On-Screen Navigation with Hand Gestures using Computer Vision

Yashas H Majmudar

Student of Bachelor of Technology CSE,

Indus University Ahmedabad, India

yashashm@gmail.com

ABSTRACT – *In the budding and upcoming field of Artificial Intelligence computer vision is a field that enables computer systems to retrieve useful information from a digital source and perform actions based on it. Hand Gesture recognition is an interactive way to interact with a device. Hand Gesture Recognition is the process of retrieving meaningful data from set of images. This paper describes the research and development of a computer vision-based Hand Gesture - Screen Navigation system that interprets moving hand gestures to switch/scroll between screens of a device. It can be further be implemented as a gesture control system as well as a novel approach for Human Computer Interaction (HCI) to eliminate the need for input devices like touchpad or mouse.*

KEYWORDS – *Computer Vision, Hand Gesture Recognition, Hand Movement, Human Computer Interface*

IJSER

I. Introduction

In MARVEL's Iron Man Movie, the protagonist uses his hands to controls the holograms placed around him. He only uses his hands to control and manipulate audio, video, 3D models and graphics in his home [1]. The scene represented such futuristic technologies which expanded the horizons for Human Computer Interaction.

Gestures [2] are a form of nonverbal communication in which visible bodily actions are used to communicate important messages, either in place of speech or together and in parallel with spoken words. Gestures include movement of the hands, face, or other parts of the body. Hand motion and gesture is a powerful means of communication between humans, we even use gestures subconsciously while talking on the telephone. These kinds of human computer interfaces exist, which aid us to manipulate virtual objects and screens utilizing sensors and buttons, but these devices are either too expensive or require special devices to run on. A computer vision-based system for manipulation of virtual data is much cheaper and is compatible with most of the devices that majority of the population holds.

The proposed HCI is based on using Computer Vision (CV)[3] which is a subfield of artificial intelligent system, which deals with images and the understanding of images to extract specific data by processing the images. CV uses the information from the images, perceives and processed the data to simulate human vision. Using a Computer Vision approach rather than electronic sensors enables us to integrate gestures and natural movement control to already existing devices without need of unnecessary and expensive mechanical and electronic equipment.

II. Related Work

Some uses of Computer vision and gesture detection are:

ASL (American Sign Language) Sign Alphabet Detection [4]

Static images are sensed by the camera and alphabet is predicted.

ASL (American Sign Language) Sign Gesture Detection [5]

Active Gestures are detected and a word is predicted.

III. Approach and Methods

Diagram 1 shows that the approach for this system is divided into three segments:

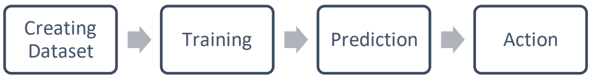


diagram 1. Flow of the Approach

III.I Creating Dataset

The dataset is created using Key Point Values of the gestures in frame. Key Points are certain points on the object that are sensed by the computer, for example, 2D position of an object, orientation of detected object etc.

In this project I have used MediaPipe[6] Library for detection of Hand, Face and Pose landmarks. The library detects the object from Video or Image and return Key Point Values. These Key Point values determines a 3D position of the detected object. Key Point Values are X, Y and Z coordinates (In case of pose, there is an additional Visibility attribute of the Key Points.), where, X & Y represents the position of the object detected in a 2D plane (X-Y Axis) and Z represents the distance of the object from the camera. Combined together these values form a virtual 3D position of the detected object. In our case the detected object is Hands (Left & Right), Face and Pose (Body). Diagram 2 shows the detection of hands, face and pose using the MediaPipe Library.

IJSER

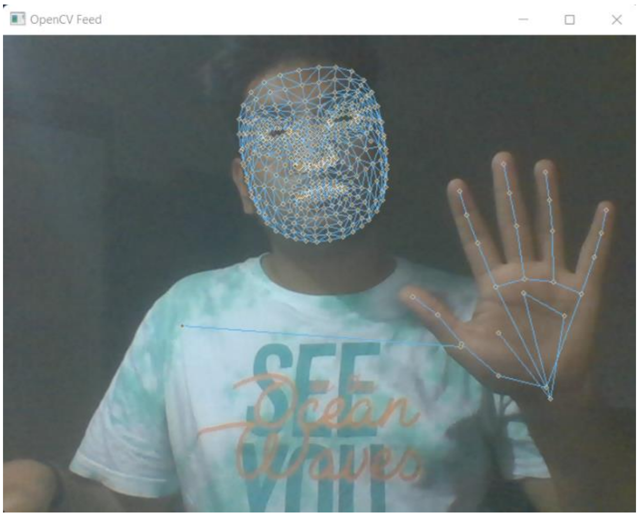


diagram 2. Hands, Face and Pose Key Points.

These collected points are then converted to a NumPy Array[7] and all points are then combined to form a single Key Point Array which is then saved as an .npy file (NumPy Array). In case, no Hands, Face or Body are detected the array is filled with zeroes to maintain the shape of the data.

In this system, a threshold of 30 Frames is set per action (e.g., Hand motion Left to Right is comprised of only 30 frames). Each frame is taken as an image and Key Point Values are extracted and formed NumPy array from the Key Points is saved in a file. For each gesture 30 videos of 30 frames are taken to form the dataset for the respective action. In total 900 files each of 1662 values are created for the dataset for a single action.

III.II Training the model using Dataset

In this system, a Sequential model is used with Six layers which is created using the CNN Architecture (Convolution Neural Network).

First three layers are LSTM Layers. LSTM[8] (Long Short-Term Memory) is an artificial Recurrent Neural Network (RNN) architecture. It can process not only single data points (images), but also entire sequences of data (speech or video). LSTM is used as it is well suited for making predictions based on time series data, since there can be lags between import events in a time series. First LSTM layer is added to the sequential model with specified input shape (30, 1662), 64 number of units to be taken as input and return sequence True. Return Sequence attribute enables the layer to return values after passing through the LSTM layer. The second LSTM layer is added to the sequential model with 128 number of units to be taken as input and return sequence as True. Finally, the third layer is added with 64 number of units to be taken as input and return sequence as False. All three layers are set with activation as 'relu' (rectified linear unit activation), A Tensor representing the input tensor, transformed by the 'relu' activation function, the Tensor will be of the same shape and dtype of input x.

Last three layers are Dense layers, Dense[9] implements the operation: output = activation(dot(input, kernel)) where activation is the element-wise activation function passed as the activation argument, kernel is a weights matrix created by the layer. Fourth layer is added to the sequential model with activation 'relu' and input unit size as 64. Fifth layer is added to the sequential model with activation 'relu' and input unit size as 32. The last layer is added to the sequential model with unit size as the number of actions in the dataset and activation 'softmax'. 'Softmax' converts a vector of values to a probability distribution. The elements of the output vector are in range (0, 1) and sum to 1. Each vector is handled independently. 'Softmax' is often used as the activation for the last layer of a classification network because the result could be interpreted as a probability distribution. The 'softmax' of each vector x is computed as exp(x) / tf.reduce_sum(exp(x)).

The model is compiled using 'adam' optimizer. Adam optimization is a stochastic gradient descent method that is based on adaptive estimation of first-order and second-order moments. The model set to return and represent epoch loss as categorical crossentropy, computes the crossentropy loss between the labels and predictions. This crossentropy loss function is used when there are two or more label classes. The model accuracy is calculated by categorical accuracy function. Calculates how often predictions match one-hot labels. You can provide logits of classes as y_pred, since argmax of logits and probabilities are same.

Model is fit with the training data in batches of 2000 epochs and callback is set as tb_callback to analyse epoch loss and categorical accuracy of the model while data is being trained. Diagram 3 shows graph for categorical accuracy & diagram 4 shows graph for epoch loss while training data.

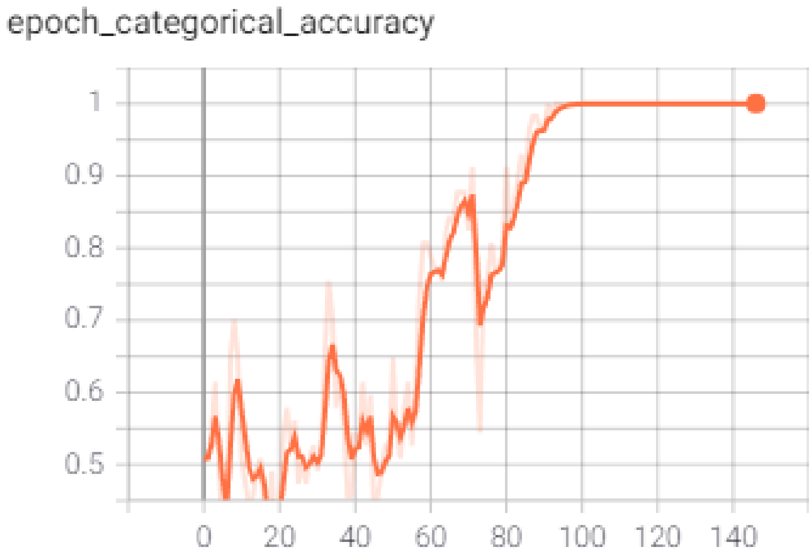
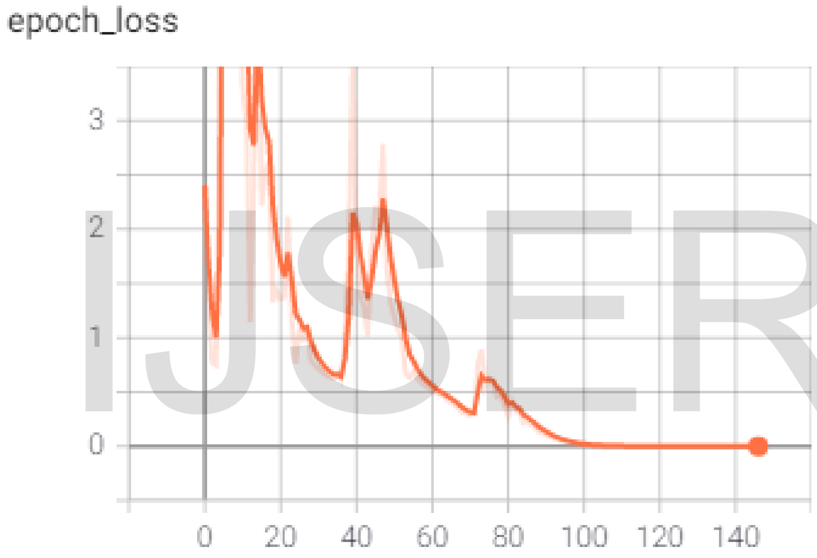


diagram 3. Epoch Categorical Accuracy while training Dataset



IJSER

diagram 4. Epoch Loss while training Dataset

III.III Prediction

Predictions are made using the Model and Dataset based on live gestures from the computer. Video input is taken from the camera; 30 frames are taken in consideration, in which hands, pose and face landmarks are extracted and sent to the model for prediction. The collected Key Point Values are processed by the model to give out a prediction matrix. On the matrix a function numpy.argmax() is used to find the maximum occurring element in the matrix and returns integer value. This integer value is passed as index to the array that contains the labels of actions. The returned value is the action that was passed as input. Accuracy for this model was calculated using the accuracy_score() function in scikit learn library and a confusion matrix is also prepared using the multilabel_confusion_matrix() function from the same library. The function returns an array of matrices which are then represented through a heatmap. Diagram 5 shows the heatmap of the accuracy of the model.

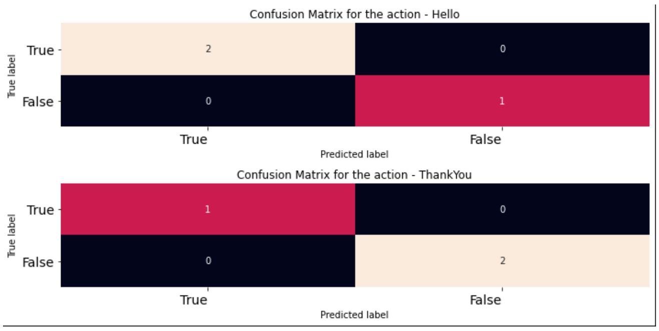


diagram 5. Confusion Matrix of Predictions

III.IV Action

Once the system detects and predicts the gesture a text is returned. This text is passed to a switch function which executes pre-defined function based of the key that is passed. For example, if the predicted gesture is hand motion left-to-right then screen change function is called.

IV. Results

Change in screen is successful using hand motion. This also provides means for innovation in Human Computer Interface development using Computer Vision and Machine Learning Technologies.

IJSER

V. Conclusion

I have shown the research of a simple computer vision-based system that detects whole action and predicts to give an output successfully. Using this technique, the use of heavy sensors and electronics can be reduced, in-turn creating a novel approach for Human Computer Interface.

VI. References

[1] Iron Man (2020) #17, February 23, 2022

[2] Gestures: Their Origins and Distribution by Desmond Morris, Peter Marsh, Peter Collett, Marie O'Shaughnessy

[3] Computer Vision: A Modern Approach. (Second edition) by David Forsyth and Jean Ponce, January 21, 2022.

[4] American Sign Language Alphabet Recognition using Deep Learning by Nikhil Kasukurthi, Brij Rokad, Shiv Bidani, Aju Dennisan

[5] C. Vogler and D. Metaxas, "Adapting hidden Markov models for ASL recognition by using three-dimensional computer vision methods," 1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation, 1997, pp. 156-161 vol.1, doi: 10.1109/ICSMC.1997.625741.

[6] MediaPipe Hands: On-device Real-time Hand Tracking Fan Zhang, Valentin Bazarevsky, Andrey Vakunov, Andrei Tkachenka, George Sung, Chuo-Ling Chang, Matthias Grundmann, 5 pages, 7 figures; CVPR Workshop on Computer Vision for Augmented and Virtual Reality, Seattle, WA, USA, 2020.

[7] Harris, C.R., Millman, K.J., van der Walt, S.J. et al. Array programming with NumPy. Nature 585, 357–362 (2020). https://doi.org/10.1038/s41586-020-2649-2

[8] Xuanyi Song, Yuetian Liu, Liang Xue, Jun Wang, Jingzhe Zhang, Junqiang Wang, Long Jiang, Ziyan Cheng, Time-series well performance prediction based on Long Short-Term Memory (LSTM) neural network model, Journal of Petroleum Science and Engineering, Volume 186, 2020, 106682, ISSN 0920-4105
.
[9] Gao Huang, Zhuang Liu, Laurens van der Maaten, Kilian Q. Weinberger; Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 4700-4708

IJSER