Conjure: A Computer Vision Controller Using Hand Gestures

Anthony Hunt

April 19, 2025

1 Introduction

Over the last 20 years, CNNs and computer vision AI have become increasingly critical to the role of automation in both an industrial and personal setting. From self-driving cars to smartphone face recognition, Human-Computer Interaction is practically inseparable from the notion of gestures, or rather, interactions within the space around a computer system. Therefore, this project, named Conjure, aims to further enable visually interactive automation by fully enabling computer control in a hands-free environment.

The main interactions of Conjure are as follows:

- Hand movement in the camera's 2D projection of 3D space coincides with mouse movement across a computer's monitor. For fine-grained movements and other interactions, Conjure creates a deadzone at the center of the screen, where all hand movements are ignored. Then, as a hand moves out of the deadzone toward the borders of the screen, the mouse will move with increasing velocity in the hand's general direction.
- Simulating a left click can be done by making an "OK" gesture towards the camera. This gesture is closest to the natural pinch gestures of VR headsets while being freely available in many datasets [1, 3, 6, 2].
- A right click conversely uses a "peace" sign, with the index and middle fingers extended and pointing away from each other.
- A click-and-hold motion makes use of a closing fist gesture. As long as the fist remains closed, the mouse will continue holding down the button. Moving around the camera space while maintaining a closed fist allows for drag-and-drop behaviours.
- Exiting the program is done with a "reverse-stop" motion, where the hand is fully extended, fingers close to one another, and the user's palm is facing away from the camera.

- Scrolling coincides with a two-finger-raised gesture, with up and down behaviour following the palm's direction - up when facing towards and down when facing away from the camera respectively. During testing, I found that scrolling behaviours worked best if it was only used in a specific area of the screen. Thus, this gesture only works within the scroll zone (by default, this is the right side of the mouse deadzone).

All of these gestures may be changed with the GUI configuration, made in Tkinter. Movement sensitivity, deadzone sizes, etc., are all configurable through the UI.

2 Model

3 Results

4 Conclusion

References

[1] Kapitanov Alexander, Kvanchiani Karina, Nagaev Alexander, Kraynov Roman, and Makhliarchuk Andrei. "HaGRID - HAnd Gesture Recognition Image Dataset". In: *2024 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. IEEE, Jan. 2024. DOI: 10.1109/wacv57701.2024.00451.

[2] *Gesture recognition task guide*. 2025. URL: https://ai.google.dev/edge/mediapipe/solutions/vision/gesture_recognizer (visited on 04/19/2025).

[3] Anton Nuzhdin, Alexander Nagaev, Alexander Sautin, Alexander Kapitanov, and Karina Kvanchiani. *HaGRIDv2: 1M Images for Static and Dynamic Hand Gesture Recognition*. 2024. arXiv: 2412.01508 [cs.CV]. URL: https://arxiv.org/abs/2412.01508.

[4] *OpenCV*. 2025. URL: https://opencv.org/ (visited on 04/19/2025).

[5] *PyAutoGUI*. 2019. URL: https://pyautogui.readthedocs.io/en/latest/ (visited on 04/19/2025).

[6] Christian Zimmermann and Thomas Brox. *Learning to Estimate 3D Hand Pose from Single RGB Images*. 2017. arXiv: 1705.01389 [cs.CV]. URL: https://arxiv.org/abs/1705.01389.