AI PROFESSOR

A system for Air Writing utilizing depth-based recognition and dynamic-tracking

|**Team Name:** Chennai Pullingos |
| **Region:** Central Asia + Southern Asia |
| **Team Type:** General|

INTRODUCTION

The act of knowledge sharing has taken up several forms throughout history with books being the major medium for several decades. Now, in this digital age, documents are shared electronically between groups of people. We seek to revolutionize this field yet again. With this goal in my mind, we give you:- ***AI Professor.***

Let me take you back a few months to the inception of our idea. The pandemic engulfed the entire globe, all the easy and mundane tasks were now ported to the digital realm. This transition was hard and people were finding new ways to make things better. One challenge we noticed was in the domain of Education. All the classes were now online and teachers were struggling to disseminate knowledge to their students. They were so used to drawing and writing on the blackboard in class, that they found it hard to communicate the same concepts virtually in a verbal manner. Even with the assistance of visual tools such as PowerPoint presentations, there was this huge void between the lines of communication between users. This is where we saw an opportunity to bridge the gap. We planned to make a product that would enable instant transfer of knowledge using the power of Computer Vision. When we came across the OpenCV AI Competition, we realized that the OAK-D device would provide us with all the necessary tools to tackle the problem at hand.

Our product enables users to write simply with a wave of their fingers. With the help of gesture recognition and depth information obtained from the scene, we can also toggle between writing, erasing and tracking modes. To further augment the product, we have enabled choosing different colors using different gestures. This eliminates any special hardware that a teacher might need to buy to teach virtually. This form also seems more natural than writing on a writing-tablet. It also enables quick sharing of the notes that the teacher has written by a simple gesture.

Related Work

The current works similar to ours use a simple contour detection method to segment the hand and then apply template matching to track the object of interest in each frame. This is computationally intensive to perform on every frame and is not very accurate. The false-positive rate while using a template matching algorithm is very high and would lead to unintentional strokes or lines on the screen.

Another way this problem was tackled in previous approaches was by employing only a gesture recognition and key-point detection algorithm on each frame. This again makes it computationally expensive and toggling between writing and erasing/tracking modes is imprecise and hence making it a cumbersome process.

One other limitation in both the works described above was the limitation in terms of space. The user only used the space available in the frame and was not able to expand it further. One way to tackle this problem is to stitch new frames as and when the key-point being tracked moves beyond the boundaries on any of the 4 sides.

Apart from this, another limitation that we saw was the use of GUI based menu-bars which consumed valuable real-estate on the screen. As the functionalities increased, the space needed also increased making it difficult to fit the content within the given space. Also choosing between various options in this GUI based menu-bar lead to a lot of unintentional selections. We hence took the Gesture based approach to create a functional menu-bar.

The lack of the right combination of detection, segmentation, recognition and tracking algorithms was the main reason these works were never really practically viable. Our approach seeks to remedy this with the help of the valuable depth information obtained from the OAK-D device along with it's inherent compute capability.

Proposed Solution

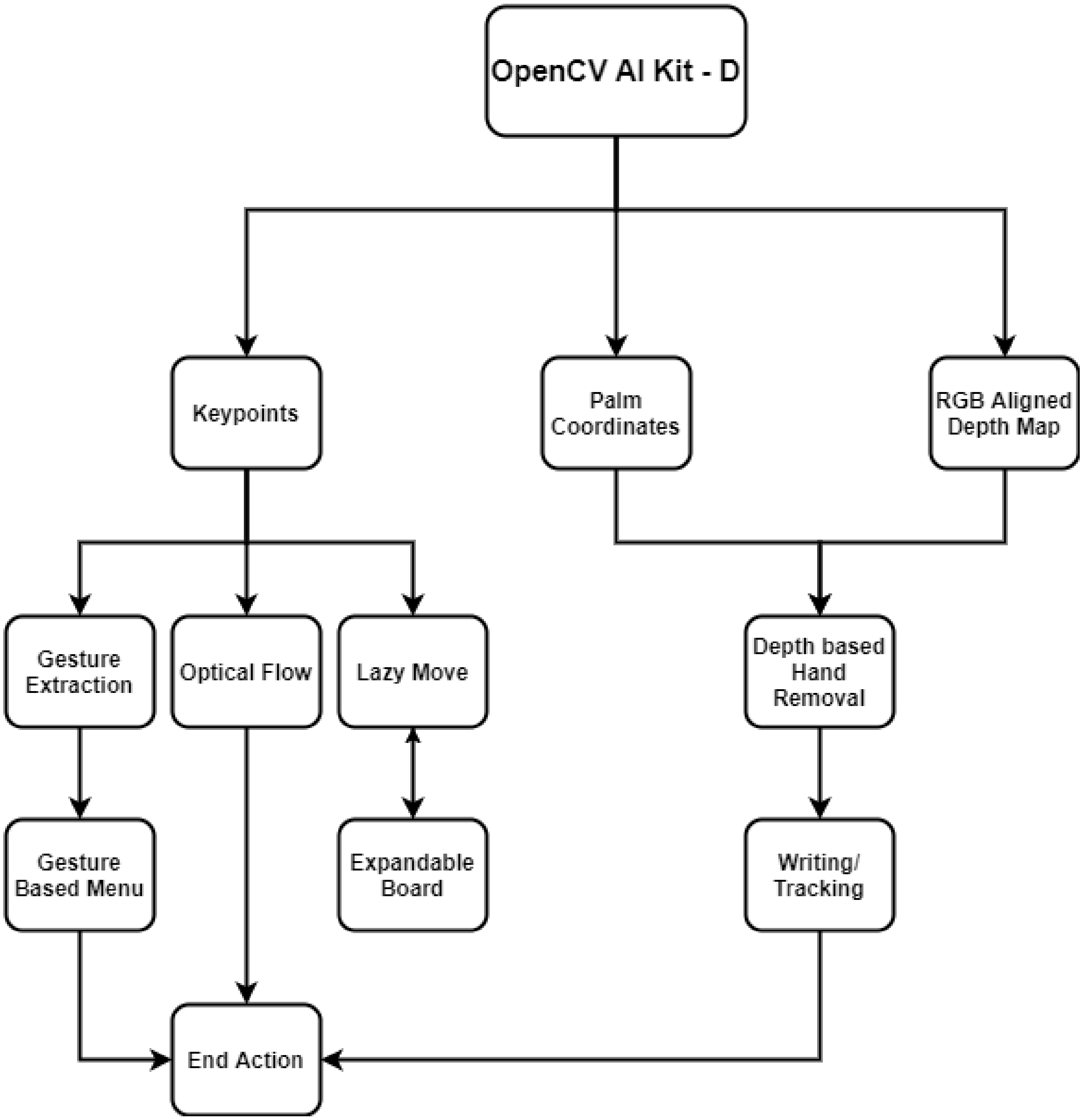


Figure 1. The figure shows the overall architecture of our product. The main inputs to the AI professor are the palm coordinates, RGB aligned depth map and localized hand keypoints which we get from the OAK-D module. Our Depth-Based Hand Removal module, Gesture-Based Menu module and the Expandable Board module work parallelly with the received inputs to generate the desired output, i.e. air writing.

(a) Depth Based Writing and Tracking

The first problem that we have addressed is key-point tracking. The finger key-point has to be efficiently tracked across frames to give a smooth writing effect. Before delving into our solution to this problem, let us first look at previous approaches to the problem. The most basic approach in finger tracking was template matching. But since it was not scale and rotational invariant, additional computations were needed to make it more robust. Incorporating them would in turn, lead to a decrease in frame processing rate. Also, as the finger of every person is different, using template matching was not an universal solution.

The next solution was optical flow, particularly the Lucas-Kanade Optical Flow algorithm. This algorithm tracks a given moving point efficiently. However, it is a bit slow when applied over the entire frame. Another disadvantage of this approach is that it is highly dependent on the given reference point. There is a possibility of errors progressively increasing, and overtime the reference point might drift away from the desired location. This leads to a need for frequent re-calibration.

The final solution is the MediaPipe based key-point localization. As it detects the key-point efficiently between frames, it can be used for air writing. However, one disadvantage with this approach is that it does not have the memory of the previous point and the end output has jitters associated with it. This leads us to our solution.

We have combined the advantages of optical flow with that of MediaPipe, the result of which eradicated their standalone issues. We have used MediaPipe for calibration and optical flow for tracking purposes. This eliminates the jitter problem we had before as well as the progressive error build-up issue with optical flow. As stated previously, using the entire image frame to write is both algorithmically costly and puts strain on the user’s hands. To avoid this, we have defined a small Region of Interest (ROI) so that the user has more control and improved performance. Applying optical flow on this ROI makes it robust. But having a small ROI limits the user to write within the confined space. To remedy this, our product uses a novel ***“lazy move”*** based approach that increases the size of the board dynamically with respect to the movement of the cursor. This enables the user to utilize a larger space using the same ROI. More details regarding this will be divulged ahead.

The second part of our solution is with regards to hand removal. Now that the user is able to write efficiently on the virtual blackboard, how can he/she stop writing? The most famous solution to this problem is using two gestures, one to enable writing and the other one to stop writing. While this does the job for us, it also adds an additional overhead to the user. Frequent changes of the gestures can be confusing to the user. Our solution addresses this problem in a more efficient way.

To see how, let us go back to the concept of traditional blackboards. When the user wants to write, he would place the marker on the plane, i.e., the board and when the user wants to stop writing, he/she would remove the finger out of the plane. This is where our idea spawned. We decided to split the world space into two zones in order to mimic the traditional boards: writing zone and tracking zone. Here, the writing zone is the board, whereas the tracking zone is the space in front of the board. When the user’s hand is inside the writing zone, we implement the writing algorithm and the user can write on the board. When the user’s hand is in the tracking zone, the tracking algorithm is implemented where the index finger is now tracked continuously. This helps the user to understand where the cursor currently is with respect to the writing board.

Another feature of our solution is that the user can decide where his writing zone can be. The threshold is set by the user himself. If at one point of time, the user is at a particular position he can set the writing zone to be in front of his body. If at another point of time, the user decides to move out of that position, he/she can correspondingly shift the writing zone as well. This gives additional flexibility to the user to move around.

To summarize, our product makes use of two deep learning models to efficiently implement the air writing algorithm:

- First, MediaPipe is used to localize the hand keypoints and the key-point of the index finger is used for calibration. This point is provided to the optical flow algorithm in the subsequent frames for smooth tracking. The reason we use a blend of MediaPipe and optical flow is because optical flow is more efficient in tracking the key point over time. As MediaPipe doesn’t have the memory of the key-point from the previous frame, using it for tracking can lead to jitters. Hence, we use MediaPipe for calibration and recalibration purposes.

- Second, we use palm detection and map it to the depth map output to detect the depth of the centre of the palm. We use the depth of the centre of the palm to determine whether the user is currently in the writing zone or tracking zone.

In addition to the above said functionalities, we have also provided a view of the user on-screen to enable him to better position himself with respect to the camera. The image of the user being displayed is inverted laterally to ensure that the movement is in accordance with that of the user.

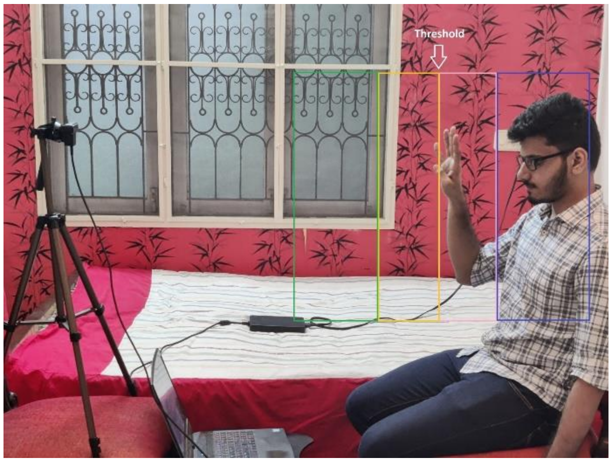


Figure 2(a). This image depicts the user setting the depth-threshold.

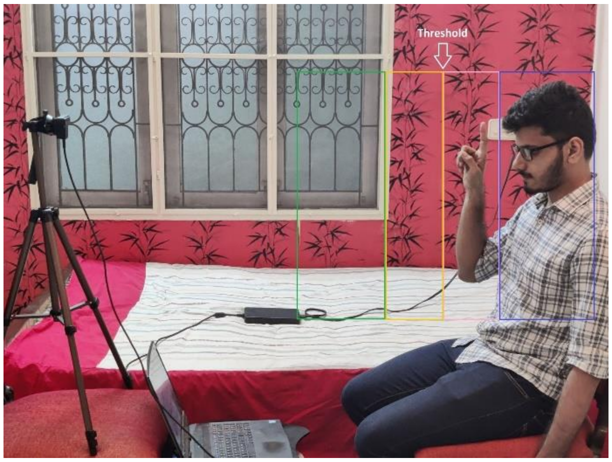


Figure 2(b). This image depicts the user shifting to tracking mode.

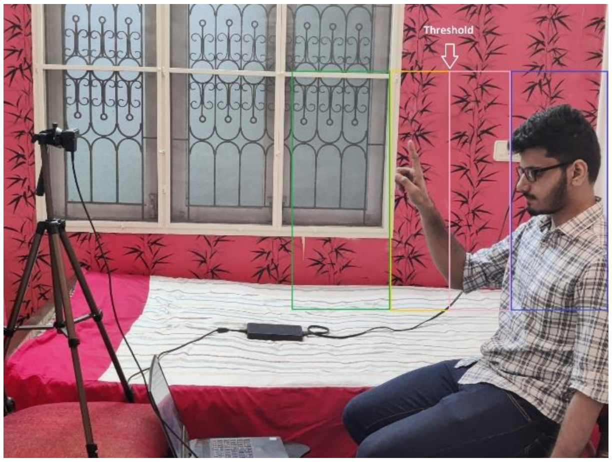


Figure 2(c). This image depicts the user shifting to writing mode by reducing the depth.

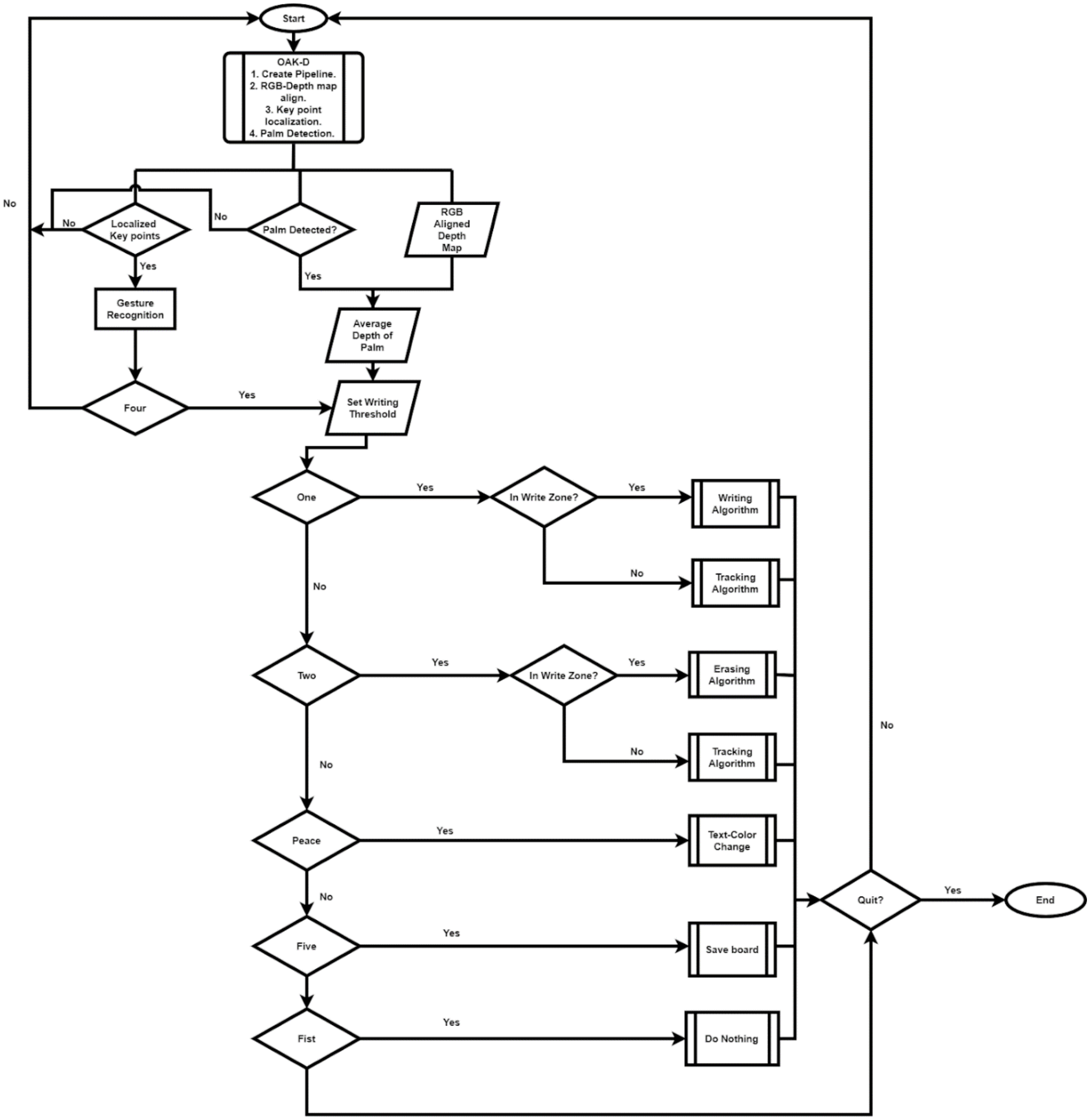


Figure 3. Flowchart of the writing module. We receive three outputs from the OAK-D device, i.e., the hand keypoints, coordinate of the detected palm and the depth map. Using the coordinates of the palm, the depth map is cropped out and the average depth of the palm is calculated. Parallelly, the writing threshold is set by the user. This is done by using the "FOUR" gesture. The depth where this gesture is shown is used as the threshold. In the subsequent frames, the depth of the palm is compared with the set threshold to determine the zone, i.e., writing or tracking. Also, the gesture recovered using the hand keypoints is used to determine the zone of writing. If the mode is writing or erasing, the determined zone is used to check if it's in the writing zone. All the other gestures work independent of the zone. The "FIVE"/ "Do Nothing" gesture is included to give an added amount of control over writing to the user.

(b) Expandable Grid

The area captured by an input camera is fixed and limited. This area is not expandable. Hence applying the tracking algorithm on this limited area will restrict the user to that confined space. As this area is small, it would prove inadequate to write from the user’s perspective.

In order to mitigate this problem, a possible solution is to allow the user an option to stitch new frames in all the four directions and toggle between them. The user can select the direction where the expansion has to happen. These expansion buttons can be strategically placed and dynamically selected with the use of the index finger. However, this would create an additional overhead as the user has to now move back his finger to the original position where the writing had stopped in order to ensure continuity. A solution to this problem can be to use pre-defined gestures to expand this grid. All these viable solutions demand extra effort from the user’s perspective, when compared to the real-life writing scenarios. As such these won't be suitable when factors such as ease of use and adoption are considered. Therefore, any implemented solution should run automatically based on the user’s movements and without requiring any additional input from the user. To address this, we have proposed and implemented a novel solution as given below.

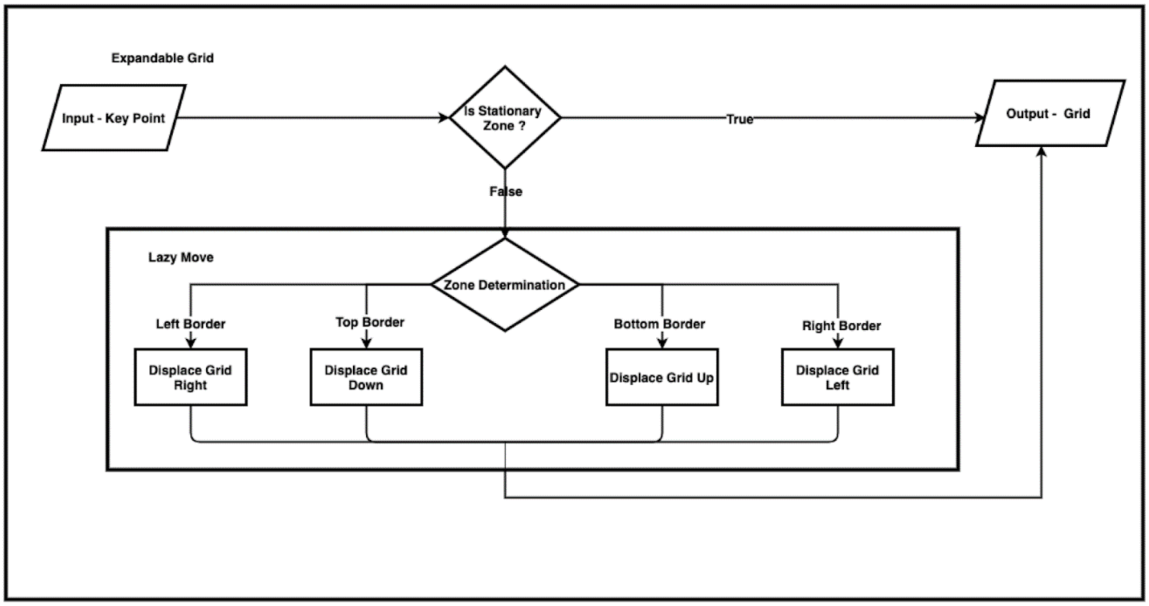


Figure 4. The figure represents the components of the expandable grid system comprising the input key point, output grid and the lazy-move system. The lazy move constitutes the 4 border trigger events and its corresponding reaction done on the grid.

In a real-life scenario, a person adds the notes on the blackboard and physically moves oneself to write on a different section of the board. In other words, the blackboard is stationary and the user navigates around it.

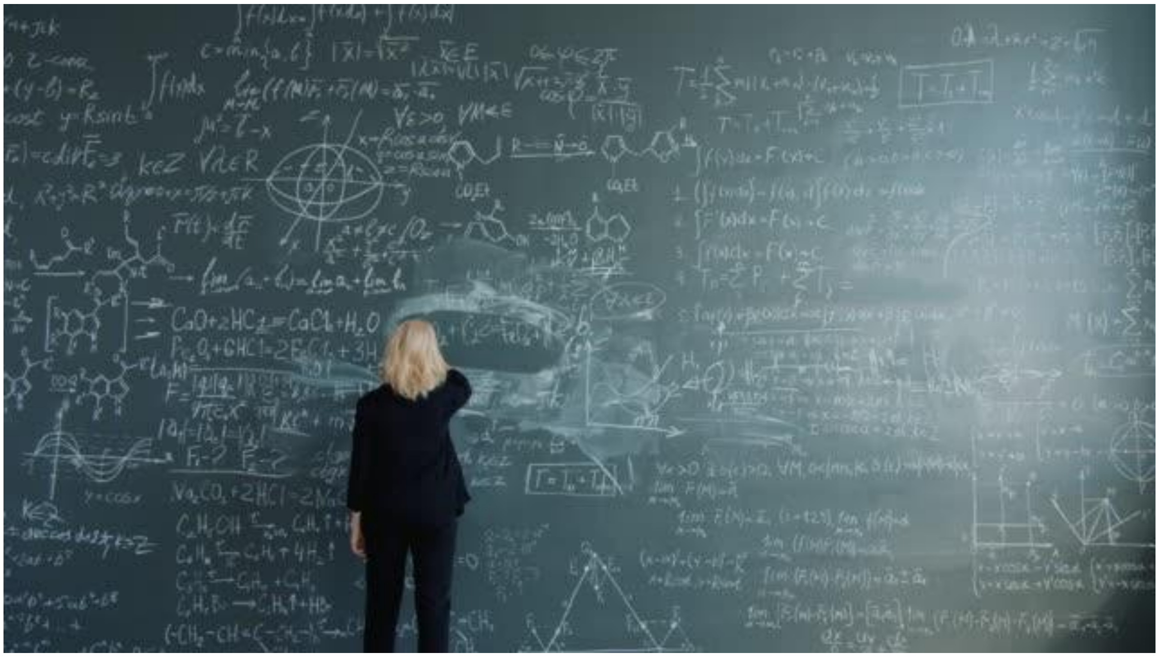
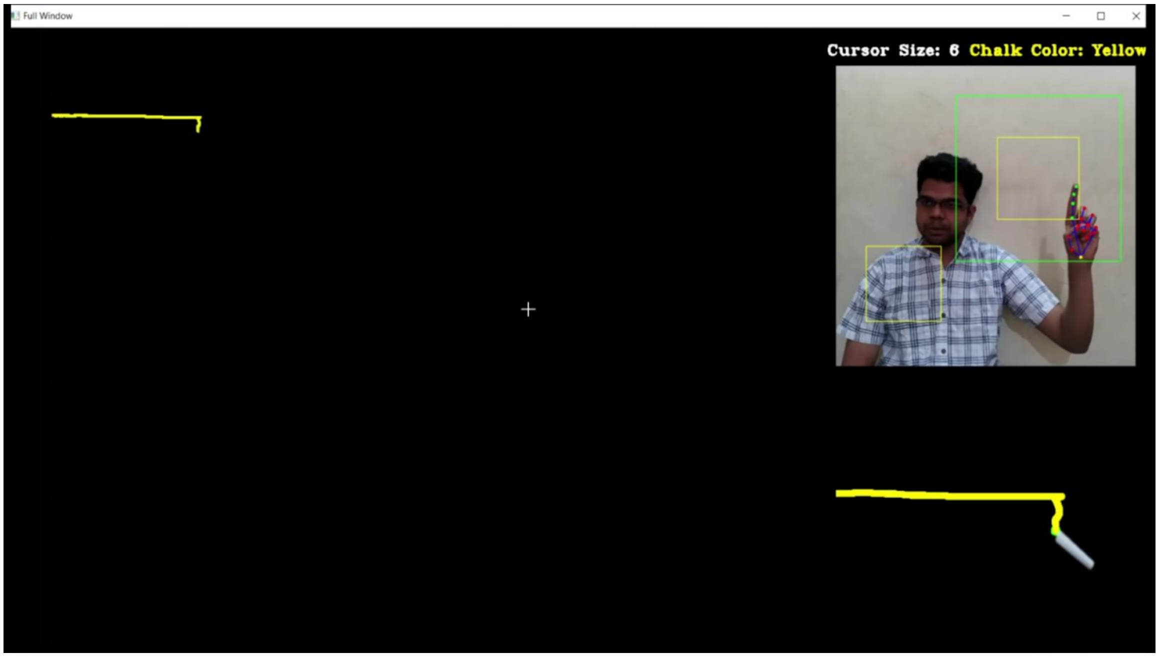


Figure 5(a). The figure represents a teacher utilizing the entire space of the board for writing

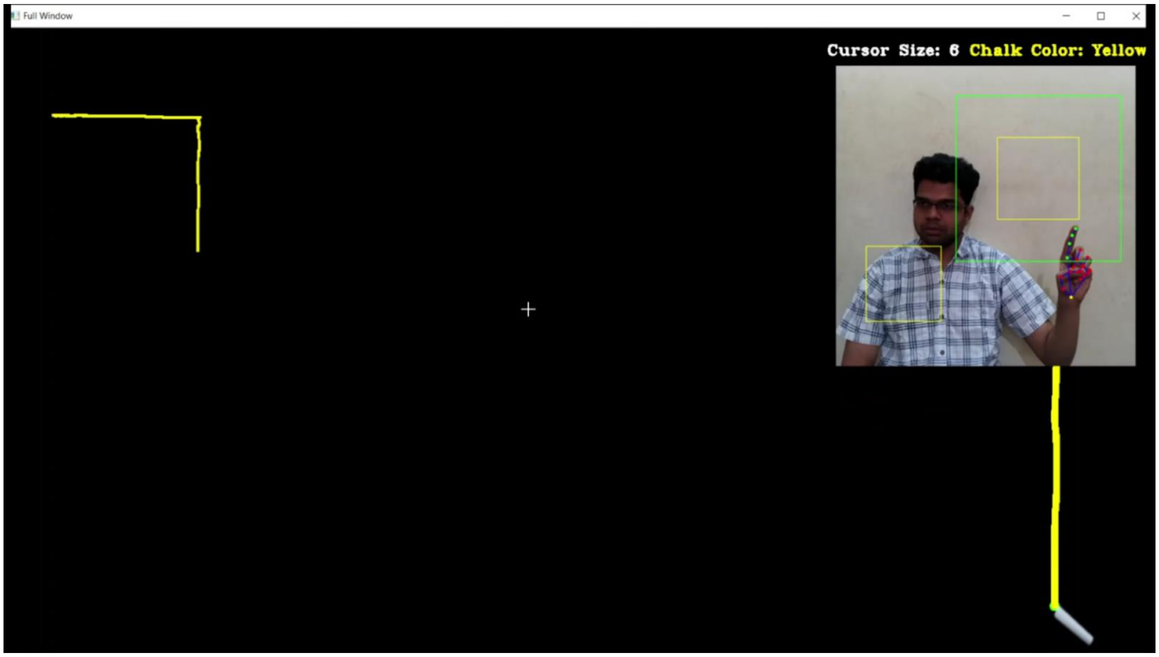
In our proposed solution of ***"Lazy Move"***, the user remains stationary and the blackboard (our canvas) is shifted according to the user's action. The board is divided into two areas called the **stationary area** and the **expandable area**. When the user navigates in the expandable area, the canvas moves in the opposite direction of the user's movement, thereby creating additional space for writing.

The board is separated into four zones representing the directions where the expansion can happen. This includes any combination of the four which enables the user to move diagonally as well. When the user moves to the right, the grid will automatically move in the left direction thereby creating new space for writing on the right. This is similar to the original analogy of the user physically moving across the blackboard. The advantage of this approach is that the user need not move away from his position to create new space. Moreover, the user need not perform any action to move the grid. Instead, the user can continue to concentrate on the writing and the grid can be displaced automatically based on the same. The user would still be able to return to the original location. The expandable grid system with lazy move would ensure that the user can navigate throughout the entire canvas.



Cursor Size: 6 Chalk Color: Yellow

Figure 5(b). The image shows the user moving within the stationary area.



Cursor Size: 6 Chalk Color: Yellow

Figure 5(c). The image depicts the working of the lazy move with the expandable board in which the marker continues to draw downward until the cursor is within the expandable area.

(C) Gesture based Menu-bar

Apart from this, we have also included gesture recognition to enable easy and smooth toggling between different modes. In order to avoid adding a separate resource-heavy gesture recognition model, we have utilized the inherent functionality of MediaPipe to determine various hand gestures based on the localized key-points.

The mapping of the gestures and their corresponding action is given below:

Gesture	Action
ONE	Write
TWO	Erase
PEACE	Change writing color (Cyclic-Toggle)
FOUR	Defines writing-tracking zone threshold
FIVE	Save the board as pdf and jpg
FIST	Resting

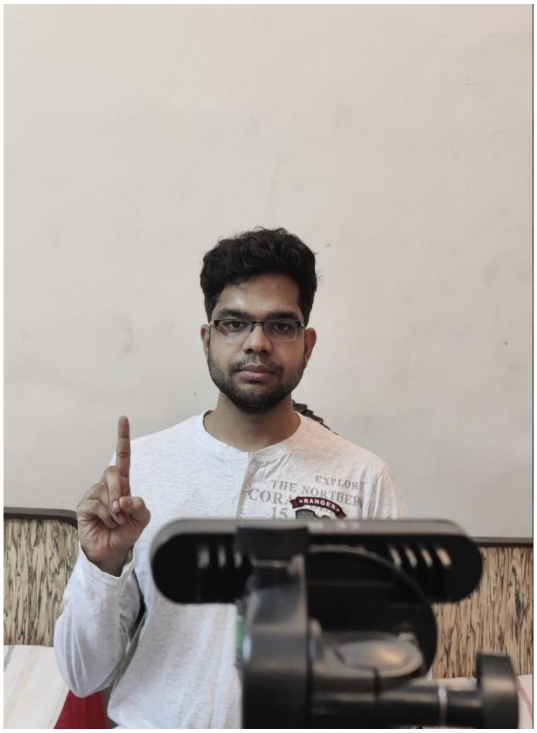


Figure 6(a). The figure represents the gesture “One” which performs the “Write” action.

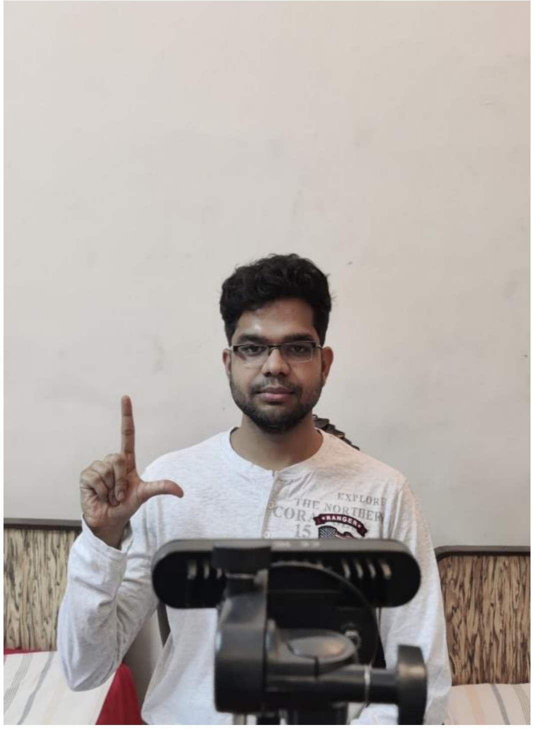


Figure 6(b). The figure represents the gesture “Two” which performs the “Erase” action.

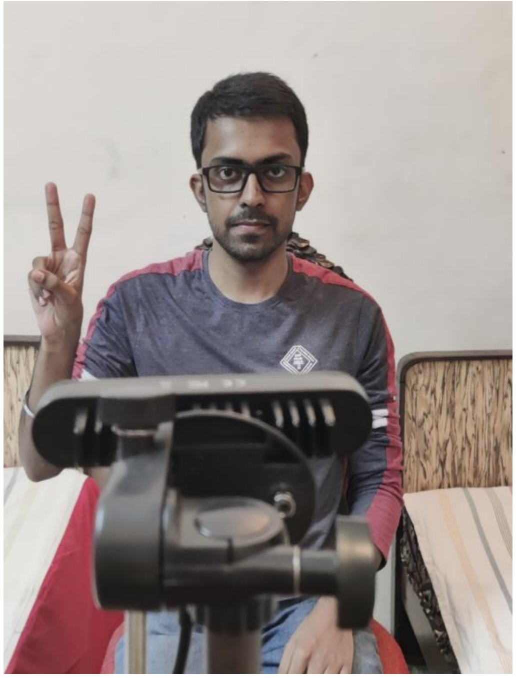


Figure 6(c). The figure represents the gesture "Peace" which performs the "Color Change" action.

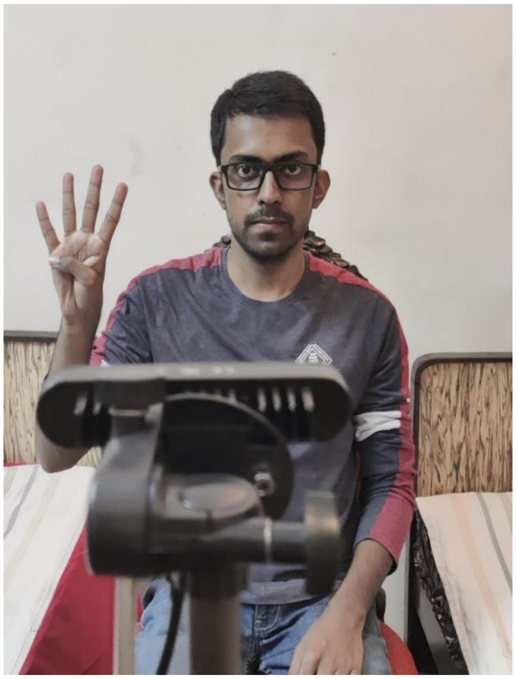


Figure 6(d). The figure represents the gesture "Four" which sets the "Depth Threshold".

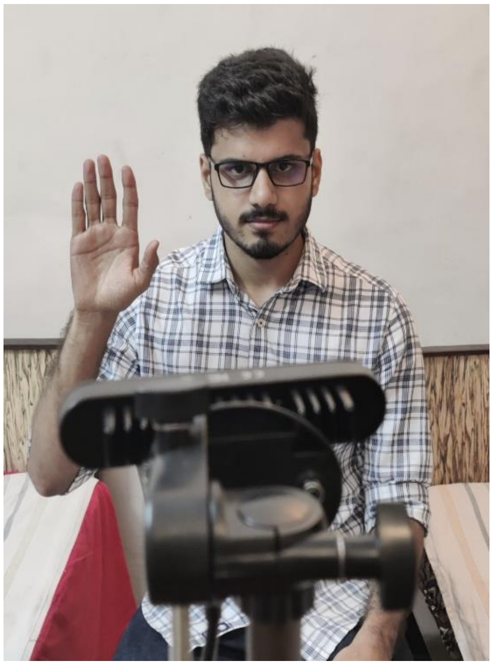


Figure 6(e). The figure represents the gesture "Five" which performs the "Save Board" action.

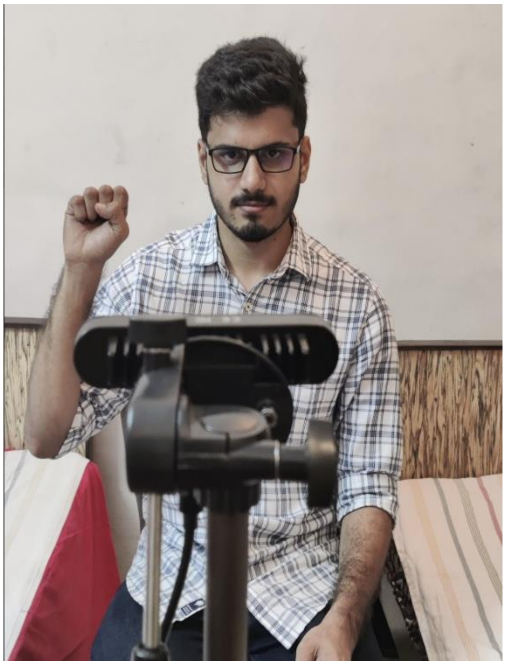
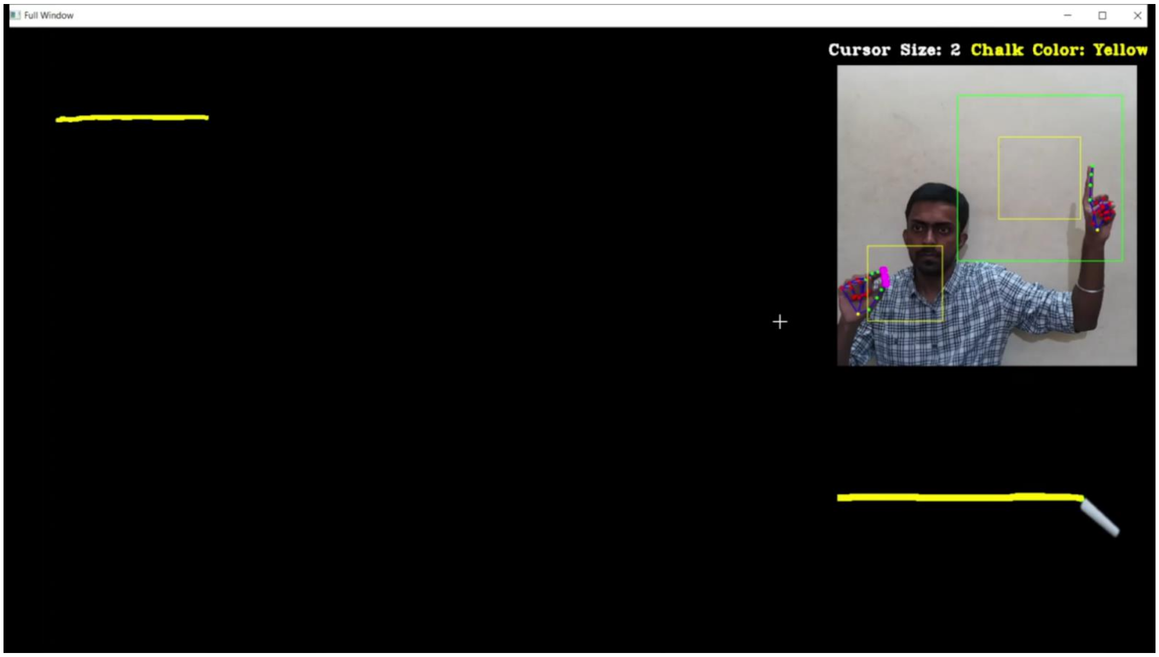


Figure 6(f). The figure represents the gesture "Fist" which indicates the "Resting" position.

An additional feature that we have added to enhance the user experience is the color changing cursor. We have divided the 3D space into 4 zones for the cursor. This helps the user to see where the cursor is with respect to the depth and enables the user to toggle between writing and tracking mode in a much more amiable fashion.

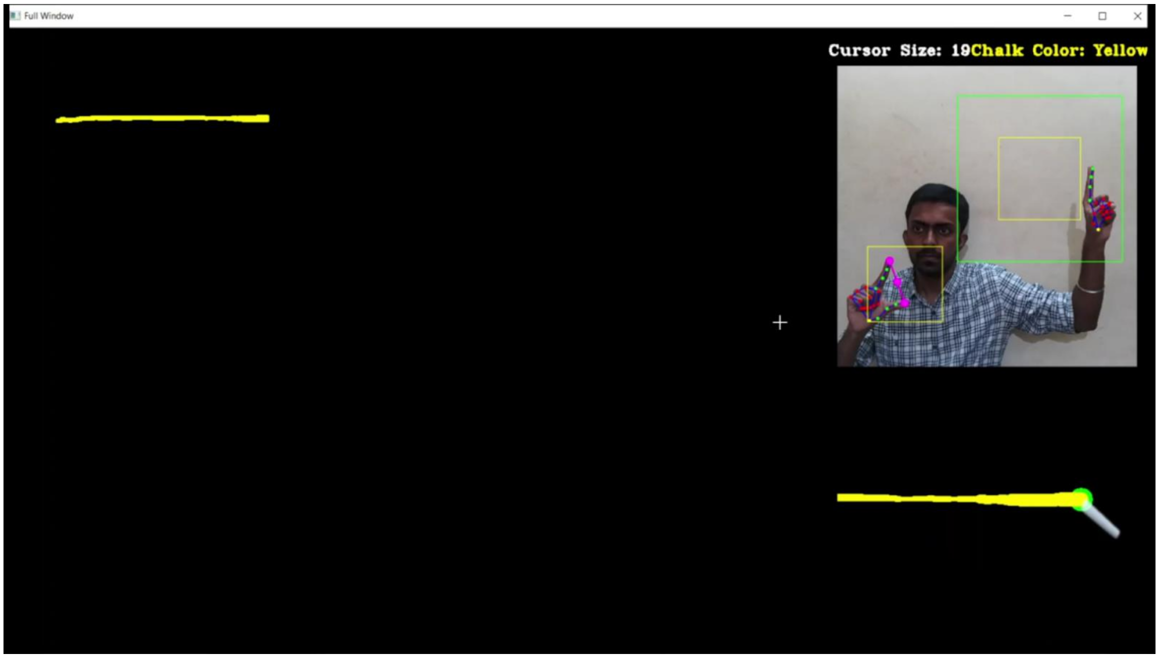
Zone	**Color**
Well-inside writing zone (Writing Mode)	Green
Close to threshold zone (Writing Mode)	White
Close to threshold zone (Tracking Mode)	Pink
Well-inside tracking zone (Tracking Mode)	Blue

We also have a dedicated region (ROI) which can be used to detect the non-dominant hand to modify the size of the cursor based on the Euclidean distance between the index finger and the thumb finger. The upper and lower bound of the cursor size is pre-set.



Cursor Size: 2 Chalk Color: Yellow

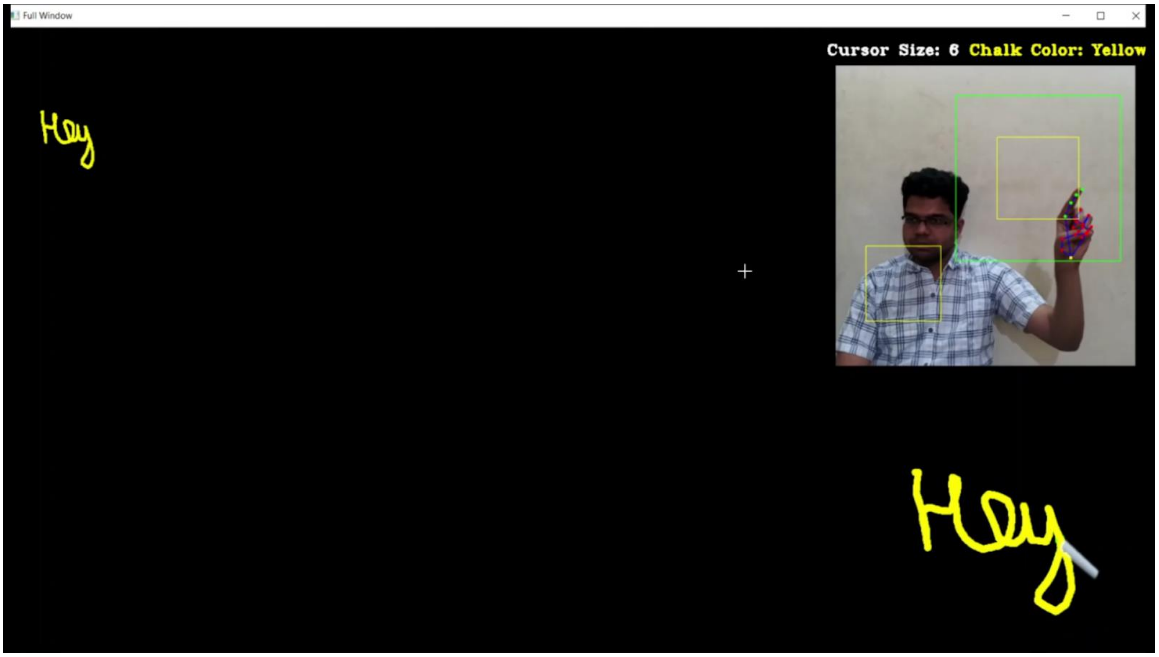
Figure 7(a). The region shown in the bottom-left corner of the user image detects the distance between the index and thumb finger. When the distance is small, the cursor size is small as shown above.



Cursor Size: 19Chalk Color: Yellow

Figure 7(b). When the distance is large, the cursor size is enlarged as shown above.

The entire structure of the user-interface is shown below. The two boxes on the right depict the writing area which comprises of the stationary area (yellow box) and the expandable area (space between the yellow and green box). There is also a separate region which is used to change the cursor size as described previously. The space below the user represents the content currently being written within the stationary area while the huge grid space on the left shows the entire information that has been transpired so far.



Cursor Size: 6 Chalk Color: Yellow

Figure 8. Structure of the user-interface

Limitations

- Good and stable lighting conditions are required for the system to perform efficiently.
- In order to achieve accurate tracking, the movement of the finger should be gradual and unrushed.
- The performance is currently capped at 15 FPS. This can be improved in the future with advancement in the hardware.
- Circular curves cannot be drawn in the expandable zone.
- The user has to practice with the product a few times to get the overall hang of the system.

Future Work

Our product works as a benchmark for the problem of air writing. There is a myriad of additional enhancements which we would like to add in the future to improve the user experience.

1. Sharing board across multiple teachers.
2. Incorporate 3D writing to enhance the learning experience.
3. Increase and decrease the speed of lazy move dynamically using history of motions of the user.
4. Additionally, Kalman Filters can be used to automatically control the movement.

Conclusion

This work is very close to our hearts as it opens up a new world of digital education which has the power to shape young minds of the future. Knowledge transfer of commendable quality can now take place between any two places of the world which are geographically distant. Better assimilation and dissemination of concepts is what we seek to achieve. Though there are limitations to our method (as discussed above), we believe that these can be overcome and remedied in the upcoming years.

We thank the OpenCV foundation for giving us this opportunity and the tools needed to innovate and bring our ideas to life. The Spatial AI Competition 2021, for us, has been a roller coaster ride with all its ups and downs amidst the pandemic. But in the end, we are happy that we made it! Now as the world gets back to normal, we're excited to put our product out in the world to enable Education in the best way we can.

References

1. MediaPipe
2. Depth AI Documentation
3. Hand-Tracking Module
4. Optical Flow