Savitribai Phule Pune University



*A PRELIMINARY PROJECT REPORT*

*ON*

**Contactless System Navigation Using Dynamic Hand Gesture Recognition By 2D Convolutional Neural Networks and Short-Term Sampling**

Submitted by

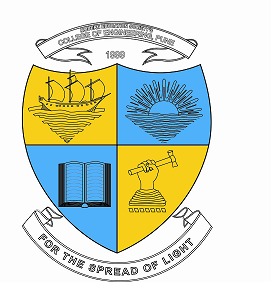
**Abhijeet Prasad 72025849K**

**Satyajit Bhosle 72025595D**

**Abdul Aseem Shaikh 72025890B**

Under the guidance of

**Prof. S. S. Raskar**



Department of Computer Engineering

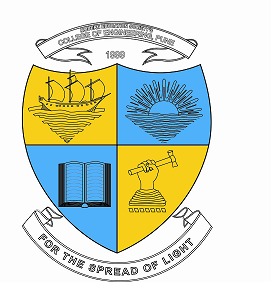
Modern Education Society’s

College Engineering, Pune-411 001

[2022-23]

**MODERN EDUCATION SOCIETY’S**

College of Engineering, Pune 01



**C E R T I F I C A T E**

This is to certify that the Project Entitled

DYNAMIC HAND GESTURE RECOGNITION USING 2D CONVOLUTIONAL NEURAL NETWORKS AND SHORT-TERM SAMPLING

Submitted by

Abhijeet Prasad Exam No: 72025849K

Satyajit Bhosle Exam No: 72025595D

Abdul Aseem Shaikh Exam No: 72025890B

*is a bonafide work carried out by Students under the supervision of Prof. Guide Name and it is submitted towards the partial fulfillment of the requirement of Bachelor of Engineering (Computer Engineering).*

**(Prof. S. S. Raskar)** **(Dr. N. F. Shaikh)**

Guide Head,

Department of Computer Engineering Department of Computer Engineering

**(Dr. M. P. Dale)**

Principal,

MES College of Engineering Pune – 01

Place : Pune

Date :

**ACKNOWLEDGEMENT**

We would like to express our deep sense of gratitude towards our project guide Prof. S. S. Raskar for her support, continuous guidance and for being so understanding and

helpful throughout the course of this project.

We have furthermore to thank Computer Department HOD Dr.(Mrs.) N. F. Shaikh to

encourage us to go ahead and for continuous guidance. We also want to thank Dr. (Mrs.)

S.P. Deore for all her assistance and guidance for preparing the report.

We would like to thank all those, who have directly or indirectly helped us for the completion of the work during this project.

**ABHIJEET PRASAD**

**SATYAJIT BHOSLE**

**ABDUL ASEEM SHAIKH**

**ABSTRACT**

Hand gestures are a natural way for human-robot interaction. Vision based dynamic hand gesture recognition has become a hot research topic due to its various applications.

This project implements a network for hand gesture recognition. The network integrates various modules together to learn both short-term and long-term features from video inputs. A substantial aim is to minimize intensive computation, since handling the video inputs can be an exhausting and computationally expensive task.

Short-term features are learned by segmenting the video input into a fixed number of frames. Then, a frame is randomly selected and represented both, as an RGB(Red, Green, Blue colour channels) image and an optical flow snapshot. These two, i.e., RGB image and optical flow snapshot, are then fused and fed into a convolutional neural network(CNN) for the purpose of feature extraction.

Long-term features are learned by a similar process. The outputs from all the convolutional neural networks are fed into a long short term memory (LSTM) network, and a hence a final classification result is obtained.

As an application to extend the scope of the project, we intend to use the hand gestures classified to perform basic system control.

**INTRODUCTION**

A human gesture can be defined as a sequence of states in the short-lived motion of some part of the body, most commonly the hands and face. Consider the wave of a hand in order to open a door controlled by sensors. The gesture has an initial position and a final position, with many small states in the transition from the initial state to the final state. For example, for a left-to-right wave, the hand is initially at the left, and is moved to the right in a quick and swift movement. This very well explains the meaning of a gesture.

Existing techniques of hand gesture recognition can be divided into two broad categories on the basis of method of obtaining input – vision-based techniques and non-vision based techniques. Vision-based techniques usually imply the use of cameras or motion sensors to detect movement and thus extract input. No extra equipment is generally required in this approach. Non-vision based techniques commonly use hardware equipment to extract input, such as wearable devices. These wearable devices are equipped with various types of mechanical or optical sensors, that translate movement of the body part into electrical signals.

Vision-based gesture tracking presents itself to be a more practical and feasible measure, since there is no need for the user to have an extra hardware device(e.g. wearables) for motion to be detected – as long as the motion is in the tracking frame of the sensor, it can be taken as an input for gesture recognition.

It is quite common knowledge that computer systems and applications can be controlled(or navigated) by the use of remote-like devices that generate signals which are tracked by the system through sensors, and the respective operations are performed by the system. Consider a Playstation Move motion controller for a clearer picture. But what if we could eliminate the need for these physical devices and achieve truly contact-less system control, through mere hand gestures in front of a camera or similar sensors?

**MOTIVATION**

Traditional(or conventional) methods of human-computer interaction, namely, keyboards, pointing devices, and similar physical input tools, and the more popular and commonly-used touchscreens, are only as useful through the application of pressure through physical touch.

The recent pandemic has driven a major argument about the substantial and often, indispensable need for contact-less operation of systems, irrespective of the scale of the industry. Say for example, supply chain systems that are handled by scores of employees in the span of a day.

It is quite fundamental knowledge now, that how public systems that work on physical contact can pose as potential vectors for diseases. Hand gesture recognition can turn out to be an excellent solution to this problem; facilitating truly contact-less interaction with computing devices, and at the same time providing dependable and respectable efficiency.

**PROBLEM DEFINITION**

To explore various techniques of recognition of hand gestures so as to study a way of efficient human-computer interaction.

To elaborate on a technique that conforms to our purposes and requirements of classifying a set of pre-determined hand gestures through a video input, i.e., using 2-dimensional convolutional neural networks and long-short term memory network.

To configure a way to effectively map the hand gestures classified to a corresponding operation performed in order to achieve basic system control through gestures.

**LITERATURE SURVEY**

Zhang, *et al.* Designed a 3-dimensional convolutional neural network model(CNN) [1], this study applies a deep learning method to recognise hand gestures. 3D Convolution neural network can be seen as a variant of 2D convolution neural network extending 2dimension filter into 3 dimensions. This 3D filter shall slide in 3 directions to extract low-level features and its output’s shape is a 3dimension space like a cuboid Uses the Jester V1.0 hand gesture dataset to train the model. According to the result of the training experiment, it got an average accuracy of 90%.

Wang *et al.* developed temporal segment networks (TSN) in [2], for action recognition. In the TSN model, each input video sample is divided into a number of segments and a short snippet is randomly selected from each segment. The snippets are represented by modalities such as RGB frames, optical flow and RGB differences. Convolutional neural networks (ConvNets) that are used to learn these snippets all share parameters. The class scores of different snippets are fused by the segmental consensus function to yield segmental consensus, which is a video-level prediction. Predictions from all modalities are fused to produce the final prediction. Experiment showed that the model not only achieved very good action recognition accuracy but also maintained reasonable computation cost.

Min, *et al.* [3] formulate gesture recognition as an irregular sequence recognition problem and aim to capture long-term spatial correlations across point cloud sequences. A novel and effective PointLSTM is proposed to propagate information from past to future while preserving the spatial structure. The proposed PointLSTM combines state information from neighboring points in the past with current features to update the current states by a weight-shared LSTM layer. This method can be integrated into many other sequence learning approaches. In the task of gesture recognition, the proposed PointLSTM achieves state-of-the-art results on two challenging datasets (NVGesture and SHREC'17) and outperforms previous skeleton-based methods. To show its advantages in generalization, we evaluate our method on MSR Action3D dataset, and it produces competitive results with previous skeleton-based methods.

Tang, *et al.* [4] combined image entropy and density clustering to exploit the key frames from hand gesture video for further feature extraction, which can improve the efficiency of recognition. Moreover, a feature fusion strategy is also proposed to further improve feature representation, which elevates the performance of recognition. To validate our approach in a "wild" environment, we also introduce two new datasets called HandGesture and Action3D datasets. Experiments consistently demonstrate that our strategy achieves competitive results on Northwestern University, Cambridge, HandGesture and Action3D hand gesture datasets

Cheng, *et al.* [5] proposed a Dynamic Graph-Based Spatial-Temporal Attention (DG-STA) method for hand gesture recognition. The key idea is to first construct a fully-connected graph from a hand skeleton, where the node features and edges are then automatically learned via a self-attention mechanism that performs in both spatial and temporal domains. We further propose to leverage the spatial-temporal cues of joint positions to guarantee robust recognition in challenging conditions. In addition, a novel spatial-temporal mask is applied to significantly cut down the computational cost by 99%. We carry out extensive experiments on benchmarks (DHG-14/28 and SHREC'17) and prove the superior performance of our method compared with the state-of-the-art methods.

Zhang, Wang, Lan [6] Presents a novel deep learning network for hand gesture recognition. The network integrates several well- proved modules together to learn both short-term and long-term features from video inputs and meanwhile avoid intensive computation. To learn short-term features, each video input is segmented into a fixed number of frame groups. A frame is randomly selected from each group and represented as an RGB image as well as an optical flow snapshot. These two entities are fused and fed into a convolutional neural network (ConvNet) for feature extraction. The ConvNets for all groups share parameters. To learn longterm features, outputs from all ConvNets are fed into a long short-term memory (LSTM) network, by which a final classification result is predicted. The new model has been tested with two popular hand gesture datasets, namely the Jester dataset and Nvidia dataset. The robustness of the model has also been proved with an augmented dataset with enhanced diversity of hand gestures. Uses the Jester dataset and Nvidia Dataset Achieved an accuracy if around 95%(Jester dataset) and 85%(Nvidia dataset).

**SOFTWARE REQUIREMENT SPECIFICATION**

**Introduction :**

**Project Scope:**

We trying to build a system which can provide contactless interaction between user and physical hardware of any machine where user need to navigate through the interface like our personal computer, laptop, and ATM machine.

**User Classes and Characteristics:**

Users who can perform hand gesture can make use of this system to navigate through the OS without making any physical contact to the physical hardware. User will need to perform meaning or proper gesture to navigate through OS.

**Assumptions And Dependencies:**

We are going to assume that user’s machine have already install all the necessary software like .Net frame work, C++ redistribution (2019) and all the necessary driver related to webcam or any optical camera for capturing the user’s gestures.

**Functional requirements**

Feature vector extraction: For feature vector extraction we require the CNN model. For our project we are going to use 2D CNN model for feature extraction.

Video sequence: For video sequence we need the webcam. Which will capture the gesture/moment of user’s hand.

Gesture classification: For classification we required RNN model. We will be using LSTM model for gesture classification.

Mapping gesture to OS operation: For Mapping particular gesture to perform OS navigation operation we will need the OpenCV library.

**External Interface Requirements:**

User Interfaces: User need to the perform the valid gesture to navigate through operating system.

Hardware Interface: Webcam is required for capturing the video stream of the user who is performing the gesture. And webcam should have recording speed of 30 FPS at least

Software Interface: OpenCV is require for mapping the particular gesture to the OS navigation operation.

**Non-Functional requirements:**

Performance Requirement: Video stream should be capture at the speed of 30FPS.

Safety Requirement: No safety risk expected.

Security Requirement: No user personal data or any image will be saved by the system.

Software quality attributes: Usability, Reliability, Protability, Testibility, Flexibility;

**System Requirement:**

Software Requirements:

Operating System: Windows 10.

Programming Language: Python.

Library: OpenCV, TensorFlow, Pandas, Numpy,Keras.

Hardware Requirements:

CPU: DUO core (of 4 generation or above).

Ram: 4GB or above.

Graphic Card (for training & testing and neural network ): GT 1050 or Intel 520.

Storage: 500GB HDD or SDD.

Webcam: should able to recording at 30FPS (Frame Per second).

**Analysis Models: SDLC description**

We will be following the water fall model for our software development life-cycle. The waterfall model is a continuous software development model in which development is seen as flowing steadily downwards (like a waterfall) through the steps of requirements analysis, design, implementation, testing (validation), integration, and maintenance.

Linear ordering of activities has some significant consequences. First, to identify the end of a phase and the beginning of the next, some certification techniques have to be employed at the end of each step. Some verification and validation usually do this mean that will ensure that the output of the stage is consistent with its input (which is the output of the previous step), and that the output of the stage is consistent with the overall requirements of the system.

Phases in Waterfall Model:

**Requirements analysis and specification phase:** The aim of this phase is to understand the exact requirements of the customer and to document them properly. Both the customer and the software developer work together so as to document all the functions, performance, and interfacing requirement of the software. It describes the "what" of the system to be produced and not "how."In this phase, a large document called **Software Requirement Specification (SRS)** document is created which contained a detailed description of what the system will do in the common language.

**Design Phase:** This phase aims to transform the requirements gathered in the SRS into a suitable form which permits further coding in a programming language. It defines the overall software architecture together with high level and detailed design. All this work is documented as a Software Design Document (SDD).

**3. Implementation and unit testing:** During this phase, design is implemented. If the SDD is complete, the implementation or coding phase proceeds smoothly, because all the information needed by software developers is contained in the SDD.

**4. Integration and System Testing:** This phase is highly crucial as the quality of the end product is determined by the effectiveness of the testing carried out. The better output will lead to satisfied customers, lower maintenance costs, and accurate results. Unit testing determines the efficiency of individual modules. However, in this phase, the modules are tested for their interactions with each other and with the system.

**5. Operation and maintenance phase:** Maintenance is the task performed by every user once the software has been delivered to the customer, installed, and operational.

**System Implementation Plan:**

**Objectives:**

Create dataset of video containing different gesture.

Decide mapping of gesture to OS navigation operation.

Setting up development environment.

Create CNN model (for feature extraction).

Create RNN model (for classification).

Create function to execute classified gesture.

Create General overlay for system (UI).

System Integration.

Testing and debugging.

**Outline deliverable:**

Dataset and mapping gesture to navigation operation has to be done in first.

Then only we start creating feature group.

Setting up the development environment with all required dependencies.

**Tasks:**

Dataset and mapping gesture.

Setting up development environment.

CNN model development.

RNN model development.

Training and testing.

Debugging.

Final tweaking.

Report.

**SYSTEM DESIGN**

**System Architecture:**

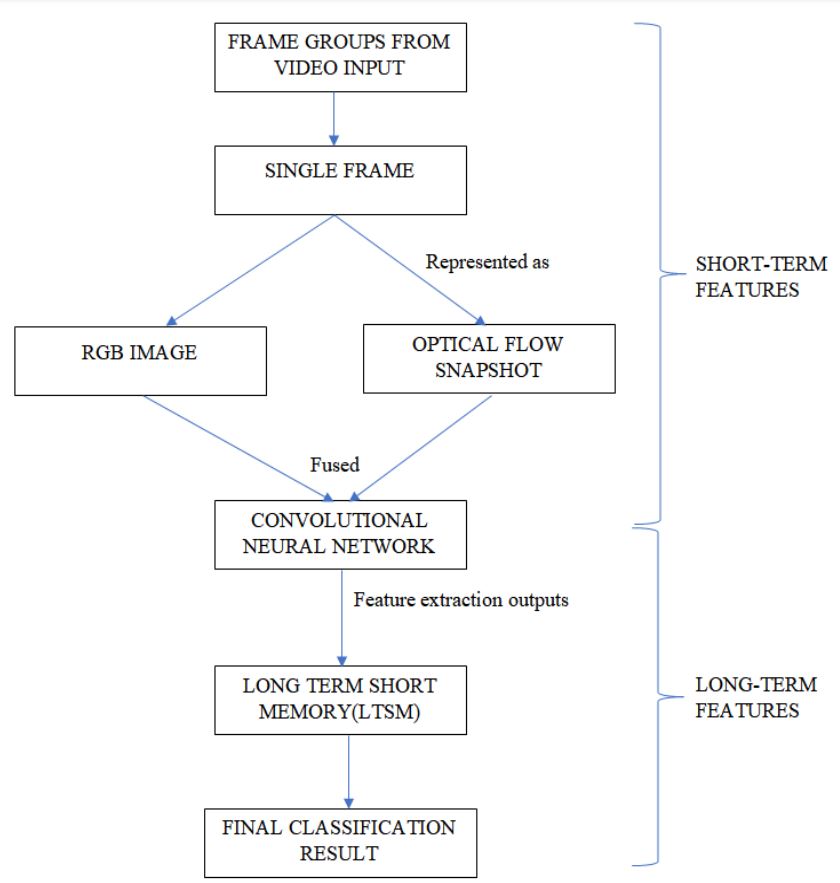


Figure. Architecture Diagram.

**Dataflow Diagrams:**

Level 0 Dataflow diagrams:

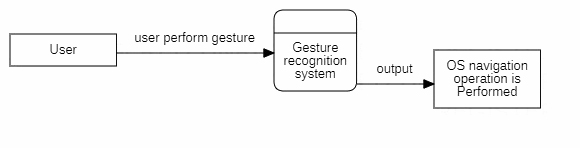


Figure. Level 0 dataflow diagram.

Level 1 Dataflow diagrams:

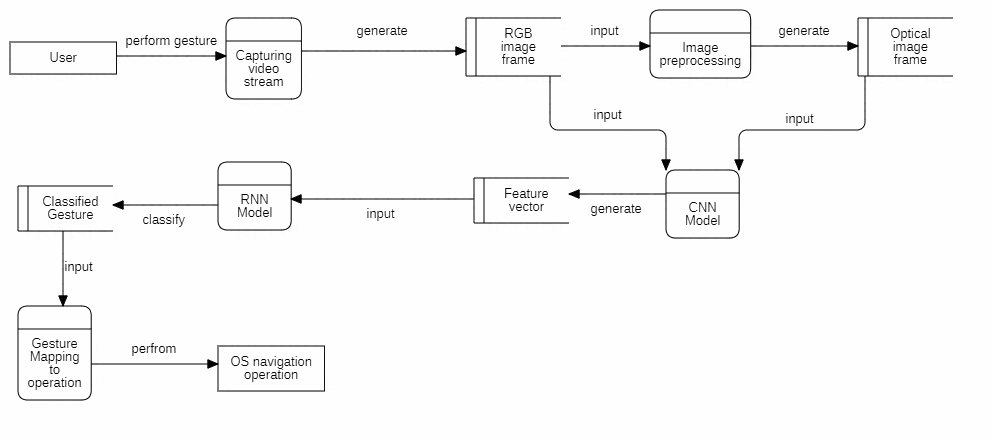


Figure. Level 1 Dataflow diagram.

Level 2 Dataflow diagrams:

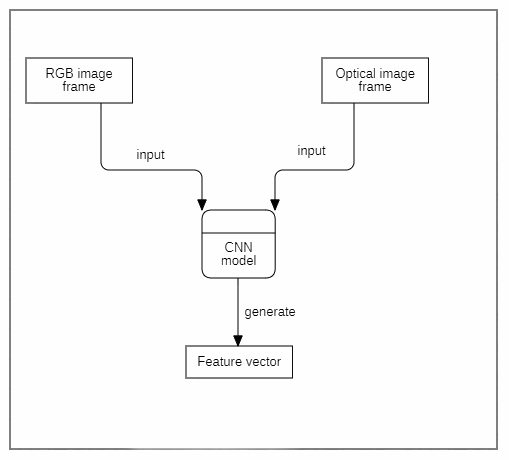


Figure. Level 2 Dataflow Diagram: generating feature vector.

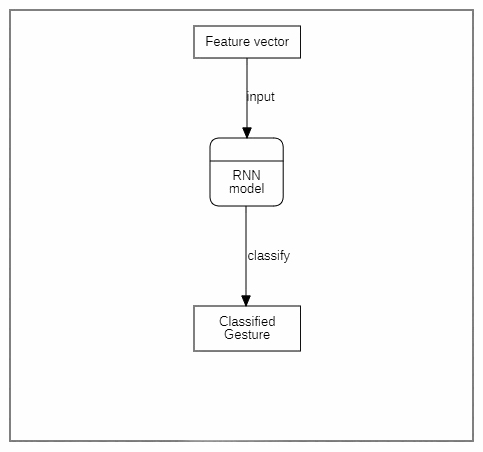


Figure. Level 2 dataflow diagram: classification.

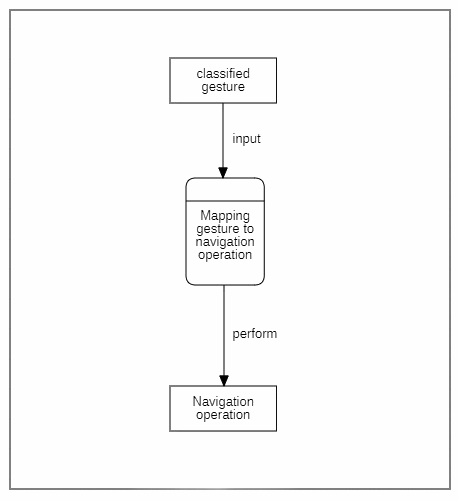


Figure. Level 2 dataflow diagram: mapping gesture to operation

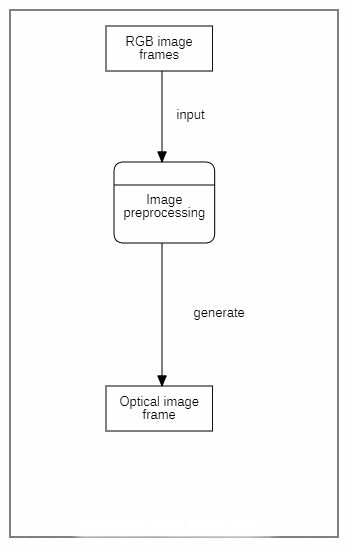


Figure. Leve 2 dataflow diagram: generating optical image.

**Entity Relationship Diagram:**

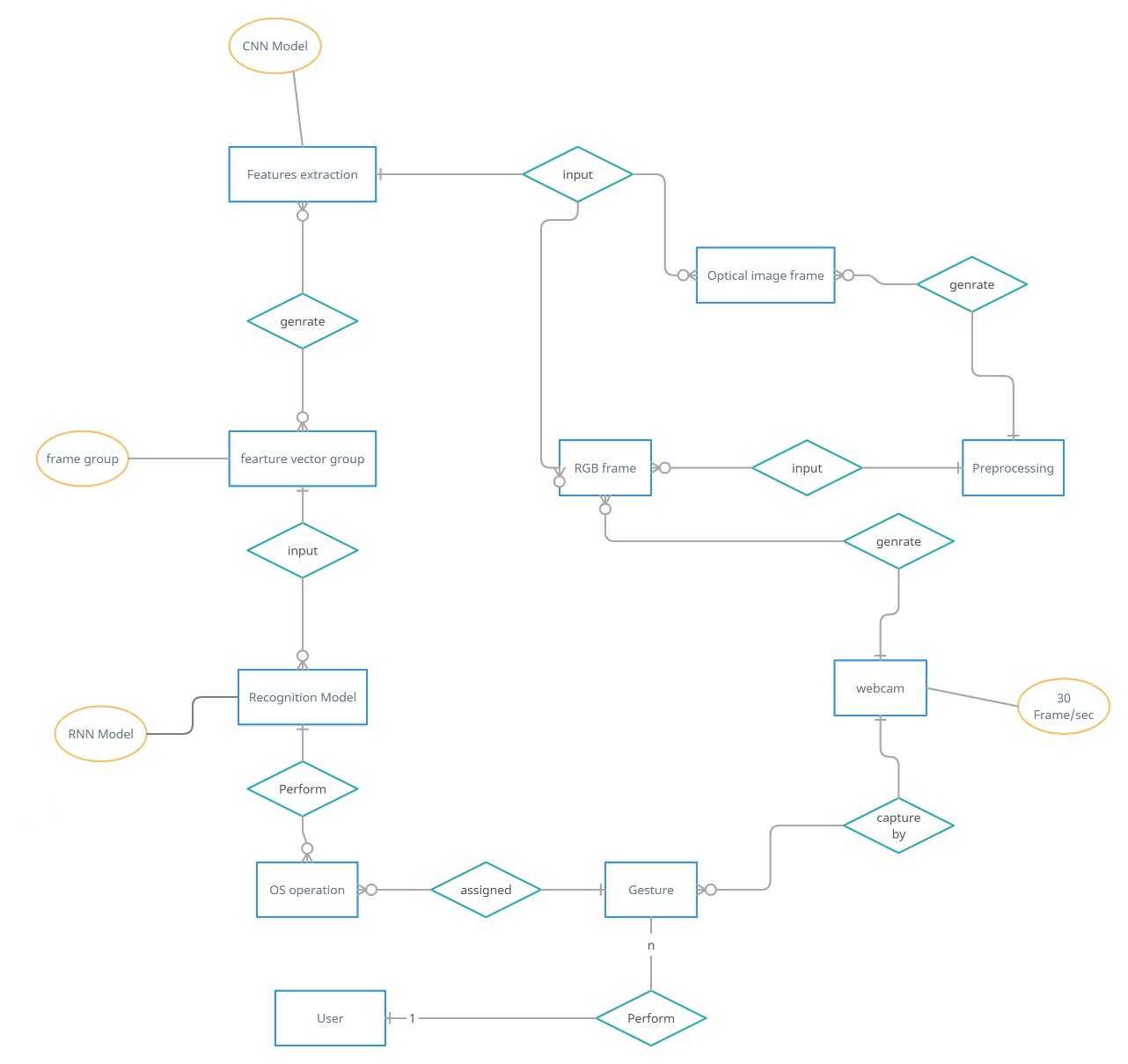


Figure. Entity Relationship diagram

**UML Diagram:**

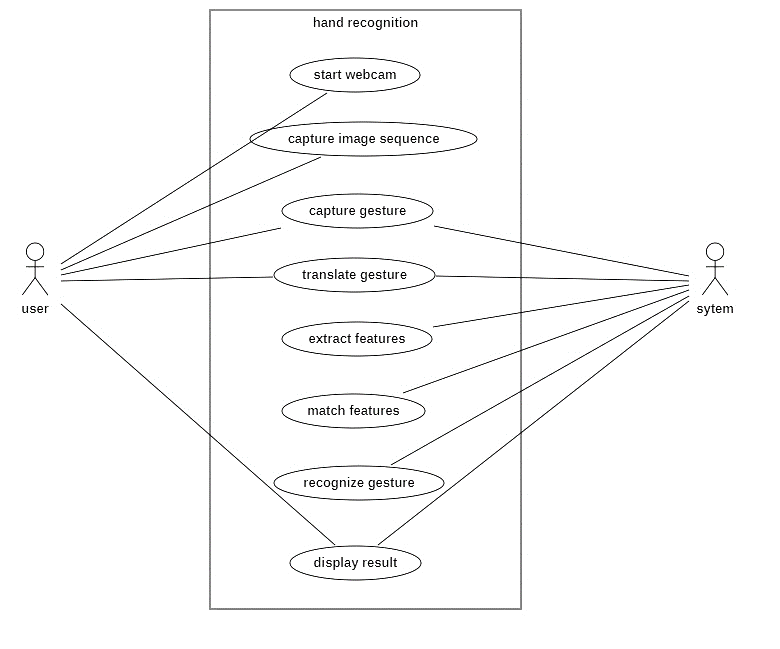
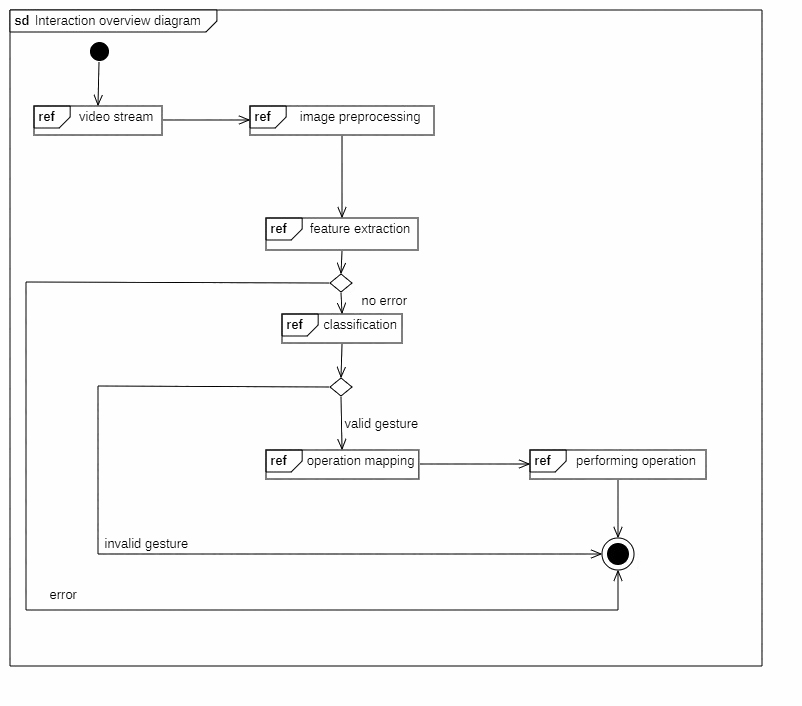


Figure. Use Case diagram



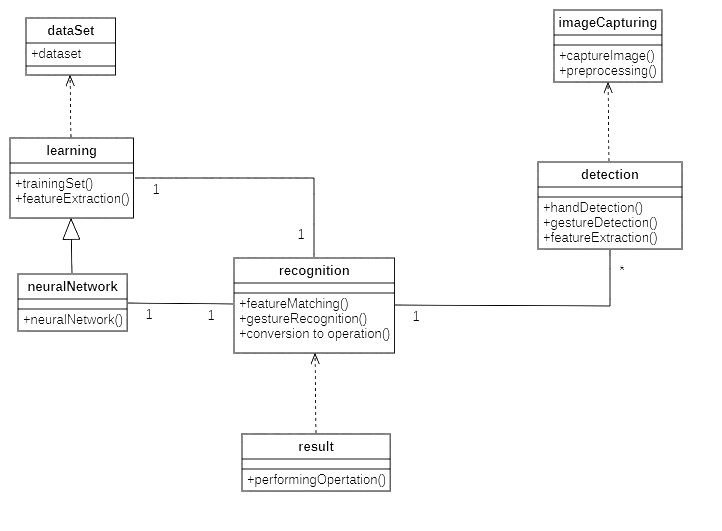


Figure. Class Diagram

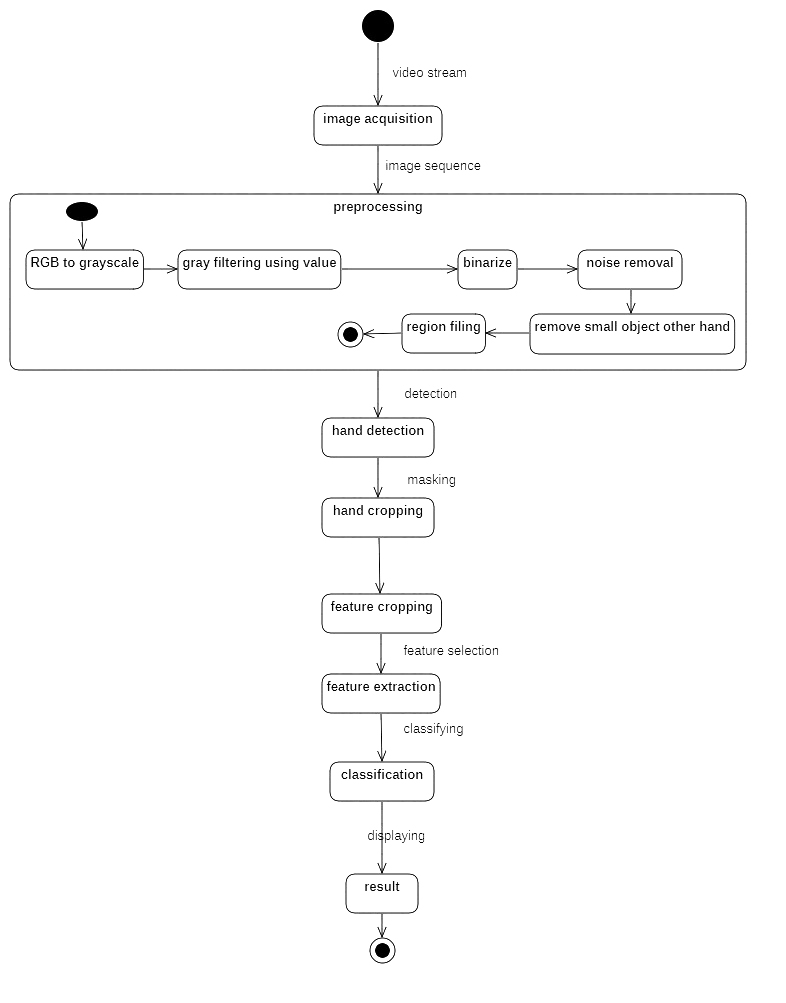


Figure. State Diagram

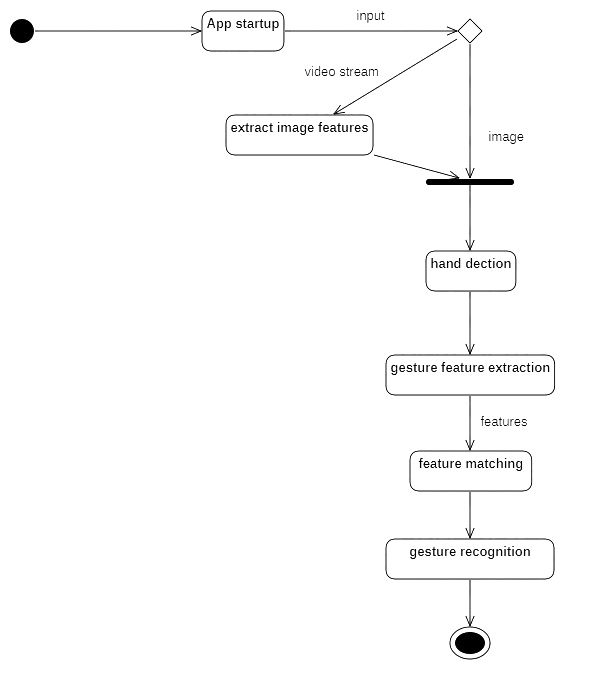


Figure. Activity Diagram

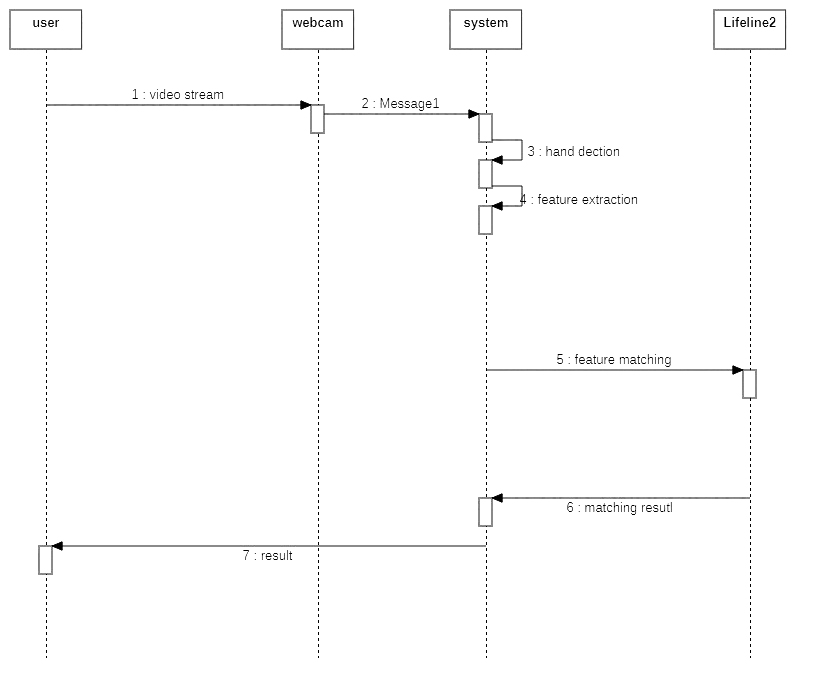


Figure. Sequence Diagram

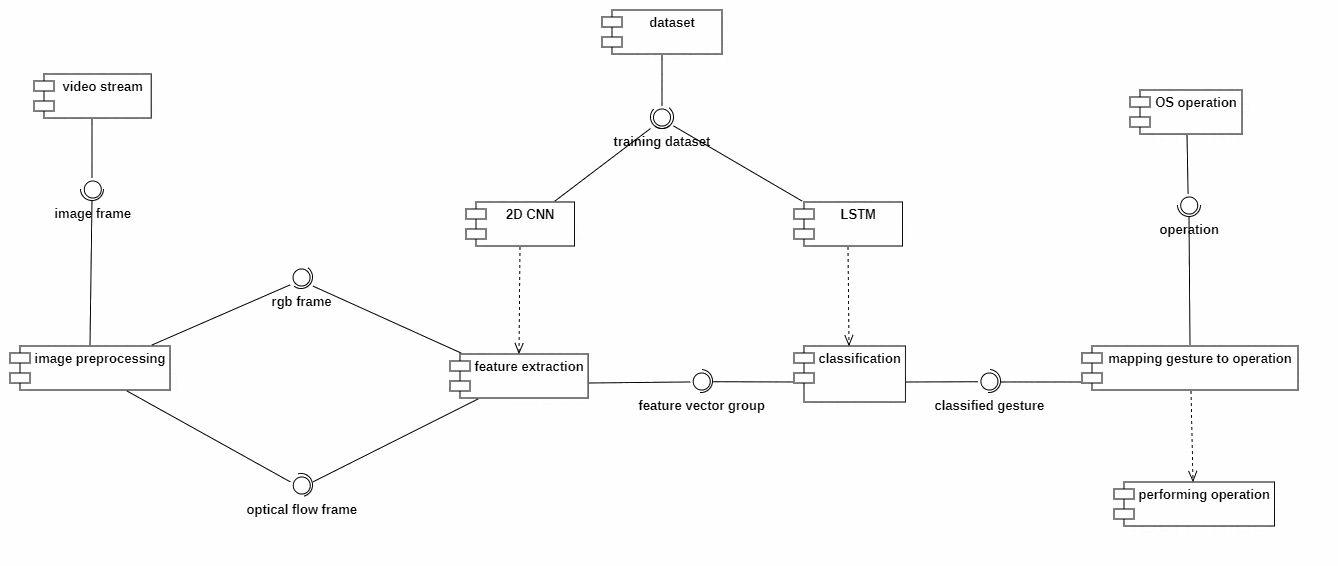


Figure. Component Diagram

Figure. Interaction Over View.

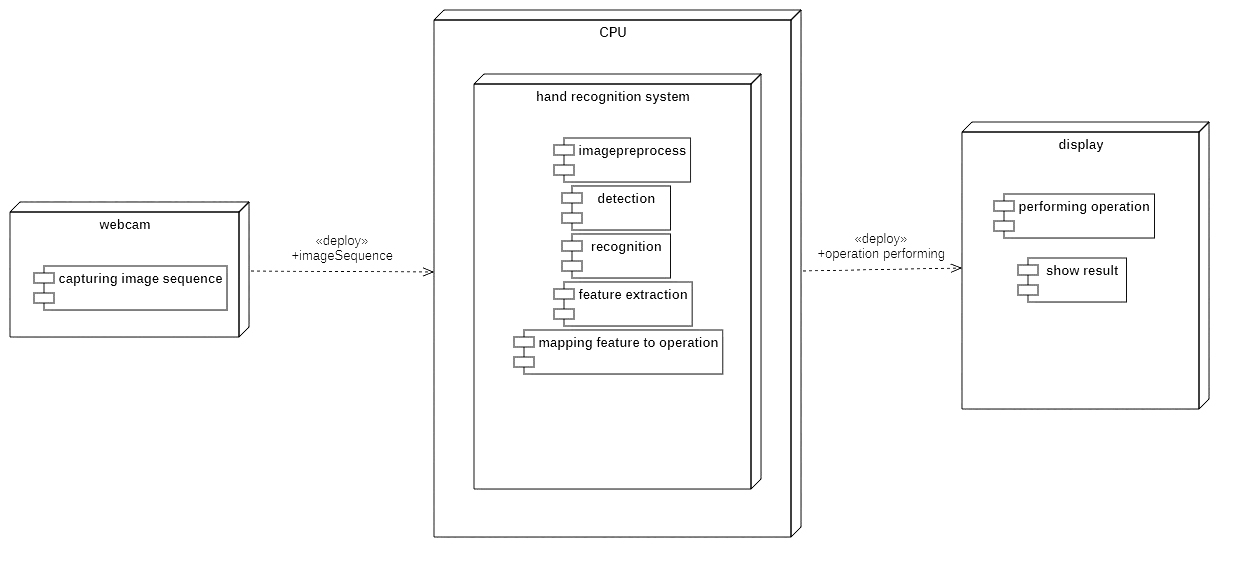


Figure. Deployment Diagram

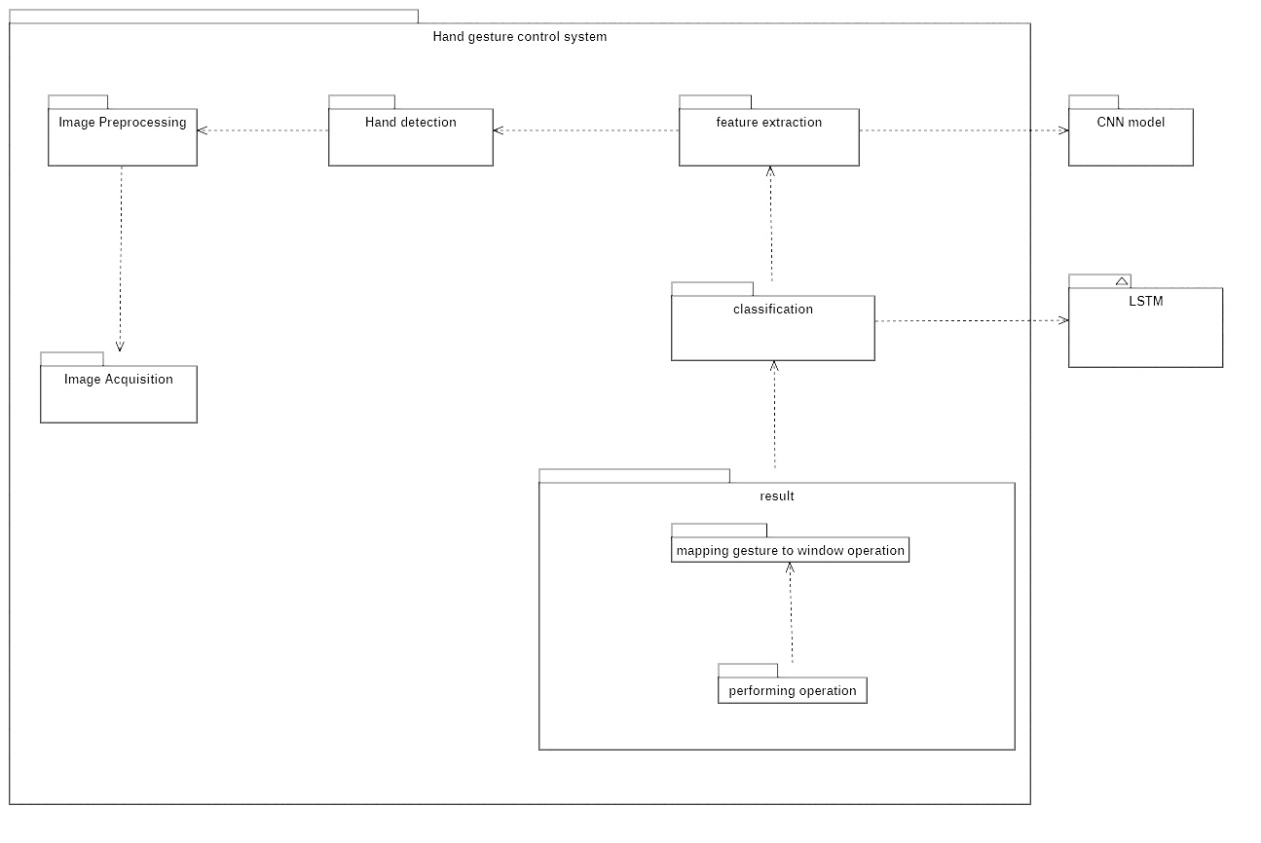


Figure. Package diagram

**OTHERS SPECIFICATION**

**Advantages**

1. Through this system we can establish basic system control without actually having to

come in contact with the system physically.

2. The mode of input, i.e., video tracking, can potentially make the computation quite

intensive; but the algorithm used for classification of gestures tries to handle it such

that the process does not become too computationally draining.

**Limitations**

1. The system may take a considerable amount of time(say, a few seconds) to correctly

classify the gesture made based on how the algorithm is trained. This can cause a

delay in the intended operation that is to be performed.

2. The system, as simple as it might be made to be, might still not be as intuitive to some

users. So, some literacy on how to use such systems needs to be provided.

3. The hand gestures made by one particular user may vary significantly in comparison

to those made by another user; this is more of a subjective choice as to how a

particular user thinks and can be difficult to generalize. It might occur that the

particular set of gestures that the system is trained to classify might be different to

what a user actually performs.

**Applications**

1. The hand gesture recognition system can be employed in public systems, such as an

automated teller machine(ATM) where potentially scores of people might handle the

same interface through physical touch. This can potentially prevent the spread of any

physical contact-intensive diseases.

2. The system can potentially help to achieve truly contact-less operation at an even

greater scale, such as workplaces where employees interact with a common interface

through physical touch, albeit through objects like ID cards; or supply chain workers

who might handle the same interface throughout the day.

**CONCLUSION AND FUTURE WORKS**

In future we can map more gesture to different navigation operation which can we user more control of the screen. And we can decrease the latency by increasing the training dataset and Neural Network nodes.