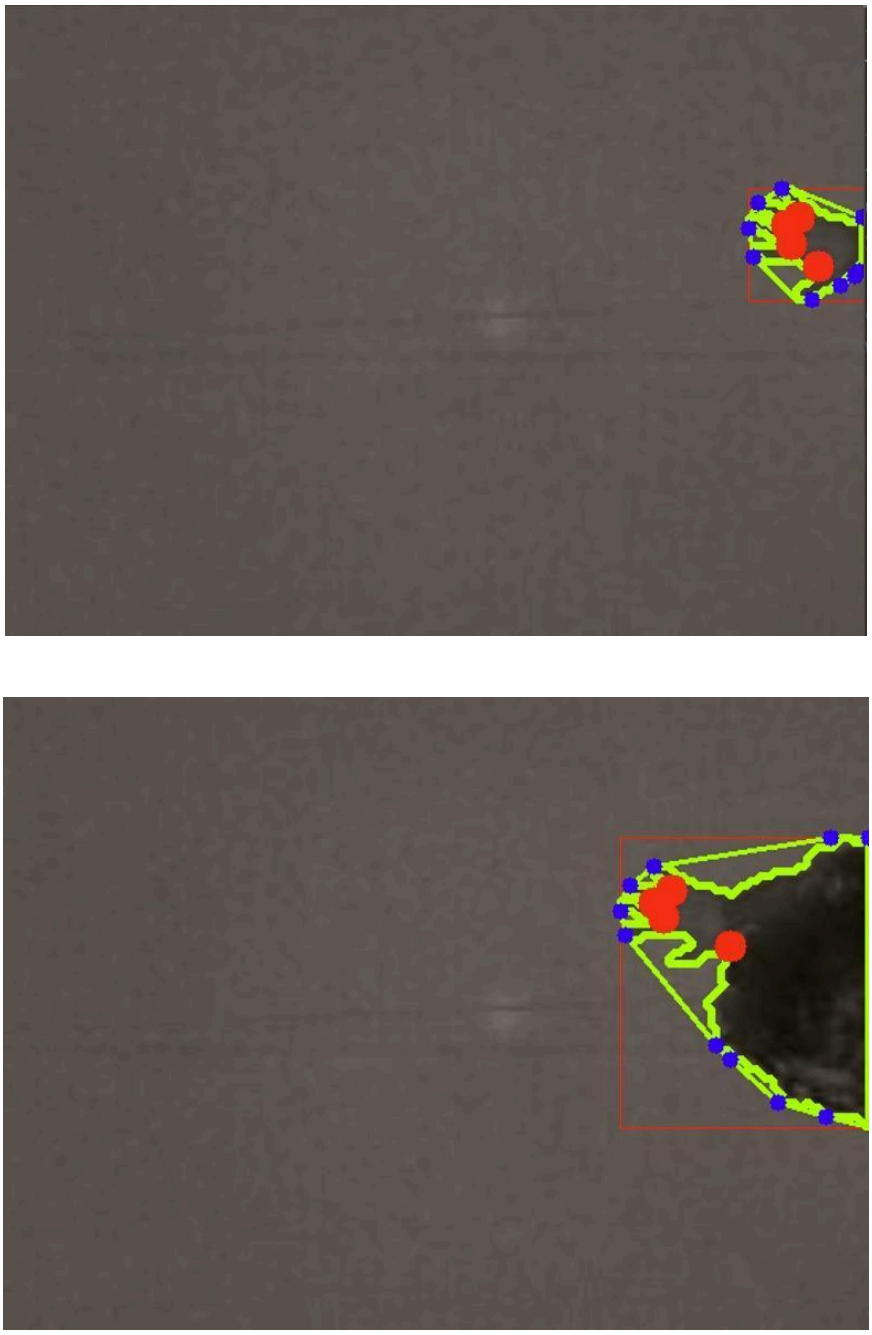


Figure 13 : Detected points with convexity defects

After these detections; it is required to differentiate the peak points that are not located in the hand. At this stage, the distances between the points are evaluated. After the hand is detected correctly, a bounding box is drawn around the hand, and this bounding box is given to the object tracking algorithm, norfair. By using tracking, the mouse moves smoothly and in accordance with the hand movement. In short, the used tools are Python, OpenCV, Norfair and Pyautogui.

- Mouse Movement

  Mouse movement is achieved using the bounding box tracking with norfair. While determining the mouse position, first, arm layout is determined by the positions of the peak points. If the layout of the arm is in y-axis, the difference of the minimum x value of peak points and maximum x value of peak points is lower than the difference between the minimum y value of the peak points and maximum y value of the peak points. It is the vice versa if the layout of the arm is in the x-axis. Then, it is determined which side of the table hand gets in (right, left, up, down). For this, peak points' density is used (Hand has more peak points closer to each other, than the peak points on arm). Then, mouse position is decided according to the hand position, to see the mouse in front of the fingertip.

- Click

  Click is called with pyautogui.click() function, when mouse position does not change for 15 frames (nearly 5 seconds). Since peak points change even if the hand does not move, there is a stabilization needed. So, if the bounding box position is not changing 10 units through the x or y axis, the mouse position is not changing. Therefore, the click gesture works properly.

- Panning

  Panning gesture is implemented by far points. If far points are increased and occur more than 3, by opening all fingers of hand and this position lasts for 5 seconds, pyautogui.MouseDown() function is called and panning has started. Until the hand turns into the click gesture and waits for 5 seconds, the map can be panned. Then, if the click gesture is done, if the mouse is down (user is in the panning gesture), the pyautogui.MouseUp() function is called and panning is terminated.

- Zoom in/out

  Zoom in/out gestures are implemented by looking at the contour area sizes. If the contour area increases to the area of two hands, the contour area is divided to two bounding boxes and zoom in/out is detected by the increasing or decreasing distance of these areas.

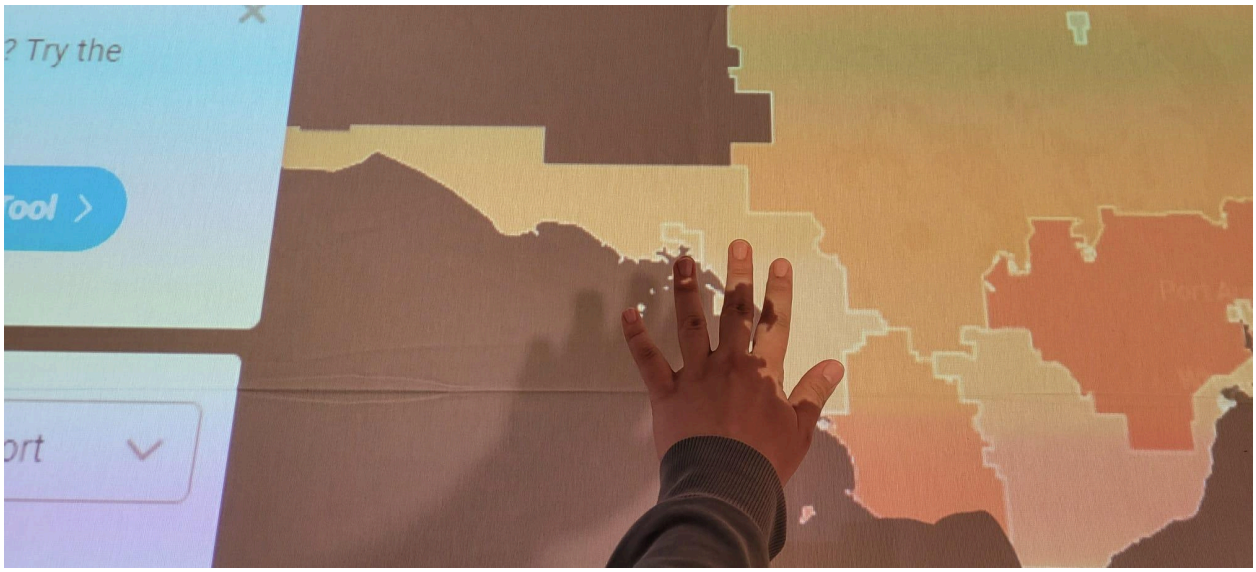


Figure 14 : Panning

Figure 15 : Mouse movement and click

4. RESULTS & DISCUSSION

As a result, a system that can give commands to the computer by interacting with the table is developed. Mouse movement, click, panning and zoom in/out are working successfully now. Although there are times that we move far from the main objective, the system we finally obtained is in full harmony with the system originally planned. Therefore, in this respect, it can be said that the project is progressing successfully.

There is a similar project done in our school, but a language called processing was used while it was being done, and there is a library that can find the hand directly from above. Since we do not use processing in our project, we focused on implementing the operations that the library has done ourselves. Migrating the project to python can also be considered as a development, since Processing is a language that is not used much. The demos of the previously implemented versions can be found in the following links:

Mediapipe demo:

https://drive.google.com/file/d/16w65hBX5WRonP_Py8bhkize3AlzuuNTj/view?usp=sharing

C++demo:
https://drive.google.com/file/d/1DnnEQT2i7qMsM5jAdRmlahfy4CvOrBJW/view?usp=sharing

Depth demo:

https://drive.google.com/file/d/10pREY66YxnKQ5hYOfTetDMMdkI9c0rZo/view?usp=sharing

For the implementation side, if we could move to the lab setup earlier, we can see the inappropriateness of mediapipe and take action accordingly. In this project, we have tried lots of methods and technologies. It can be obviously said that this project teaches us how to change plans, take actions and implement them in a short period of time.

5. IMPACT

As discussed before, this project aims to improve communication between parties using online connections. So, the impact we want to make is to see better and more clear communication in online meeting setups. This project can also influence other projects that take place where the interaction is important, such as architecture, art displays or museums. There is currently no freedom to use issues, and it is not expected in the near future.

6. ETHICAL ISSUES

This project is an implementation of gesture recognition using the libraries of Python. We used example implementations in the internet sources and our previous knowledge. These implementations are open-source codes which can be used for any purpose. If the design part is considered, the idea of the gesture recognition by Kinect camera from the top, to control the map on the table is an idea produced by our professor and his team, in previous years, and started its implementation. However, while that project aims architectural purposes, this project's aim is to design a more realistic online meeting environment. Additionally, the programming language being used is completely different. Therefore, there are no ethical issues about the design concept or technical part of this project. In this project, there are no ethical issues to consider for the health of society, material used or privacy of personal life.

7. PROJECT MANAGEMENT

Initial plan

- Gesture recognition

- Implement necessary gestures with the existing hand recognition model.
- Arrange the changes in gestures with respect to changing depth.

- Speech to text recognition

- Find an efficient speech recognition code.
- Improve its weak points

- Define what to do with recognized words or sentences.

- Emotion recognition
- Improve the code to detect face.
- Implement emotion models.

Elaborations on Initial plan

Plan of the project had 3 main parts at the beginning. Gesture recognition, emotion recognition and speech to text recognition. In the first period of the project, we had literature search on these topics to form the general structure of our project. Similar implementations have been tried and previously implemented projects which have similar aims to this project have been studied. These research on the topics helped to make a plan on how long the implementations will take and which parts are more beneficial to implement first.

After these processes, project details are formed and new details are added to the project. The reason for these developments is to have a better meeting platform as in the aim of the project. At the end of the first term, the objective became only gesture recognition, since it forms the main structure of the project. For this reason, in the second term the planning is done accordingly.

Final Project Plan

This plan is done in the half of the second term. This can be an example of how the project plan is changed throughout the term. At that time, we experienced problems with the mediapipe and it was tried to be solved by cropping the image in accordance with the location that the person is located.

The final intended plan in the half of the term can be explained in short as following:

11-18 April

- Solving problems with person tracking and differentiating individuals with their hands'

information

18-25 April

- - Implement gesture recognition with the camera looking from upwards

25 April - 2 May

- - Avatar implementation improvement and combination with Atlas

2 May - 23 June

- - Improve the gesture recognition algorithm

However, now mediapipe is not used and the whole implementation technique is changed. The coordinates are purely tried to be found with the code that we implemented, not with the help of any library. Today's main objective is to get rid of the false peak points which are not located in the hand and according to that forming the correct bounding box that helps us to track the hand smoothly and consistently.

Elaborations on final plan

After the initial plans, tasks are more specified and changed according to the benefits of the project. We had a few implementations which satisfied the needs, however problems occured on the laboratory setup (camera angle, camera and table distance, etc.). Therefore, the last implementation is created which can detect the hand from the camera looking upwards and can click and move the mouse. In addition, avatar implementation has been done, however, it will result with some other project group because tasks are shared and changed in some phases of the project.

As a result, project management was successful in terms of task definitions and general design of the project. More briefly, changes in the design were improving the project.

8. CONCLUSION AND FUTURE WORK

This project has an important place in the improvement of online meetings, to move them further through the realistic and efficient environment in terms of understanding each other's emotions, behaviors and what they told them. In addition, our gesture recognition implementations are very crucial to understand the problems that will be faced while working on this topic. While having the camera at the top, gesture recognition and separating individuals are problematic because most of the existing libraries are using the body to detect hands. Camera from top cannot detect the body. Another restriction is the long distance between camera and hand which causes problems with the libraries which will be used.

On the next steps of this project, multiple people's hands must be detected and assigned to separate them during their movements on the table. Different individuals will be able to move their hands on the table but not all of them will have control on the cursor. Only one individual will have control on the mouse and the others only will be able to show some area on the map with their avatars. Then, NLP assistant and live videos of other participants on the separate screens will be combined to this gesture recognition setup. An NLP assistant will show the topics on the screen which participants are talking about. Crucial part here is that what the leader speaker talks about is more worth sharing on the screen than the other participants' speeches. Live videos of other participants will be shared on the screens to increase the reality of the meeting. After these, making analysis on the participants may be beneficial for the meeting owners to analyze their weaknesses and improve them. Also, if one part of the meeting Is presenting their projects or products to the other side, they may know better that their product has been liked or not. Analysis on the participants can be done by the algorithms that analyze the vibration and tone of the sound.

To sum up, these next steps include improvement of existing hand gesture recognition implementation and adding further features for the meeting environment. While improving the gesture implementation, the obstacles we face in the development of the project should be taken into account and evaluated well.

9. REFERENCES

Atlas of Opportunity. Retrieved from https://opportunity.mit.edu/

Beyeler, M., Gevorgyan, D., & Mamikonyan, A. (2020). *OpenCV 4 with Python Blueprints - Second Edition*. [S.l.]: Packt Publishing.

Cicirelli, G., & D'Orazio, T. (2017). Gesture Recognition by Using Depth Data: Comparison of Different Methodologies. *Motion Tracking And Gesture Recognition*. https://doi.org/10.5772/68118

Hands. (n.d.). Retrieved 6 January 2022, from https://google.github.io/mediapipe/solutions/hands.html

Kaya, E., Alacam, S., Findik, Y., & Balcisoy, S. (2018). Low-fidelity prototyping with simple collaborative tabletop computer-aided design systems. Computers & Graphics, 70, 307-315.

Özerdem, M., & BAMWENDA, J. (2019). Recognition of static hand gesture with using ANN and SVM. *DÜMF Mühendislik Dergisi*, *10*(2), 561-568. https://doi.org/10.24012/dumf.569357

10.

pykinect2. (n.d.). Retrieved 6 January 2022, from https://pypi.org/project/pykinect2/

gRPC. (n.d.). Retrieved from https://grpc.io/docs/languages/python/basics/

Hormi, Sami. Multi-Camera-Person-Tracking-and-Re-Identification. (2021). Retrieved 10 April 2022, from https://github.com/samihormi/Multi-Camera-Person-Tracking-and-Re-Identification