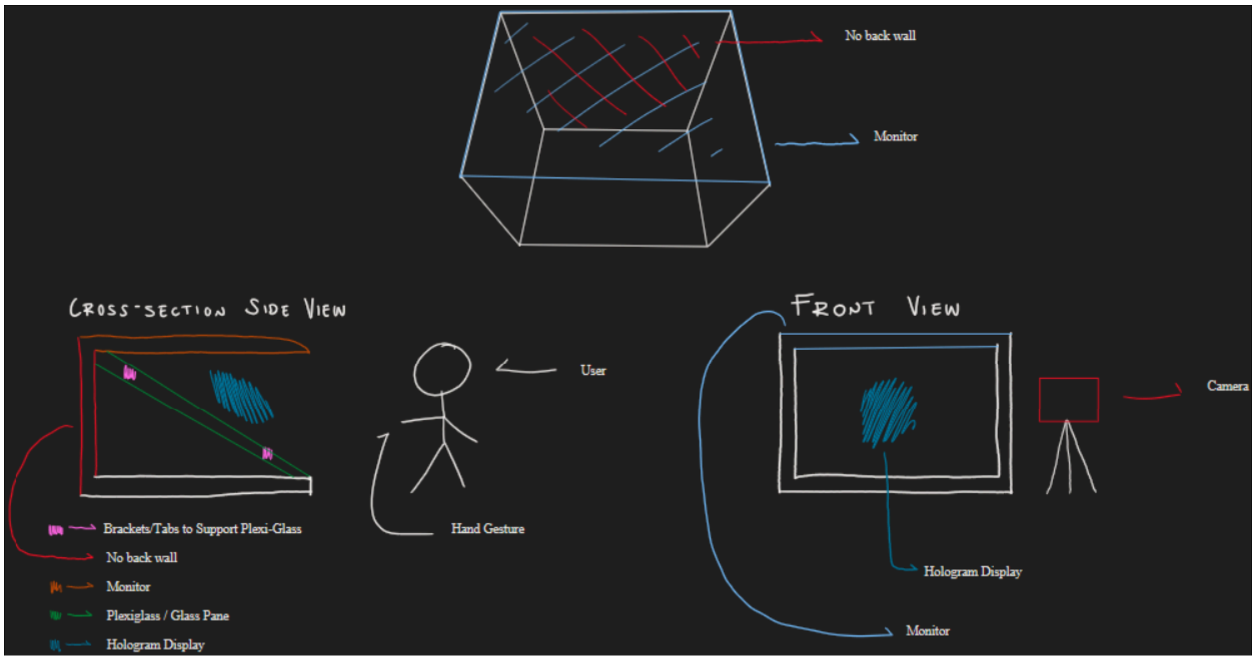
Senior Design Project Final Report

HoloTouch

Abraham Garcia (Computer Engineering), Yanai Avila (Computer Engineering), Elyver Turla (Electrical Engineering), Andrew Call (Electrical Engineering)

Faculty advisor: *N/a*

Final Updated System Diagram:



Functions, Modules, Parts That Have Been Accomplished in the Past Months:

May - September 2024

- Updated system diagram and hologram implementation. We will do an enclosed box with only the front side open. Integrated within the housing/framing will be the front-facing camera with LED lights to help with camera detection, then our monitor (which is used for our image that will be converted into the hologram), and wires hidden within for hardware connectivity. Our Nvidia Jetson Nano Developer Kit should be mounted on the rear side of the box. Inside the box, our acrylic sheets convert the image to a hologram fitted to the box set at a 45° angle. So far, we have our input device (iPhone/Laptop/etc.) wired and connected outside of the housing, though in the future we plan to somehow integrate it within to provide a sleeker, cleaner, and more presentable final design
- Went from 32 GB to 64 GB
- Rather than using two cameras, we decided to stay with one camera
- Regarding the three hand gestures we have, we will be discussing and planning on adding an extra (at least) two additional gestures to implement
- For Roboflow, we might need to purchase a legitimate subscription, which may exceed our budget. Will be discussing with Dr. Ming and within ourselves before making that decision
- Looking into connecting Yolov5 and PyAutoGUI (more in-depth information will be provided at the bottom of this document, under "Application Programming Interface", as well as a screen recording done)

September - October 2024

- Was initially going to utilize a 3D printer for our housing/box, though decided otherwise. Instead, we will be building a housing out of wood (that will be covered with black cloth for lighting and cosmetic purposes). First model has been created and will further be explained under the "**Functions, Modules, Parts That are in Progress**" section as well as pictures and videos included.
- Created first (or potential) final model of hologram and display housing. More information, as well as pictures and videos can be found under "**Functions, Modules, Parts That are in Progress**" on page 5 - 6.

- We were originally using Roboflow for dataset preparation and management and Yolov5 for training and deployment of our model, but we have decided to completely move away from this. We tried having a combination of both yolov5 and MediaPipe, but this proved to be difficult and unnecessary. Instead, we decided to use MediaPipe independently, and we were able to successfully implement the main gestures we wanted including zoom in, zoom out, rotate left and rotate right.
- We successfully implemented some navigation controls using hand gestures. The main one is the cursor. We knew that if we wanted users to have basic control of a computer through a hologram, then they would have to be able to control the cursor with their hand. Our code allows them to use intuitive hand gestures (as if they were dragging and clicking on a physical mouse) to use the cursor.
- We originally wanted to use NVIDIA's NeRF to present what our code can do, but we quickly realized that this technology was dedicated more to video making with a preset camera movement. Then we thought about using the Unreal Engine game engine, but we knew that this would be computationally heavy. We finally figured out a great way to show what we wanted to show without having to sacrifice speed and time, and the solution was using Google Earth. We successfully implemented our code using Google Earth, and this is how we will show the 3D object manipulation part of our project.

October - December 2024

- Updated and finalized hologram and display housing (version 2.1). Removed back wall to give holographic effect/floating 3D image (users are now also able to physically stick their hand through the back of the housing to further see holographic/floating image effect). Since drooping was an issue from the last housing (version 2), added plexiglass supports to prevent this. Additionally, the back wall was a structural component, so additional bracing was required to account for the wall removal.
- The list of hand gestures our system can detect implemented was finalized

** Some of Standardization Organizations are:

- Institute of Electrical and Electronic Engineers (IEEE)
- International Organization for Standardization (ISO)
- International Electrotechnical Commission (IEC)
- NFC Forum for NFC-related Standards
- American National Standard (ANS)
- American National Standards Institute (ANSI)
- Society of Automotive Engineers (SAE)
- National Standards Network (NSSN)

Updated Table of Specification & Design Restrictions:

Function module	Specification (use bullets)	Why Choose this Specification?
USB Camera	● 4056 x 3040 ● 75° Field of view ● 30fps	We chose the most affordable camera since it was not too important to have the best camera.
Monitor	● 1080p ● 75 HZ	This monitor will allow us to scale the hologram to a larger size. The refresh rate of 75 Hz will give the appearance of smooth motion. Many different sizes, resolutions, and refresh rates would suffice.
Processing	● 10 Core Intel CPU ● Nvidia Gtx 1070 ● 32 GB DDR4 Memory	Over the evolution of the project, we realized that the Jetson Nano did not have the processing power we required. While it was able to detect the hand gestures, it ultimately did not give us the smooth experience that we desired. To eliminate hardware limitations as much as possible, we used desktop processing hardware. This allowed us to focus on adding functionality with gestures without concern of slow operation or system crashing.

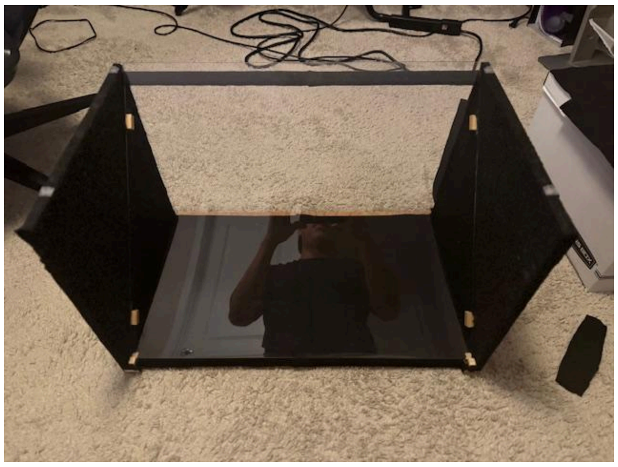
Updated Table of Necessary Standards:

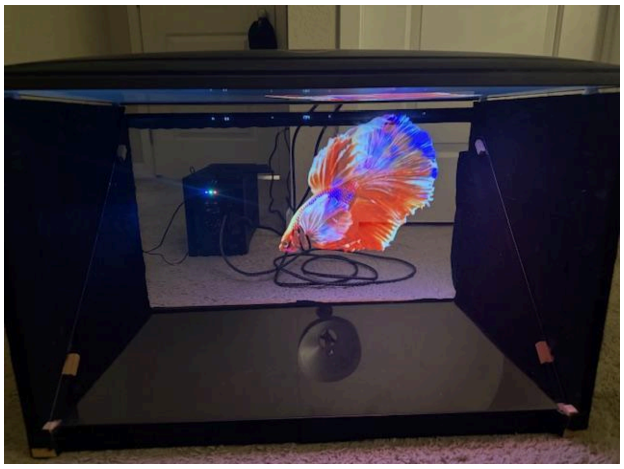
Standard	Description	Link or References	Why is this Standard Employed/Necessary?
HDMI	Any regular standard for image display		Need to display a device screen
IHMA	- Principles include maintaining integrity, respecting intellectual property rights, ensuring quality and safety, and striving for customer satisfaction. - Ethical business practices, respect for intellectual property, and adherence to environmental health and safety standards should be upheld. - While not for commercial use or sale, projects should mirror standards followed by industry professionals in holography.	https://ihma.org/wp-content/uploads/2021/10/IHMA-Copyright_guidelines-2017.pdf	Adherence to IHMA principles recommended for custom hologram devices developed as school projects.
USB 3.0	Computer interface to allow connection of peripherals.		The standard allows easy connection of peripherals and devices without the need for extensive configuration or adjustments.

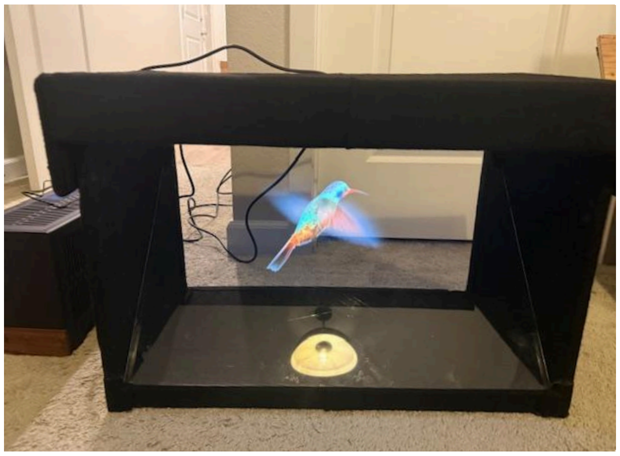
Finalized Functions, Modules and Parts:

Function 1: Hologram

The final version of the holographic display consists of a rectangular box. There is an opening in the front, rear, and top of the box. The opening at the front of the box allows the user to view the projected image. The opening at the top of the box houses the display which projects an image downward onto an acrylic panel that sits at approximately 45°. The opening at the rear of the box is to make the image appear as if it is floating in free space, which creates the holographic effect. There is a top section in the same rectangular shape which hides the display and wires. The walls of the box are covered in black cloth, which minimizes the visibility of each wall.







Function 2: Application Programming Interface

The APIs that we used are PyAutoGUI and pynput. The PyAutoGUI API is used for automating GUI interactions including moving the mouse, right-clicking, left-clicking, etc. The pynput API is used to control and monitor input devices. Together, these APIs allowed us to map gestures to keyboard and mouse actions. Each gesture is mapped to a corresponding command signal and the command signal is sent through these APIs to interact with the computer. The computer then executes the commands.

In the detect_gesture() main loop:

- If a gesture is detected, then execute a PyAutoGUI/pynput command
    - Examples:
        - mouse.press(Button.left)
        - pyautogui.moveTo(1527, 1027)
        - pyautogui.hotkey('ctrl', 'alt', 'tab')
        - keyboard.press(Key.up)
        - pyautogui.scroll(-30)

Function 3: MediaPipe Hand Tracking

<u>What is MediaPipe?</u>

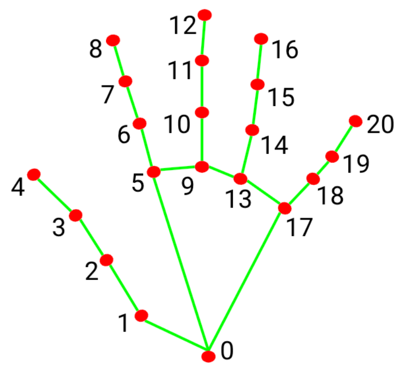
MediaPipe is a open source framework developed by Google for building perception pipelines, which involve processing and understanding visual and audio data. It provides a collection of machine-learning models for a wide range of computer vision tasks like face detection, hand tracking, object detection, pose estimation, and more.

MediaPipe provides an advanced solution that uses machine learning models to detect and track hands in images or video streams. It identifies 21 key landmarks on each hand, enabling precise and real-time gesture detection. This is the model we used.

Unlike other models, this model did not require us to do any additional model training. Our code simply took advantage of an already existing trained model, and doing angle and distance calculations to map gestures.

MediaPipe Landmarks:

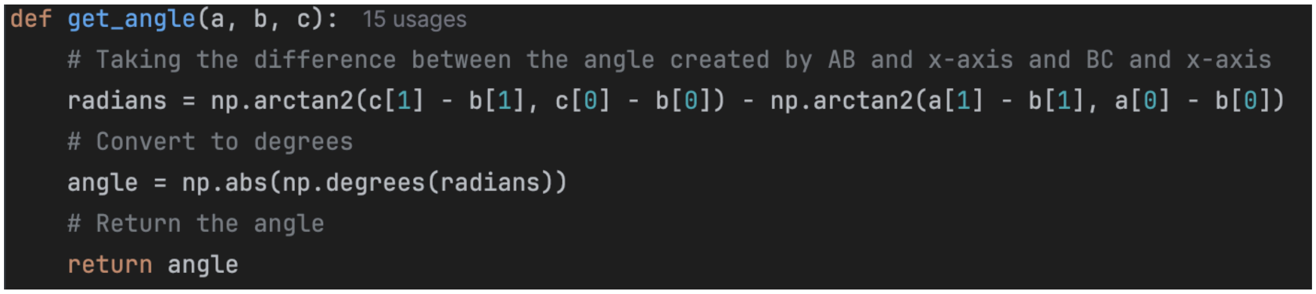
As mentioned, the MediaPipe hand-detecting model we used identifies 21 key landmarks. A landmark is a particular point on a hand. These landmarks are configured as such:



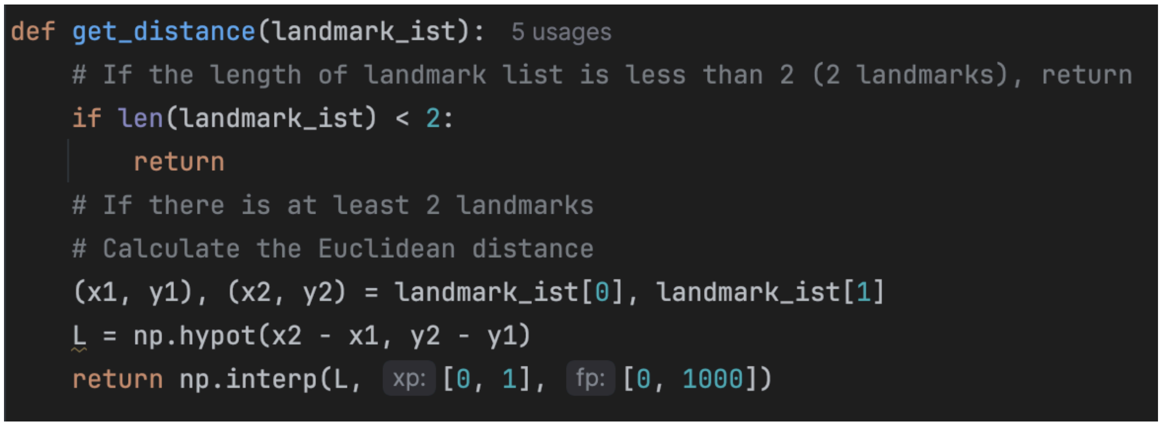
0. WRIST
1. THUMB_CMC
2. THUMB_MCP
3. THUMB_IP
4. THUMB_TIP
5. INDEX_FINGER_MCP
6. INDEX_FINGER_PIP
7. INDEX_FINGER_DIP
8. INDEX_FINGER_TIP
9. MIDDLE_FINGER_MCP
10. MIDDLE_FINGER_PIP
11. MIDDLE_FINGER_DIP
12. MIDDLE_FINGER_TIP
13. RING_FINGER_MCP
14. RING_FINGER_PIP
15. RING_FINGER_DIP
16. RING_FINGER_TIP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

Python Utility File For Detecting Hand Gestures:

We obtained a Python utility file (util.py) from a YouTube video tutorial repository. This code calculates the angle and distance between two landmarks. This allows us to create conditions for different computer actions. For example, if the angle for landmarks on a finger is less than 90, we know it is bent. If it is over 160, we know it is straight. For our zoom features, we used the get_distance() function to measure the distance of our pinches as well as set the thresholds for when fingertips touched each other.

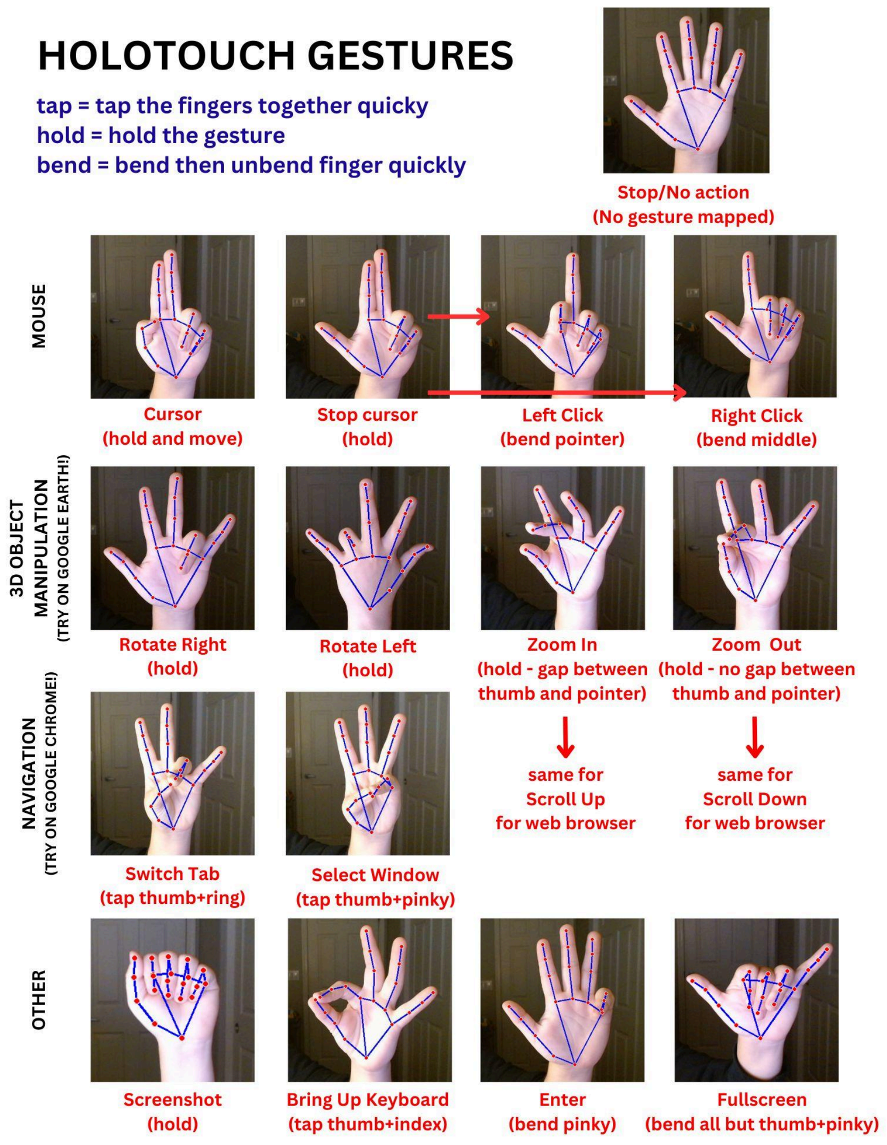


```
def get_angle(a, b, c):  15 usages
    # Taking the difference between the angle created by AB and x-axis and BC and x-axis
    radians = np.arctan2(c[1] - b[1], c[0] - b[0]) - np.arctan2(a[1] - b[1], a[0] - b[0])
    # Convert to degrees
    angle = np.abs(np.degrees(radians))
    # Return the angle
    return angle
```



```
def get_distance(landmark_ist):  5 usages
    # If the length of landmark list is less than 2 (2 landmarks), return
    if len(landmark_ist) < 2:
        return
    # If there is at least 2 landmarks
    # Calculate the Euclidean distance
    (x1, y1), (x2, y2) = landmark_ist[0], landmark_ist[1]
    L = np.hypot(x2 - x1, y2 - y1)
    return np.interp(L, xp: [0, 1], fp: [0, 1000])
```

Function 4: Hand Gestures



1. Cursor
2. Stop cursor
3. Left click
4. Right click
5. Rotate right
6. Rotate left
7. Zoom in
8. Zoom out
9. Switch Tab
10. Switch window
11. Scroll Up
12. Scroll Down

13. Screenshot
14. Bring up Keyboard
15. Enter
16. Fullscreen

The basic navigation gestures are #1-#4. Thea 3D manipulation gestures are #5-#8. The navigation gestures for web browsers and applications are #9-#12. Additionally, we had some other gestures that did not fit into these categories #13-#16.

Note 1: When the thumb is in, the mouse cursor moves. When the thumb is out, the cursor stops. Therefore you can click either when the mouse is moving, or you can stop the cursor and then click.

Note 2: PyAutoGUI and pynput were used for these computer actions. Some of the functions were better and easier to use for one API than the other, so we decided to use both of these libraries instead of limiting ourselves to one of them.

VIDEOS:

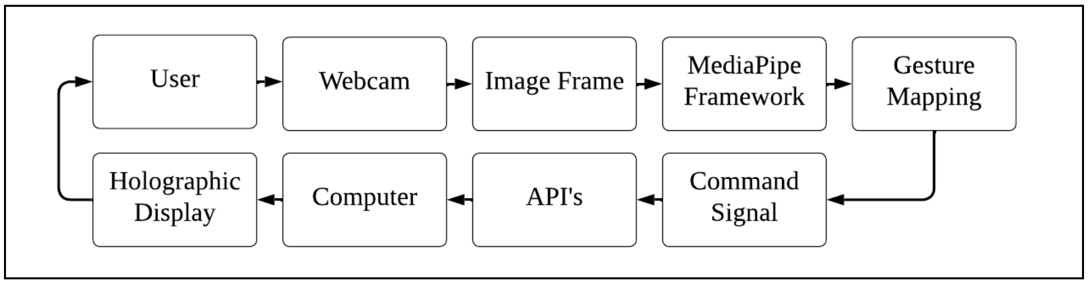
Video from progress report 1: https://youtu.be/3RVkRhotQWI

Video from progress report 2: https://youtu.be/Glf8zG6envc

Link to Playlist with our final demos:
https://www.youtube.com/watch?v=lsw2k-dQBGs&list=PLAhMgX9etoIUE-2fuqYnozIpesjFgpQIM

Link to our gitHub with the code: https://github.com/YanaiAvila/HoloTouch.git

Putting it all together:



User: The user performs specific hand gestures.

Webcam: The webcam captures the user's gestures as a video stream.

Image Frame: The video stream is divided into individual image frames.

MediaPipe: MediaPipe processes each frame to detect and track hand landmarks.

Gesture Mapping: The detected landmarks are analyzed to recognize specific gestures.

Command Signal: Each gesture is mapped to a corresponding command signal.

API: The command signal is sent through an API to interact with the computer.

Computer: The computer executes the command (e.g., zooming or rotating).

Holographic Display: The result of the command is displayed on the holographic projection.

User sees display and reacts to this feedback, restarting the loop.

Challenges (if any):

1. We encountered a bug when dealing with using the Windows Touch Keyboard. Everything worked fine, and the user could type and use the gesture for enter, but for some reason, it would not let us close the keyboard window even if we "left clicked" on the close window button. It was not so easy to debug because sometimes it worked and sometimes it didn't, so it would take a while to figure out what caused it to not let us close the window.

Potential Solutions:

1. Troubleshoot what may be cueing this to happen sometimes.

Questions (if any):

N/a

Total Hardware & Budget Cost so Far:

part description	function	amount needed / unit price	subtotal	purchase link	datasheet link
NVIDIA Jetson Nano Developer Kit	Deep learning and data processing	1/149.00	161.48	https://www.amazon.com/dp/B084DSDDLT?starsLeft=1&ref_=cm_sw_r_apin_dp_JH21Z474A7PCC7E5NEHJ	
Arducam 12MP 477P Motorized Camera for Nvidia Jetson Nano	Gesture/hand motion capture	2/79.99	189.98	https://www.uctronics.com/imx477-12mp-high-quality-camera-motorized-focus-autofocus.html	

ADTDA 2 Pieces 1/8" Thick (3mm) Acrylic Sheets	Convert image into hologram	2/9.99	18.41	https://www.amazon.com/ADTDA-Acrylic-Plexiglass-Protective-Projects/dp/B0C2C3JC21/ref=sr_1_4?crid=L8EFAVBVLLGM&keywords=plexiglass%2Bsheets&qid=1707679203&sprefix=plex%2Caps%2C147&sr=8-4&th=1	
Wood and brackets	Frame for the hologram. Houses the monitor and acrylic panel.	1/32.72	32.72	IMG_6827.jpg (3024×4032)	
Black cloth	Aids in hologram visibility	1/10.27	10.27		
Keyboard	Control demo computer	1/13.84	15.00		
64GB SD Card	For Jetson Nano	1/9.03	9.79		
Webcam Camera w/ Light	To capture hand gestures to be applied to hologram	1/47.99	52.01		

Curtain Rings	To hold up curtains to reduce light for darker lighting environment	2/19.89	40.95		
Camera 2 (without light) + tripod	Second camera for second laptop; tripod to hold up main USB camera	1/49.99 & 1/19.99	75.84		
Table cloth	Block light in tent during presentations	17.99	17.99		
Total:		$624.44			

Employed Software/Program Resources (e.g., Open Source Codes, APIs, etc.)

Function	Key features	Source link/references
MediaPipe	Hand gesture recognition	https://github.com/kinivi/hand-gesture-recognition-mediapipe https://ai.google.dev/edge/mediapipe/solutions/guide
Ultramon	Display flipper	https://www.realtimesoft.com/ultramon/download.asp