Gesture Controlled Power Window Using Deep Learning

Jatin Rane(✉) and Suhas Mohite

Department of Mechanical Engineering, College of Engineering Pune Technological University, Wellesley Road, Shivajinagar, Pune, Maharashtra 411005, India
{ranejw21.mech,mohitess.mech}@coep.ac.in

Abstract. Researchers are working to fill knowledge gaps as new eras are ushering in by the rapid growth of informatics and human-computer interaction. With speech-based communication, touch-free engagement with electronic gadgets is growing in popularity and offers consumers easy-to-use control mechanisms in areas other than the entertainment sector, including the inside of cars, these engagement modes are now being successfully used. In this study, real-time human gesture identification using computer vision is proven, and the possibility of hand gesture interaction in the automobile environment is investigated. With the use of this noncognitive computer user interface, actions can be carried out depending on movements that are detected. By adding Python modules to the system, the design is carried out on a Windows OS. The platforms used for identification are open-cv and keras. The vision-based algorithms recognize the gesture displayed on the screen. A recognition algorithm was trained in keras using the background removal technique and the LeNet architecture. In this paper, four models were created and their accuracy was compared. The convex hull and threshold model outperformed the other models.

Keywords: computer vision · open-cv · LeNet · keras · gesture recognition

1 Introduction

By evaluating the user's gestures, body postures, or voice and interpreting human intention in to the device control commands, natural user interfaces enable human-machine interaction. The intelligent assistant concept is the result of advances in intelligent systems, artificial intelligence, gesture and voice recognition, dialogue systems, data science, and natural language processing. Such solutions are becoming more common in new cars, and their purpose is to aid drivers in the usage of advanced driver assistance systems (ADAS), such as systems for detecting driver fatigue, blind spot detection, lane detection & departure warning etc.

Natural user interface (NUI) is based on simple and effortless interaction with electronic devices. The interface should be barely noticeable and composed of natural elements. Entire movements cannot be incorporated into the interior of the car, but gestures

S. Nandan Mohanty et al. (Eds.): ICISML 2022, LNICST 471, pp. 335–344, 2023.
https://doi.org/10.1007/978-3-031-35081-8_28

and voice are quite well suited for information exchange with a smart assistant or multimedia system. It should be noted that all these methods of communication may not always be appropriate. When a driver is playing the radio or driving in a noisy conditions, the voice-controlled interface will always fail. In many cases, both hands are used for manoeuvring (steering wheel handling and gear shifting), making additional gestures impossible, thus static gestures with one hand shows their usefulness here.

More than a billion individuals worldwide have some sort of disability, such as hand and speech abnormalities, according to the World Report on Disability from the World Health Organization and the World Bank. Also India has five million people with communication disabilities (Vikas Sheel, Additional Secretary and Mission Director, National Health Mission (NHM)). Most of them struggle significantly on a daily basis. In order to aid the daily activities of individuals with impairments, new types of technologies and equipment are being developed. Thanks to specialized technological solutions like smart systems or driving help gadgets, many of these people can operate automobiles on their own. However, a button console, like that of a power window, multimedia system, is inoperable to a disabled person (due to difficulty to access or due to disability). In this situation, NUI can show its value and usefulness here. In the next section, relevant literature is reviewed in brief.

2 Related Works

In an article, Graziano Fronteddu, Simone Porcu, Alessandro Floris, and Luigi Atzori [1] proposed a dataset of 27 dynamic hand gesture types obtained at the full HD resolution from 21 various subjects. The participants were carefully advised prior to actually performing the gestures and watched whereas performing the gesture; if the performed hand gesture seemed to be incorrect, the subjects were required to repeat the movement. Since the proposed dataset contains high-quality recordings of participants correctly performing 27 various hand gestures, it encourages researchers to develop accurate gesture recognition system systems. This proposed dataset is quite useful for training & testing purpose for multi gesture recognition system and have characteristics which are not present in publically available datasets.

Indian sign language (ISL) hand gestures are both static and dynamic in the time domain, claims Dushyant Kumar Singh [2]. Although there is an established standard for Indian sign language, few people actually use it. In this study, the author modelled the most widely used gestures in the Indian society using a convolutional-based convolution neural network. The author trained his model, which was produced by imitating the motions of those gestures, using 20 gestures from the conventional Indian sign language. The trained model is able to produce output in normal language that corresponds to ISL signs. Hence, the challenges associated with conversing with dumb and deaf persons will be lessened by this. These dynamic motions can also be applied in a variety of different disciplines, including the clinical and industrial ones.

In their research, Yassine Rabhi, Makrem Mrabet, and Farhat Fnaiech [3] proposed a human machine interface (HMI) that offers a useful hands-free option. The wheelchair may be controlled by the user by altering its face expressions. Based on the use of machine learning and particular picture preparation techniques, this clever solution. The Viola-Jones combo is first used to identify the face in a video. After that, the expressions seen on

the face are categorized using a neural network. The "The Mathematics Behind Emotion" approach can recognize a variety of facial emotions in real - time basis, such as smile and raised eyebrows, and then convert them into inputs for wheelchair operation. Authors also suggested that system can be easily mounted on wheelchair by using microcontrollers like raspberry pi and compatible camera.

To address the issue of a poor rate of gesture image identification, Fei Wang, Ronglin Hu, and Ying Jin [4] suggested a transfer learning-based image recognition system titled Mobilenet-RF. To further enhance the precision of image recognition, they mix Random Forest and the two MobileNet convolutional network models. This process applies the MobileNet model architecture and weight files to gesture photographs first, trains the model, extracts image features, classifies the features obtained by the convolutional network using the Random Forest model, and then yields classification results. When compared to Random Forest, Logistic Regression, K-Nearest Neighbor, and other approaches, the results of the testing are significantly improved.

The project's methodology was determined as a result of this literature review. The methodology for the project is explained in detail in the following section.

3 Proposed Method

We demonstrated how to incorporate real-time hand gesture recognition into advanced technology such as Computer Vision in the following technique. This project was created to identify two sets of gestures displayed. The open source computer vision Library, developed by Intel, was used to implement this strategy. For machine learning tasks and ensuring consistency in large matrix processing tasks, Keras and numpy libraries were used. The input is a live webcam to recognize and predict gestures.

The experiment was carried out in specific and flourishing lighting conditions. Before developing datasets from the captured frames, each frame is pre-processed under specific conditions.

The proposed methodology can be divided into several sections: image acquisition, dataset preparation, CNN model development, and arduino interfacing. The following block diagram depicts all of the steps taken in the proposed system. In the next section detailed methodology is explained step by step (see, Fig. 1).

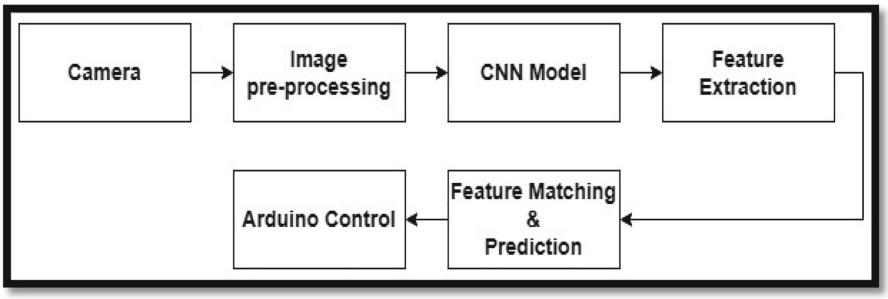

Fig. 1. Flow chart of system

4 Methodology

4.1 Image Acquisition

The first step was to capture the camera frame and identify the region of interest in the image. The challenge is to use an appropriate computer vision method capable of detecting even the smallest difference between similar signals in order to prevent inaccurate translation in real time. In open-cv, there are over 150 color-space conversion methods. However, thresholding and BGR to Gray, are mainly used here. In this paper, we convert the colored RGB format to BGR and then BGR to GRAY because it is more convenient to extract the gesture and easier to process than others.

To capture and store images, a python program was created. The Code was successfully updated for easy data collection, and the in-built python OS module was used to create appropriate folders to store captured images. Images were captured at the rate of 30 fps using laptop in built camera (0.3 megapixel). Following figure shows (see, Fig. 2.) the layout of python program.

```
    os.makedirs("data/test/increase")
    os.makedirs("data/test/decrease")

# train or test data

mode = input("type 'test' for TEST DATA :-\ntype 'train' for TRAIN DATA :- ")
directory = 'data/' + mode + '/'

cap = cv2.VideoCapture(0)

while True:
    _, frame = cap.read()
    # mirror image
    frame = cv2.flip(frame, 1)

    # count of existing images
    count = {'increase': len(os.listdir(directory + "/increase")),
             'decrease': len(os.listdir(directory + "/decrease"))
             }

    # Printing the count in each set to the screen
    cv2.putText(frame, "MODE : " + mode, (10, 50), cv2.FONT_HERSHEY_PLAIN, 1, (0, 255, 255), 1)
    cv2.putText(frame, "IMAGE COUNT", (10, 100), cv2.FONT_HERSHEY_PLAIN, 1, (0, 255, 255), 1)
    cv2.putText(frame, "increase : " + str(count['increase']), (10, 120), cv2.FONT_HERSHEY_PLAIN, 1, (0, 255, 255), 1)
    cv2.putText(frame, "decrease : " + str(count['decrease']), (10, 140), cv2.FONT_HERSHEY_PLAIN, 1, (0, 255, 255), 1)
    cv2.putText(frame, "press 'c' to change mode", (10, 160), cv2.FONT_HERSHEY_PLAIN, 1, (0, 255, 255), 1)

    # Coordinates of the ROI
    x1 = int(0.5 * frame.shape[1])
    y1 = 10
```

Fig. 2. Python program for data acquisition

4.2 Dataset Preparation

The proposed methodology also includes segmentation. In the simplified application, the process displays a binary object that describes the segmentation. Background pixels are black, while foreground pixels are white. In simple implementations, the intensity threshold is the only parameter that determines segmentation.

The main three datasets were created by taking into account conditions such as varying the threshold, using a full-scale colour image, recording various gestures, and employing an image contouring method (convex hull). In addition, an online dataset from kaggle.com was used for training. All four datasets in this project contain 2000 images each for the two gestures increase and decrease. Increase to raise the window and decrease to lower it (Fig. 3).

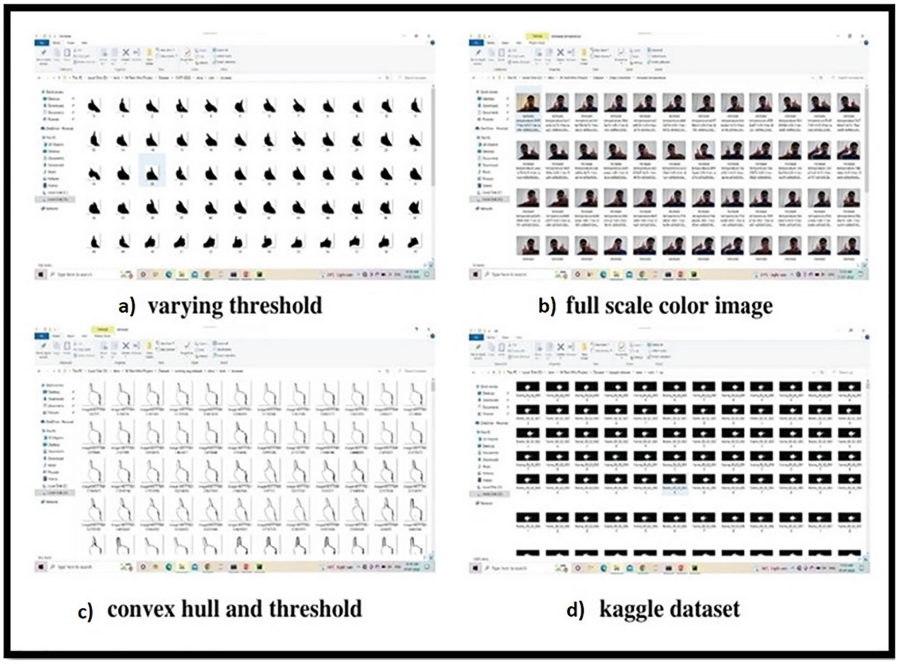

Fig. 3. Various datasets of gestures processed using different techniques

In the next section CNN model development process used in this work is described in brief.

4.3 CNN Model Development

CNN was chosen for its weight sharing feature, which reduces the number of trainable network parameters, allowing the network to improve generalization and avoid over fitting. It is also capable of handling spatial data (images) and detecting distinct features from images without any actual human intervention.

A model was developed using jupyter lab. All programming was done using python and various python modules were used such as keras models, keras preprocessing, open-cv, numpy.

The CNN network architecture has layout as enumerated below:

a) First convolutional layer with filter size = 32, kernel (3, 3), and activation function relu is used with maxpool size (2, 2).

b) Second convolutional layer with filter size = 64, kernel (3, 3), and activation function relu is used with maxpool size (2, 2).
c) Third convolutional layer with filter size = 64, kernel (3, 3), and activation function relu is used with maxpool size (2, 2).
d) Then flatten function to 1-D array.
e) Input neuron layer with 128 neurons activation function relu is used.
f) Then dropout layer with 50% dropping rate.
g) Output neuron layer with 1 neurons activation function sigmoid is used.
h) Loss functions is binary crossentropy and adam compiler is used.

```
[5]: from keras.models import Sequential

[6]: from keras.layers import Activation, Dropout, Flatten, Conv2D, MaxPooling2D, Dense

[7]: model = Sequential()

     model.add(Conv2D(filters=32,kernel_size=(3,3),input_shape=input_shape,activation='relu'))
     model.add(MaxPooling2D(pool_size=(2,2)))

     model.add(Conv2D(filters=64,kernel_size=(3,3),input_shape=input_shape,activation='relu'))
     model.add(MaxPooling2D(pool_size=(2,2)))

     model.add(Conv2D(filters=64,kernel_size=(3,3),input_shape=input_shape,activation='relu'))
     model.add(MaxPooling2D(pool_size=(2,2)))

     model.add(Flatten())

     model.add(Dense(128))
     model.add(Activation('relu'))

     model.add(Dropout(0.5))

     model.add(Dense(1))
     model.add(Activation('sigmoid'))

     model.compile(loss='binary_crossentropy',
                   optimizer='adam',
                   metrics=['accuracy'])

[8]: model.summary()
```

Fig. 4. CNN model programming

Every dataset was trained for 100 epochs, and the resulting trained models were saved. Total 4 models were prepared and their performance were compared with each other. The same data collection program was used for prediction purposes, but with modifications based on the respective data collection method used to prepare that dataset. 80% of dataset used for training purpose and 20% is used for testing and respective accuracy of model were evaluated (Fig. 4).

4.4 Python Arduino Interfacing

The master-slave concept was used for python-arduino interfacing. After gesture recognition, this command is sent to the computer, where it is processed by the Arduino

microcontroller and the appropriate action is taken. To communicate between Python and Arduino, the Pyfirmata library was used.

Power window mechanism and body control module were studied to create a demo of the power window circuit. Then an arduino circuit was created with a 12v DC motor and an L293D motor control module was used to control the motor (Fig. 5).

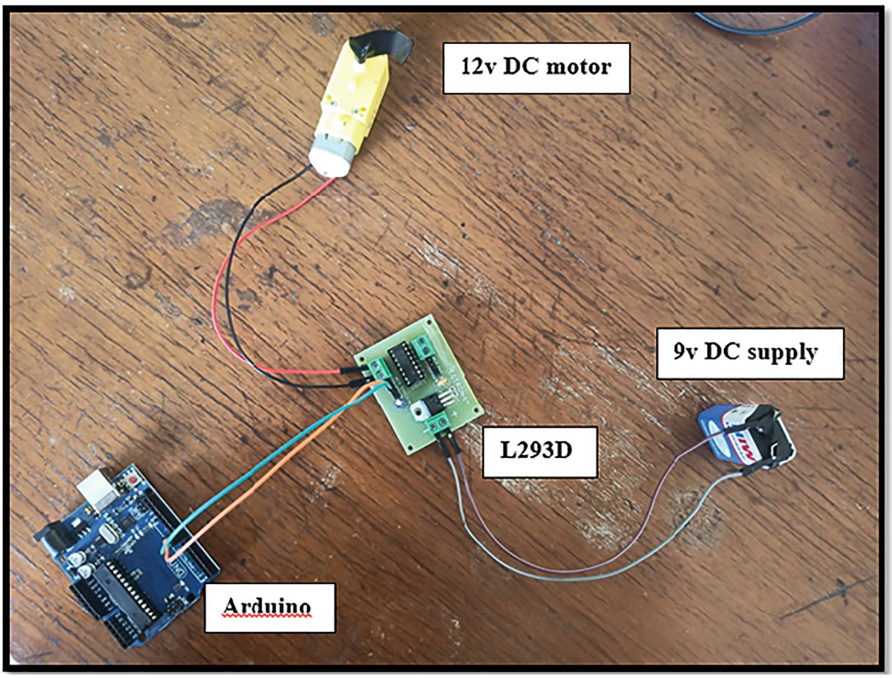

Fig. 5. Python arduino interfacing circuit

4.5 Testing and Validating

The project setup was tested with various people and in various locations such as the home, classroom, in-cabin, car-cabin, and so on. For the various models that were trained using the various datasets mentioned above.

Convex hull and threshold dataset model works well in all environments, whereas online dataset model performed poorly in all models. All other models performed well at home and in the car, but not well in classrooms due to increased noise in the input image (too many people around). In the next section results obtained by trained models are discussed.

5 Result and Discussion

The recognized hand gesture system was tested using hand pictures under various conditions. The methodology describes the overall system's performance using various methods. The limitations of existing systems are overcome by fine-tuning the CNN model parameters and developing an accurate method for background classification. Understanding the system's limitations reveals the focus and direction of future work (Fig. 6 and Table 1).

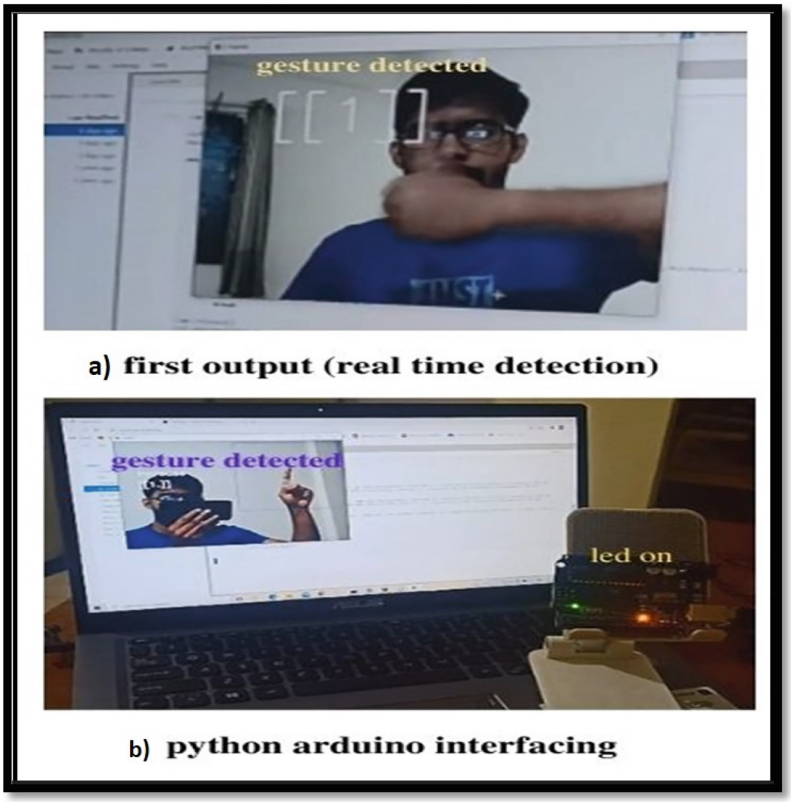

Fig. 6. Output showing detected gesture and arduino output (LED on)

The CNN model with convex hull and threshold dataset performs better than other models because it employs better image separation techniques. This model recognize and execute window control in less than second. However, when we start the system, we must wait at least 4–5 s for the model to calibrate the running average.

This lag can be reduced by decreasing the frame average size (currently 30 frames average is used for convex hull & threshold model), but it will have a significant impact on the system's accuracy.

Table 1. Results showing models, datasets and their % accuracy

SR No.	Model (Dataset)	Accuracy (%)
1	Varying threshold	94
2	Full scale colour image	83
3	Convex hull and threshold	96
4	Kaggle.com	34

6 Conclusions and Future Scope

On the basis of the project's results, it is feasible to develop a hand gesture recognition system using python and open-cv by putting the concepts of hand segmentation and hand detection method.

In conclusion, this system has accomplished several project goals, including: Using python and open-cv, develop a comprehensive system for sensing, identifying, and evaluating hand gestures; develop a hand gesture sign that is coherent with the project's name; and compare various models to determine the most effective background separation and gesture recognition methods.

Additional gestures will be incorporated to the system's future recommendations so that users with different skin tones and hand sizes can easily complete more tasks also background subtraction, semantic segmentation algorithms can also be used to improve performance.

This system can be put in more sophisticated microcontrollers like the Raspberry Pi and Jetson Nano with the help of Tensorflow Lite to reduce system size and for efficient working.

References

1. Fronteddu, G., Porcu, S., Floris, A., Atzori, L.: A dynamic hand gesture recognition dataset for human-computer interfaces. a) DIEE, University of Cagliari, 09123 Cagliari, Italy b) CNIT, University of Cagliari, 09123 Cagliari, Italy
2. 3D-CNN based Dynamic Gesture Recognition for Indian Sign Language Modeling Dushyant Kumar Singh CSED, MNNIT Allahabad, Prayagraj-211003
3. Rabhi, Y., Mrabet, M., Fnaiech, F.: A facial expression controlled wheelchair for people with disabilities. University of Tunis, National Higher School of Engineers of Tunis, Laboratory of Signal Image and Energy Mastery (SIME), 5 Avenue Taha Hussein, P.O. Box 56, Tunis 1008, Tunisia
4. Wang, F., Hu, R., Jin, Y.: Research on gesture image recognition method based on transfer learning. Huaiyin Institute of Technology, Huai'an, China
5. Gopal, N., Bhooshan, R.S.: Content based image retrieval using enhanced SURF. In: 2015 5th National Conference on Computer Vision Pattern Recognition, Image Processing and Graphics, NCVPRIPG 2015 (2016)
6. Mathe, E., Mitsou, A., Spyrou, E., Mylonas, P.: Arm gesture recognition using a convolutional neural network. In: 2018 13th International Workshop on Semantic and Social Media Adaptation and Personalization (SMAP) (2018). https://doi.org/10.1109/smap.2018.8501886

7. Shimada, A., Yamashita, T., Taniguchi, R.-I.: Hand gesture based TV control system—towards both user & machine-friendly gesture applications. In: Proceedings of the 19th Korea-Japan Joint Workshop on Frontiers of Computer Vision (FCV 2013), pp. 121–126, February 2013
8. Keskin, C., Kiraç, F., Kara, Y.E., Akarun, L.: Real time hand pose estimation using depth sensors. In: Proceedings of the IEEE International Conference on Computer Vision Workshops (ICCV 2011), pp. 1228–1234, November 2011
9. Zafrulla, Z., Brashear, H., Starner, T., Hamilton, H., Presti, P.: American sign language recognition with the Kinect. In: Proceedings of the 13th ACM International Conference on Multimodal Interfaces (ICMI 2011), pp. 279–286, November 2011