Conjure: A Computer Vision Controller Using Hand Gestures

Anthony Hunt

April 20, 2025

1 Introduction

Over the last 20 years, extraordinary progress in image processing and CNNs have brought about revolutionary methods of interacting with the world. From self-driving cars to early cancer detection, computer vision has become integral to interactions with technology and our environments. The usefulness of such software is self-evident in its pervasiveness within smartphones and personal computing; home buttons have been replaced with retina scanners and facial recognition, VR headsets no longer need dedicated controllers, and any device can read handwritten text directly from a photo.

In the gradual rollout of virtual hands-free computing, gesture-based interactions serve as a natural and intuitive platform for communicating with computers. This project, named Conjure, aims to take gesture-based interactions one step further by providing human-computer interactions with only a camera and hand recognition software. However, unlike the Xbox Kinect or Apple Vision Pro headset, which make use of depth sensors in addition to regular cameras, we attempt to use only a standard camera. Of course, the use case of Conjure is far simpler than that of VR headsets or game consoles, since the target platform this program is user-facing laptop webcams. We can further assume that most people interact with webcams in an egocentric view, that is, facing the camera with one prominent subject and a mostly static background.

The rest of this report will outline the features of Conjure along with instructions to get started, the model architectures explored and used within the program, some key results, and difficulties throughout the project.

2 Usage

To get started, clone the GitHub repository and download all dependencies through `pip install -r requirements.txt`. Ensure the computer's webcam is plugged in and working, then run `python main.py`. Note that this project was tested with Python 3.12 on a Windows machine.

Upon startup, a GUI containing settings and other options should appear similar to Figure 1

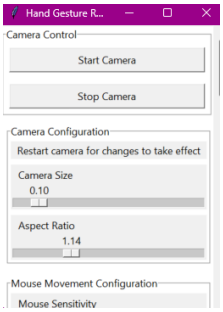


Figure 1: Startup configuration UI. Click Start/Stop Camera to activate the virtual controller.

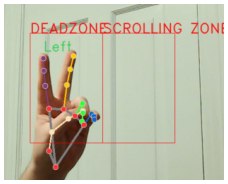
The main interactions of Conjure are as follows, with coinciding images in Figure 2:

- 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, 5, 10, 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).

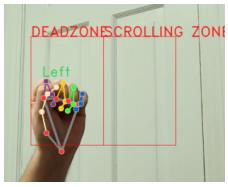
All gesture-specific actions may be changed with the GUI configuration, made in Tkinter. Other options, like movement sensitivity, deadzone sizes, etc., are also configurable through the UI.

(a) The left deadzone prevents hand movement within that area from controlling the mouse. Any gestures will work in this area. The right scroll zone prevents mouse movement but enables scrolling movement. Pointing two fingers up when the index finger is within this zone will perform a scroll-up action.

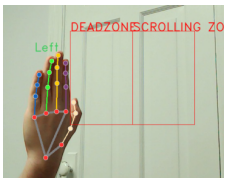
(b) OK symbol for left click.



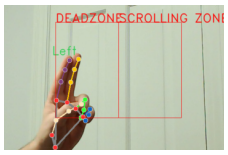
(c) Peace symbol for right click.



(d) Clenching a hand will mirror click-and-hold behaviour.



(e) A stop symbol with the palm facing away from the camera will exit the camera control program.



(f) Two fingers pointing up will scroll up (only in the right side scroll zone).

(g) Showing the back of the scroll-up gesture will scroll down (only in the right side scroll zone).

Figure 2: A collection of gestures described in section 2.

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] *Hands parent: MediaPipe Legacy Solutions*. 2025. URL: https://mediapipe.readthedocs.io/en/latest/solutions/hands.html (visited on 04/19/2025).

[4] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. "SSD: Single Shot MultiBox Detector". In: *Computer Vision – ECCV 2016*. Springer International Publishing, 2016, pp. 21–37. ISBN: 9783319464480. DOI: 10.1007/978-3-319-46448-0_2. URL: http://dx.doi.org/10.1007/978-3-319-46448-0_2.

[5] 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.

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

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

[8] Usman Rizwan. *Implementing a Single Shot Detector Model in TensorFlow 2.0*. 2025. URL: https://usmanr149.github.io/urmlblog/computer%20vision/2022/09/10/Implenting-SSD-TF2.html (visited on 04/19/2025).

[9] Dibia Victor. "HandTrack: A Library For Prototyping Real-time Hand TrackingInterfaces using Convolutional Neural Networks". In: *GitHub repository* (2017). URL: https://github.com/victordibia/handtracking/tree/master/docs/handtrack.pdf.

[10] 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.