Development of a Learning-aid tool using Hand Gesture Based Human Computer Interaction System

Bidyut Jyoti Boruah
Dept. of Electronics and Communication Engineering
Gauhati University
Guwahati,Assam,India
bidyutjyoti15@gmail.com

Anjan Kumar Talukdar
Dept. of Electronics and Communication Engineering
Gauhati University
Guwahati,Assam,India
anjantalukdar@gauhati.ac.in

Kandarpa Kumar Sarma
Dept. of Electronics and Communication Engineering
Gauhati University
Guwahati,Assam,India
kandarpaks@gauhati.ac.in

***Abstract*—A hand gesture recognition system is a natural and simple way of communicating in today's world. The development of teaching methods by using technology-dependent useful items to increase communication and interaction between the teacher and the student is a major part of today's e-learning. In this paper, we have proposed an interactive learning-aid tool based on a vision-based hand gesture recognition system. The system uses MediaPipe for hand gesture recognition. The recognized hand gestures use a virtual-mouse-based object controlling system to control various virtual objects created using Unity. The system has been tested using six hand gestures and it is found that the system can be used effectively for controlling various virtual objects.**

***Index Terms*—Human Computer Interaction, Hand Gesture, MediaPipe, Unity**

I. Introduction

When using devices such as a mouse or keyboard for human-computer interaction (HCI), a natural interface between the user and the computer is inhibited. It is feasible to track a person's movements and determine what gestures they are making while talking using a variety of ways. Kinetic user interfaces (KUI) are a new type of user interface that lets users interact with computers by moving their bodies and things around. A touchless user interface (TUI) is a new type of technology for gesture control. Touchless user interface refers to controlling a computer with body motions and gestures rather than a keyboard, mouse, or screen (TUI). Touchless interfaces, as well as gesture controls, are gaining popularity because they allow users to interact with gadgets without touching them. MediaPipe Hand is a high-fidelity hand and finger tracking solution. It employs machine learning (ML) to infer 21 3D landmarks of a hand from just a single frame[1]. This project has the potential to reduce the workspace and the weight of additional hardware devices. It puts the user and the workspace closer together than before because it removes the encumbrance of technology.

978-1-6654-2337-3/21/31.00 © 2021 *IEEE*

Shajideen et al.[2] proposed a real-time human-computer interaction system (HCI) that uses two hand gestures: pointing and clenching fist. They tracked hand gestures in 2D space from a distance using a single camera hand gesture estimate technique, which was then transferred to screen coordinates. In addition, they also proposed the orthogonal camera to estimate hand gestures in 3D space from a distance and map to the coordinates of the screen. Prakash et al.[3] presented a finger tip recognition approach for HCI, namely mouse control activities employing a real-time camera. A real-time camera is used to capture hand motions. The hand region is first segmented using a region expanding method, then morphological procedures are performed. Thakur et al.[4]offered a method for regulating a computer mouse via 2D hand movements utilizing a webcam, with a focus on developing an HCI Virtual device in a value-effective manner. Haldera et al.[5]using MediaPipe's opensource framework and machine learning algorithm, presented a way for simplifying Sign Language recognition. The prediction model is simple to use and adaptable to different types of smart devices. To test the framework's capability, multiple sign language datasets such as American, Indian, Italian, and Turkish are employed. Lee et al. [6] presented a study in which a Kinect-based 3D gesture detection system for interactive manipulation of 3D objects in educational visualization software was developed and implemented. The proposed system recognizes human gestures by counting the number of open fists in each fist and tracking the 3D motion of both hands using RGBD images taken by a Kinect sensor. Its performance shows an overall average accuracy of roughly 90 % in recognizing the status of hands and gesture commands in a variety of illumination settings. Hu et al. [7] highlighted the tendency that the ideal relationship between a smart phone and a human is user-centered, making the mobile phone more than just a tool, but also a psychological companion. They proposed some guidelines based on the debate to assist designers in creating user-centered interaction designs, which will result in a user-centered, comforting, pleasant, and consumer-satisfied human-machine interface. The research proposed some prediction technologies that may be utilised to develop a user-centered system by combining traditional interaction design with the characteristics of interactive method. Shibly et al. [8] used computer vision and hand motions to develop a virtual mouse system based on HCI. Color segmentation and detection technology were used to process gestures collected with a built-in camera or webcam. Mhetar et al. [9] suggested a product that sought to be a virtual marker with the added benefit of having mouse-like features. In this paper, we have presented a virtual-mouse-based object controlling system using vision-based hand gesture recognition. A popular approach to communicate with computers without a mouse device is to use fingertip tracking as a virtual mouse. This fingertip-gesture interface enables humans to interact with computers by hand. The paper is organized as follows: Section II describes the proposed system, Section III includes experimental results and discussions and Section IV depicts the conclusion.

II. Proposed System

The proposed system is shown in Fig. 1.

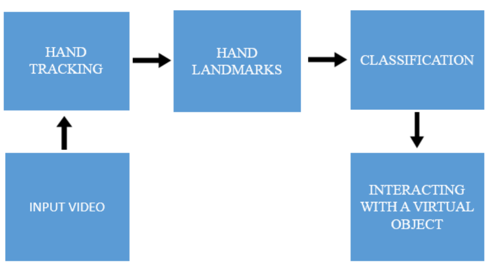

Fig. 1. Block diagram of the proposed system

A. Hand tracking by mediapipe

Hand tracking steps are:

1) Realtime Hand Detection: First, instead of training a hand detector, we train a palm detector because estimating bounding boxes of inflexible objects like palms and fists is much easier than recognizing hands with articulated fingers. Furthermore, because palms are smaller objects, the non-maximum suppression method performs effectively even in two-hand self-occlusion situations such as handshakes [1].
2) Hand Landmark: Following palm detection over the entire image, our next hand landmark model uses regression to accomplish exact keypoint localization of 21 3D hand-knuckle coordinates within the detected hand regions, i.e. direct coordinate prediction. Even with partially visible hands and self-occlusion, the model develops a consistent internal hand posture representation [1].

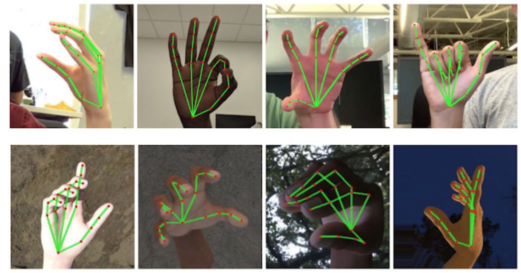

Fig. 2. Hand Landmarks [1]

3) Gesture Classification: We generate the gestures from the projected hand skeleton. To begin, the cumulative angles of joints determine the condition of each finger, such as whether it is bent or straight. We divided gestures into six categories: zero, one, two, three, four, and five as shown in Fig.3. This technique enables us to estimate gestures with a reasonable level of accuracy [1].

B. Mediapipe

Google's Mediapipe is a cross-platform library that delivers great ready-to-use machine learning solutions for computer vision workloads. MediaPipe is a structure that allows developers to create cross-platform multi-modal (video, audio, and any time series data) applied machine learning pipelines. Human body detection and tracking models from MediaPipe have been trained using Google's enormous and diverse dataset. They track critical spots on different regions of the body as a skeleton of nodes, edges, and landmarks. Three-dimension normalization is applied to all coordinate points. The flow of information is easily adaptable and adjustable via graphs. The training and classification of hand gestures is done using MediaPipe.

Fig. 3. Six hand gestures

C. Human Computer Interaction

The block diagram of the human computer interaction (HCI) is shown in Fig.4. The steps for HCI are discussed below:

1) Hand Tracking Module: Here, we start by creating a python file and defining a class called Handetector, which has four member functions: find hands, find positions, fingers up, and find the distance. The function find hands will accept an RGB image and detect our hand in the frame, locate the key points, and draw the landmarks of our hand, the function find positions will give the hand's position, the function fingers up will tell us which fingers of our hand are visible, and the function find distance will tell us the distance between two fingers, which will help us assign a threshold in the main function. We also construct a while loop to run the model and define the main method to initialize our module.
2) Finding hand landmarks: For locating hand landmarks, we employ the Mediapipe palm detection model, which provides precise key point localization of 3D palm coordinates in the detected hand region.
3) Checking which fingers are up: Following hand landmarks, we determine which of our five fingers is up. The result is displayed as a matrix; for example, [1,0,0,0,0] indicates that one finger is up; [1,1,1,1,1] indicates that all five fingers are up.
4) Index finger moving: We use our index finger to operate the cursor, but any other finger will suffice.
5) Find the distance between fingers: Now we need to calculate the distance between our index and middle fingers. The distance is determined since it will aid in the assignment of a threshold.
6) Cursor control: We assign a threshold to the distance value in output by determining the distance between the index and middle fingers in the previous step. We tested the model for different threshold values like 20, 25, 28 but the mouse cursor control lagged then at 30 we got an appropriate result and we could properly control our cursor as shown in fig 7. By setting a threshold of 30, we indicate that if the distance between two fingers is 30 or less, it's a click; if it's more, it's not. Click here refers to the control of our mouse pointer.
7) Interacting with a virtual object: In Unity software, we interact with a virtual object, such as a globe, after cursor control.

D. Unity3D

Unity is the most popular platform for developing and managing real-time 3D content in the world. Unity is used by a variety of creators, including game developers, painters, architects, automobile designers, filmmakers, and others, to bring their visions to reality. Unity's platform includes a game engine as well as a full collection of software tools for creating, running, and monetizing interactive, real-time 2D and 3D content for mobile phones, tablets, PCs, consoles, and augmented and virtual reality devices. Unity is an integrated development environment (IDE), which is a user interface that allows you to access all of the programming tools you need in one spot. The 3D object that we have chosen is a globe, which can be used by a teacher to simply explain to students the 3D structure of our globe as well as different regions inside it. Other virtual objects would also be easier for the students to comprehend [10].

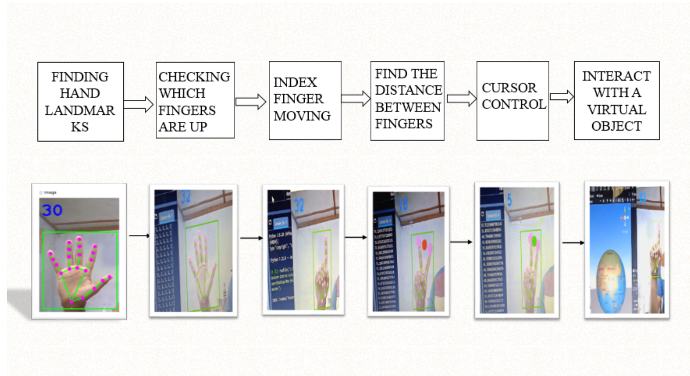

Fig. 4. Block diagram of HCI

E. Steps for creating a virtual object in Unity:

Unity3D is a powerful cross-platform 3D engine with a user-friendly programming environment. The first step after installing the software is to launch it and start a new project. Then we select an object from the Unity game object that we want to use. Next, we assign coordinates to the virtual object, then we import the asset and paste it on the 3D object. At last, we save the model. Fig. 6 shows a virtual 3D Globe created in Unity.

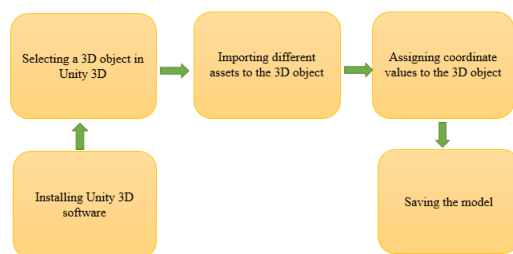

Fig. 5. Steps for creating a virtual object in Unity

III. Experimental Results

In Fig 7 we control the mouse cursor, without the need for any electronic equipment. The hand is detected by the Mediapipe library, which then draws the landmarks on the hand. For each hand, a total of twenty-one landmarks are drawn. The library can detect, track, and draw landmarks on several hands, but we altered the setup to be able to detect at most one hand among numerous hands (if any) in the frames to get the mouse cursor to work properly. The position of land

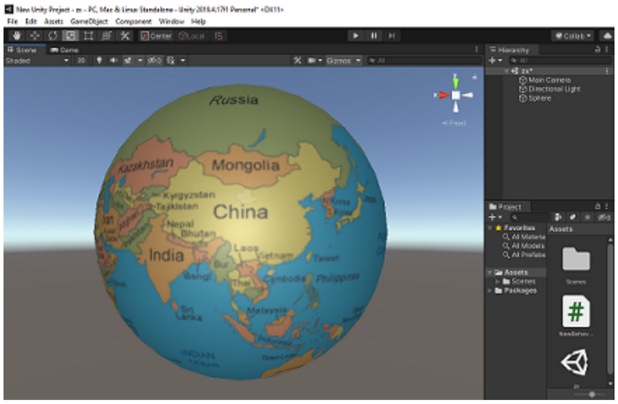

Fig. 6. A 3D globe in unity

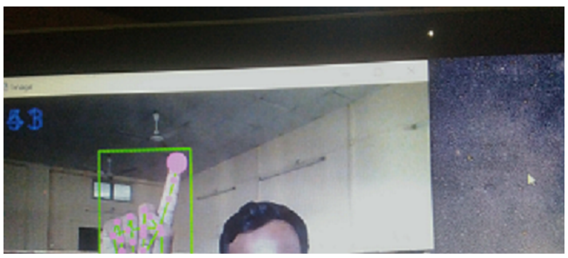

Fig. 7. Cursor control

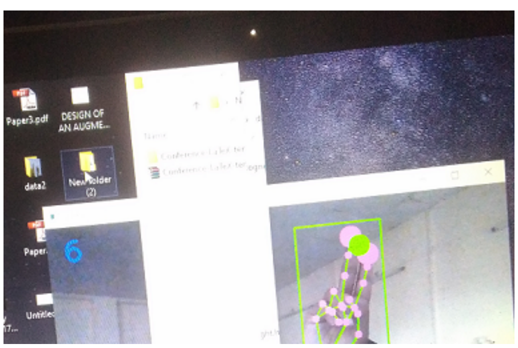

Fig. 8. Clicking to open a file

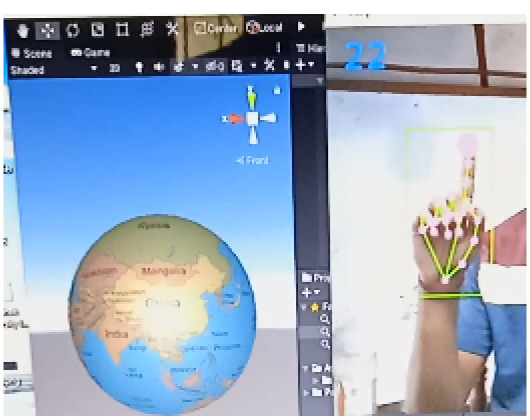

Fig. 9. Interacting with the virtual object

markings can be determined when the landmarks are sketched on the hand. The location of the landmark at the tip of the finger is then determined. The mouse cursor will be pointing to this location. Also painted across the hand is a rectangle boundary box. The bounding box's coordinates are calculated using the coordinates of the identified hand's landmarks. In Fig 8 we use our index finger and middle finger, the index finger is used to control the mouse. For the opening, closing of any folder, file, etc we use our middle finger for clicking purposes. In Fig. 9, we interact with a virtual object created in Unity3D. First, we use mediapipe to control the mouse cursor, and then we construct a 3D globe in Unity3D. Using the mouse cursor, we can now control the virtual object and execute various activities on it, such as rotation, movement, and so on.

IV. Conclusion

In this paper, we provide a study on human-computer interaction (HCI) that uses hand gestures to manipulate virtual objects. Unity3D is used to create virtual objects and MediaPipe for gesture recognition. The proposed HCI system can be used as a learning-aid tool that will be beneficial to both instructors and students. Teachers can create various 3D virtual items using Unity and control them using the proposed HCI system for improving the learning efficiency.

References

[1] F. Zhang, V. Bazarevsky, A. Vakunov, A. Tkachenka, G. Sung, C.-L. Chang, and M. Grundmann, "Mediapipe hands: On-device real-time hand tracking," *arXiv preprint arXiv:2006.10214*, 2020.

[2] S. M. S. Shajideen and V. Preetha, "Human-computer interaction system using 2d and 3d hand gestures," in *2018 International Conference on Emerging Trends and Innovations in Engineering and Technological Research (ICETIETR)*. IEEE, 2018, pp. 1–4.

[3] R. M. Prakash, T. Deepa, T. Gunasundari, and N. Kasthuri, "Gesture recognition and finger tip detection for human computer interaction," in *2017 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS)*. IEEE, 2017, pp. 1–4.

[4] S. Thakur, R. Mehra, and B. Prakash, "Vision based computer mouse control using hand gestures," in *2015 International Conference on Soft Computing Techniques and Implementations (ICSCTI)*. IEEE, 2015, pp. 85–89.

[5] A. Halder and A. Tayade, "Real-time vernacular sign language recognition using mediapipe and machine learning," *Journal homepage: www. ijrpr. com ISSN*, vol. 2582, p. 7421, 2021.

[6] J. Lee, H. Gu, H. Kim, J. Kim, H. Kim, and H. Kim, "Interactive manipulation of 3d objects using kinect for visualization tools in education," in *2013 13th International Conference on Control, Automation and Systems (ICCAS 2013)*. IEEE, 2013, pp. 1220–1222.

[7] M. Hu, P. Bian, and N. Zhang, "The research of human-machine interactions unity on ios and android smartphone platform," in *2014 9th International Conference on Computer Science & Education*. IEEE, 2014, pp. 640–643.

[8] K. H. Shibly, S. K. Dey, M. A. Islam, and S. I. Showrav, "Design and development of hand gesture based virtual mouse," in *2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT)*. IEEE, 2019, pp. 1–5.

[9] A. Mhetar, B. Sriroop, A. Kavya, R. Nayak, R. Javali, and K. Suma, "Virtual mouse," in *International Conference on Circuits, Communication, Control and Computing*. IEEE, 2014, pp. 69–72.

[10] G. Yazdani, F. Qazi, and S. A. Khan, "Adoption of vr influencing ai on 3d objects," in *2019 International Conference on Information Science and Communication Technology (ICISCT)*. IEEE, 2019, pp. 1–5.

[11] A. Banerjee, A. Ghosh, K. Bharadwaj, and H. Saikia, "Mouse control using a web camera based on colour detection," *arXiv preprint arXiv:1403.4722*

[12] C. Wan, T. Probst, L. V. Gool, and A. Yao, "Self-supervised 3d hand pose estimation through training by fitting," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 10 853–10 862.

[13] P. Kumar, J. Verma, and S. Prasad, "Hand data glove: a wearable real-time device for human-computer interaction," *International Journal of Advanced Science and Technology*, vol. 43, 2012.

[14] F. Karray, M. Alemzadeh, J. Abou Saleh, and M. N. Arab, "Human-computer interaction: Overview on state of the art," *International journal on smart sensing and intelligent systems*, vol. 1, no. 1, 2017.