Contents

List of Figures

Chapter 1

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. wearable) 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?

Chapter 2

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.

Chapter 3

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 convolutions 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.

Chapter 4

LITERATURE SURVEY

Zhang, et al. Designed a 3-dimensional convolutions neural network model(CNN)[9], 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. [7] developed temporal segment networks (TSN) in [7], 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. Convolutions 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. [6] 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 Hand Gesture 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. [1] 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

Zhang, Wang, Lan [10] 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 long term 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

Chung et al. [2] proposed a method for hand gesture recognition using deep convolutional neural networks(CNNs). They used a webcam to track the hand region. Firstly, they used skin color detection and morphology to remove unnecessary background information(noise) from the image, and subtract the background to obtain the region of interest. To avoid background influences affecting the region of interest, they used kernelized correlation filters(KCF) algorithm to track the detected region of interest. The image was then entered into the deep convolutional neural network. Two deep CNN architectures were developed in the study. The whole process was repeated in order to obtain an instant effect; with the system continuing execution as long as the hand is in the camera range. In this study, the training dataset reached a recognition rate of 99.90

Rahim et al [4] proposed a approach which segments hand gestures by comparing the segmentation methods of YCbCr, SkinMask and HSV(Hue, Satuation, Value). The chroma red(Cr) component is extracted from the YbCbCr model, after which operations like binarization, erosion and hole filling are carried out. Color segmentation is applied to SkinMask procedure that detects the pixels that match with the color of the hand in the frame. By the HSV process, threshold masking determines the dominant features of the input. The Softmax classification algorithm was used for the pupose of classification of hand gestures. Convolutional neural network was used to extract features.

Xu et al [8]. proposed a recognition method of both static and dynamic hand gestures. Firstly, for static hand gesture recognition, starting from the hand gesture contour extraction, the palm centre is identified by Distance Transform (DT) algorithm. The fingertips are localized by employing the K-Curvature-Convex Defects Detection algorithm (K-CCD). On the basis, the distances of the pixels on hand gesture contour to palm centre and the angle between fingertips are considered as the auxiliary features to construct a multimodal feature vector, and then recognition algorithm is presented to robustly recognize the static hand gestures. Secondly, combining Euclidean distance between hand joints and shoulder centre joint with the modulus ratios of skeleton features, this paper generates a unifying feature descriptor for each dynamic hand gesture and proposes an improved dynamic time warping (IDTW) algorithm to obtain recognition results of dynamic hand gestures.

Sarma [5], Bhuyan presented a model-based method for hand gesture recognition using convolutional neural network. The model was fed with trajectory-to-contour based images that were obtained from isolated trajectory gesture though segmentation and tracking of the hand motion, and the hand motion trajectory was estimated. Deep learning techniques were used to learn image features hierarchically from local to global with multiple layers of abstraction based on the sample images in the dataset. In this method, feature learning capability of CNN architecture gave quite commendable results, while tested on three different datasets.

Chapter 5

SOFTWARE REQUIREMENT SPECIFICATION

5.1 Introduction

5.1.1 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.

5.1.2 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.

5.1.3 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.

5.2 Functional Requirements

5.2.1 Feature vector extraction

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.

5.2.2 Video sequence

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

5.2.3 Gesture classification

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

5.2.4 Mapping gesture to OS operation

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

5.3 External Interface Requirements

5.3.1 User Interfaces

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

5.3.2 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

5.3.3 Software Interface

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

5.4 Nonfunctional Requirements

5.4.1 Performance Requirement:

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

5.4.2 Safety Requirement:

No safety related risk are expected.

5.4.3 Security Requirement

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

5.4.4 Software quality attributes

Usability, Reliability, Portability, Testable, Flexibility.

5.5 System Requirements

5.5.1 Software Requirement

- Operating System: Windows 10.
- Programming Language: Python.
- Library: OpenCV, TensorFlow, Pandas, Numpy, Keras.

5.5.2 Hardware Requirement

- 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).

5.6 Analysis Models: SDLC Model to be applied

We will be following the water fall model for our software development lifecycle. 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).

- 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.

- 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.

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

5.7 System Implementation Plan

5.7.1 Objective

- 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.

5.7.2 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.

5.7.3 Tasks

- Dataset and mapping gesture.
- Setting up development environment.
- CNN model development.

- RNN model development.
- Integration.
- Training and testing.
- Debugging.
- Final tweaking.
- Documentation.

Chapter 6

SYSTEM DESIGN

6.1 System Architecture

We will be video stream of user performing the gesture. Then we will be creating the RGB image and Optical flow image using image prepossessing for each image. The RGB and optical flow image will be grouped together and send it to CNN model for feature extraction. CNN model will take RGB and optical flow image frame by frame as an input for feature extraction. CNN model will create feature vector groups.

Feature group are the characteristic of the input provided to the CNN model. The feature vector will be taken as the input for RNN model. RNN model are used for classification based on the feature vector group. RNN model will classify the gesture based on the feature vector. Based on the classified gesture the particular navigation operation is executed.

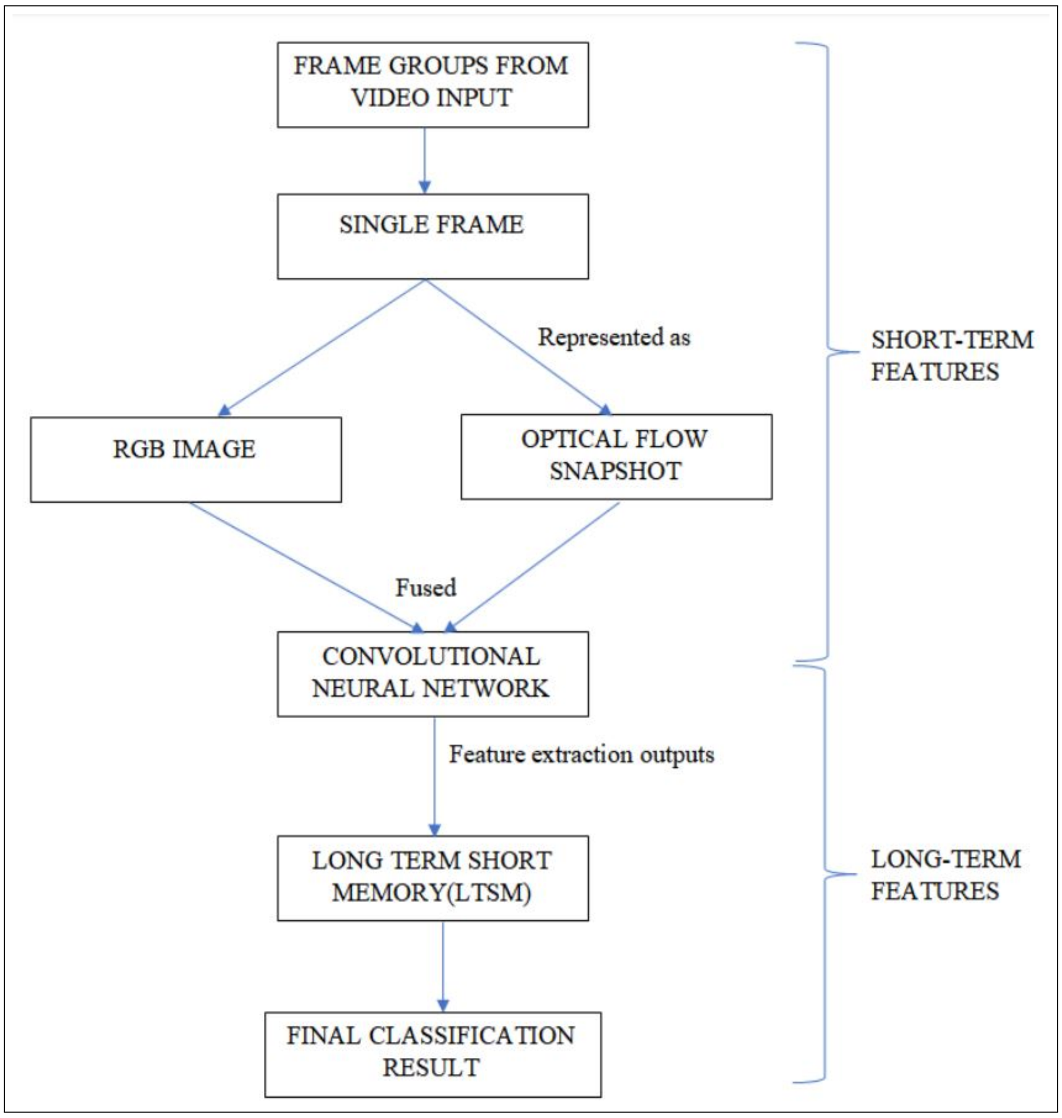
FRAME GROUPS FROM VIDEO INPUT
SINGLE FRAME
Represented as
RGB IMAGE
OPTICAL FLOW SNAPSHOT
Fused
CONVOLUTIONAL NEURAL NETWORK
Feature extraction outputs
LONG TERM SHORT MEMORY(LTSM)
FINAL CLASSIFICATION RESULT
SHORT-TERM FEATURES
LONG-TERM FEATURES

Figure 6.1: Architecture diagram

6.2 Data Flow Diagram

A data-flow diagram is a way of representing a flow of data through a process or a system. The DFD also provides information about the outputs and inputs of each entity and the process itself. A data-flow diagram has no control flow — there are no decision rules and no loops.

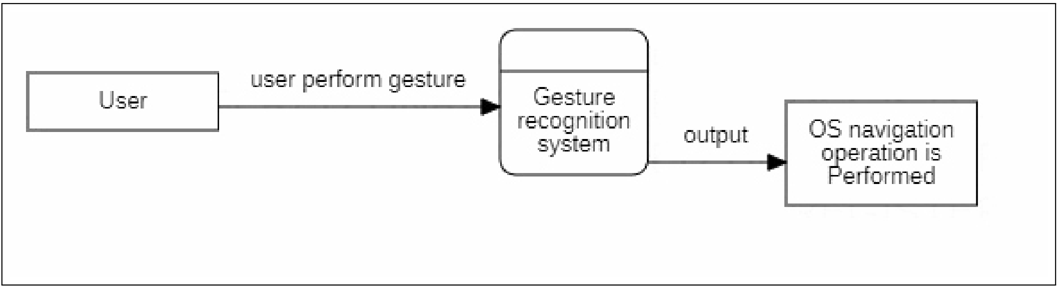
User
user perform gesture
Gesture recognition system
output
OS navigation operation is Performed

Figure 6.2: Data flow level 0 diagram

In figure 6.2 user will perform gesture. It will be capture by the webcam. video stream in given as input to the system. Then system recognise the gesture and system will perform the related navigation operation to the classified gesture.

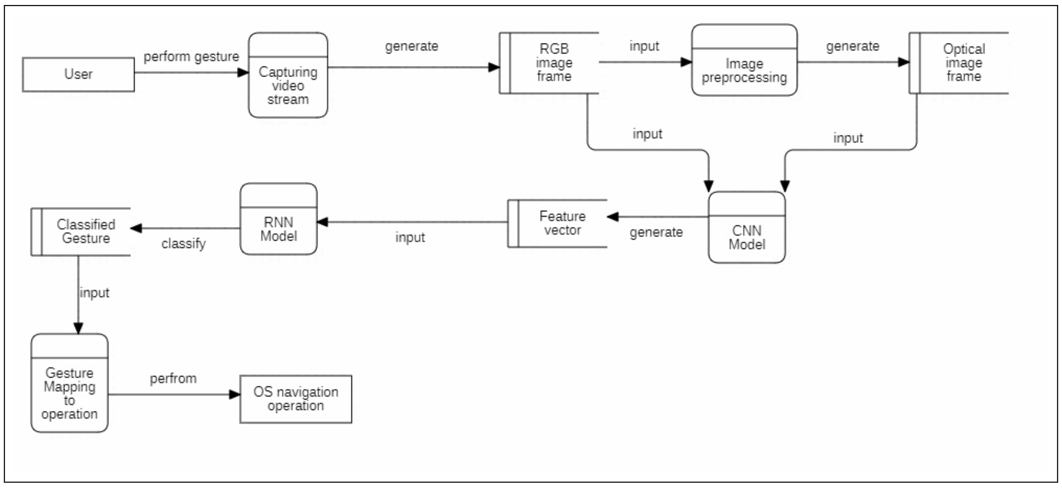

Figure 6.3: Data flow level 1 diagram

In figure 6.3 From the capture video stream we will be generating the RGB image frame and it will be given as input to the image pre-processing for generating the optical flow image of input RGB frame.

Then RGB image frame and optical image frame is grouped together. And CNN model we take the grouped images as an input and generates the feature vector group.feature vector groups are the group of characteristics extracted from the images.

Then RNN Model will take the feature vector groups and classify the gesture performed by the user. We will be already mapping the gesture to the corresponding navigation operation like fist will be mapped to click, right swipe will be mapped to forward page, etc.

After gesture get classified system will be executing that navigation operation on real time basis.

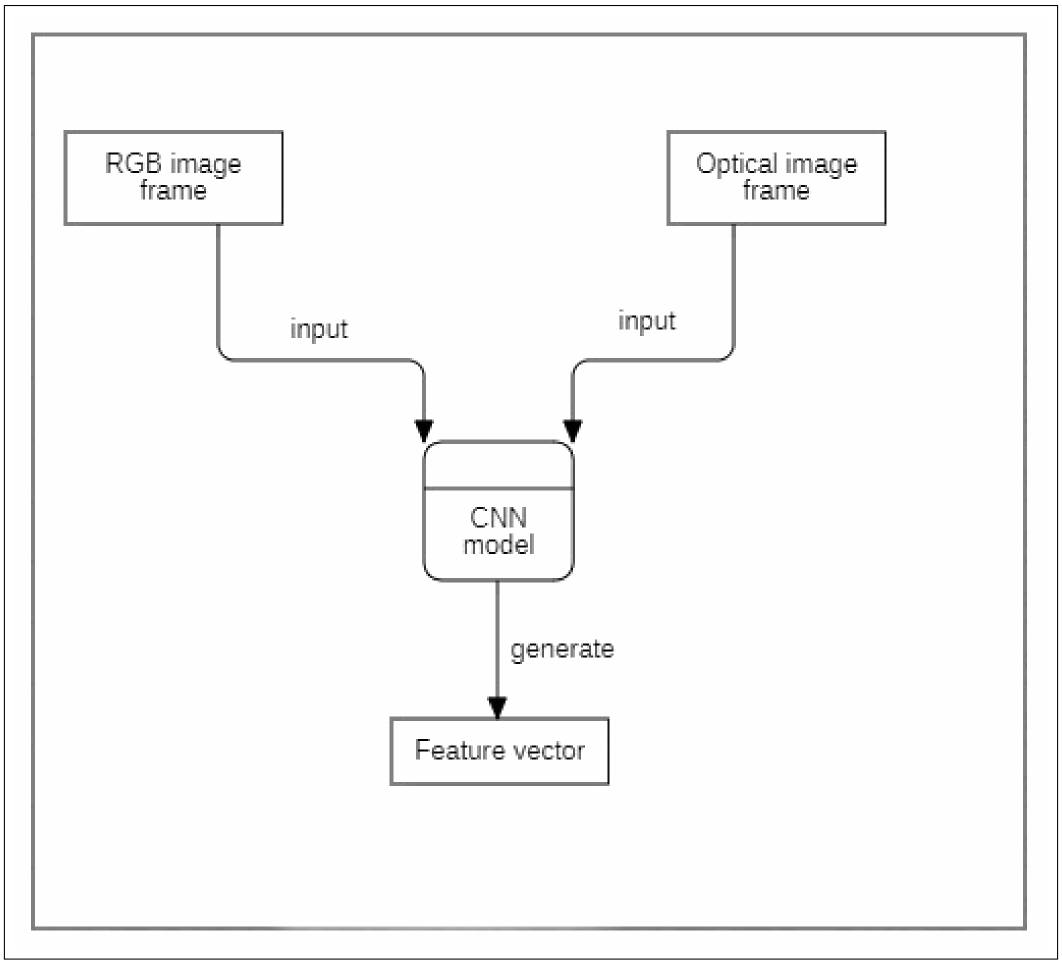

Figure 6.4: Data flow level 2 (1) diagram

The figure 6.4 show the input and output generated by the CNN model. For input it is taking RGB image frame and optical image frame and generating output Feature vector.

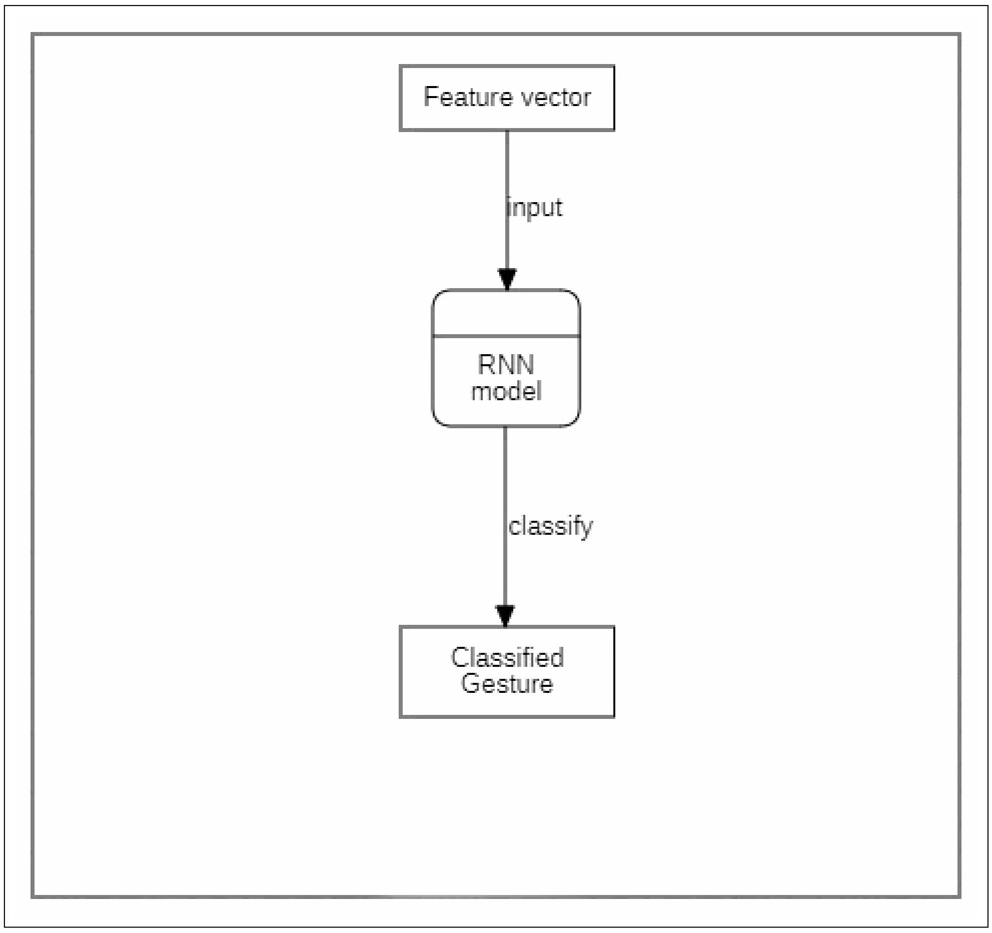

Figure 6.5: Data flow level 2 (2) diagram

The figure 6.5 show the input and output generated by RNN model. It takes the feature vector generated by CNN model as an input and classifies the gesture perform by the user.

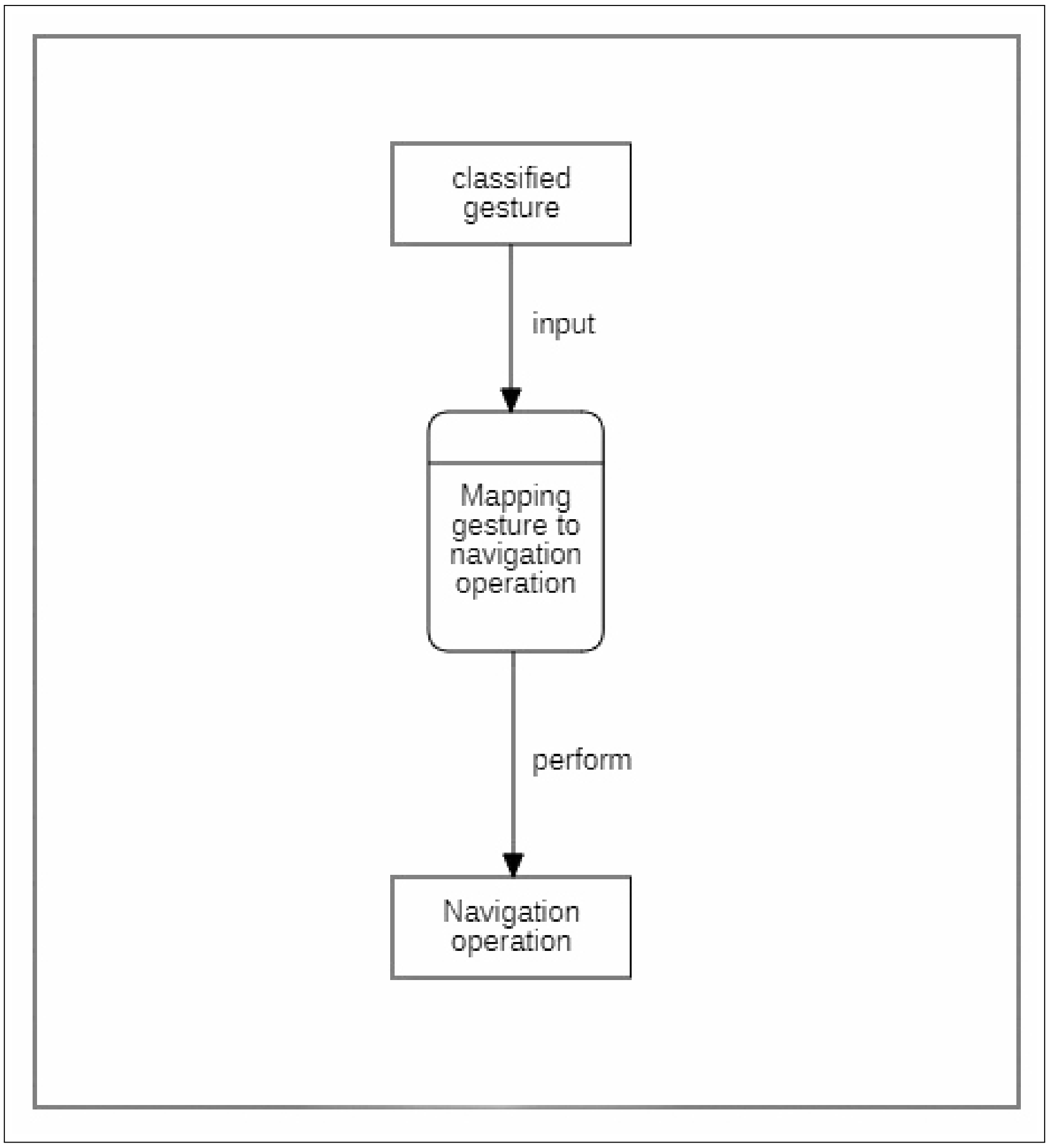

Figure 6.6: Data flow level 2 (2) diagram

The figure 6.6 show the mapping of the classified gesture to the corresponding OS navigation operation.

6.3 Entity Relationship Diagram

An entity–relationship model describes interrelated things of interest in a specific domain of knowledge. A basic ER model is composed of entity types and specifies relationships that can exist between entities.

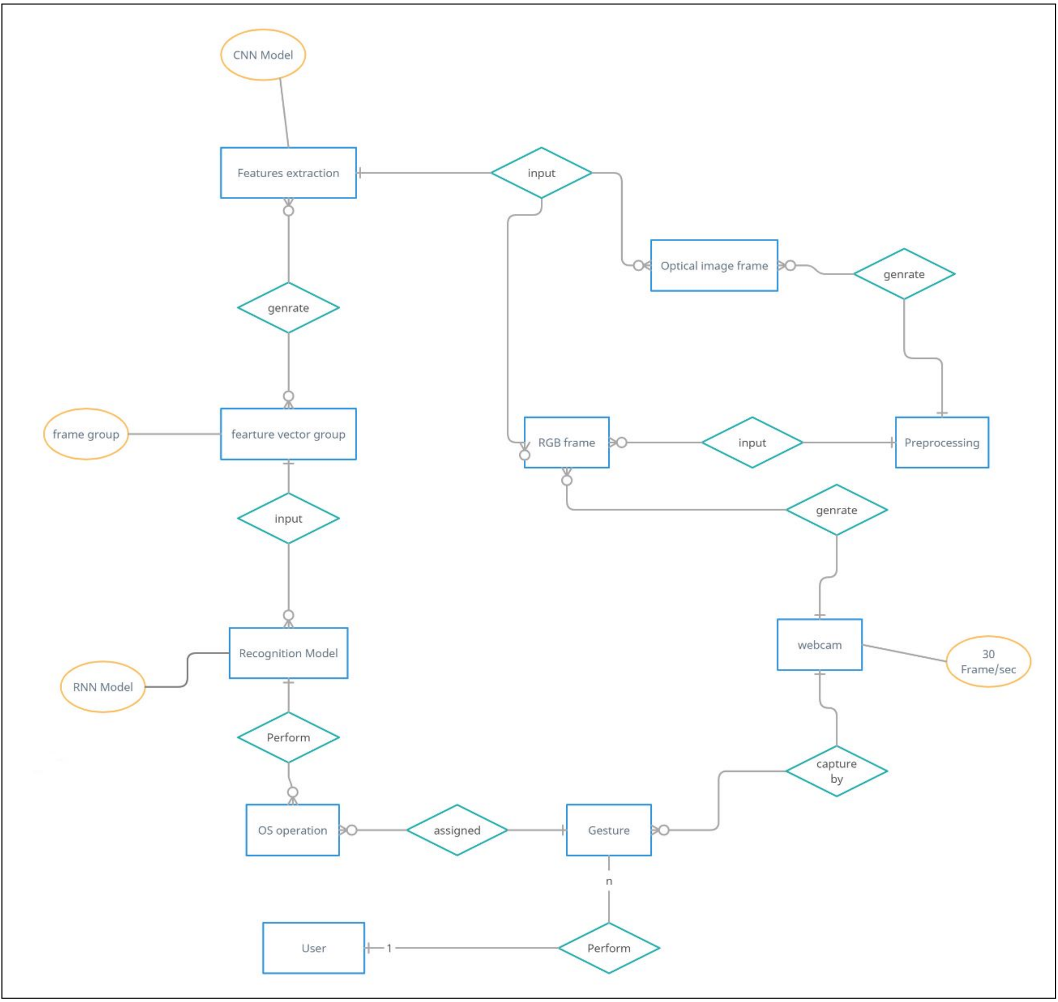

Figure 6.7: Entity Relationship diagram

In figure 6.7 show the relationship between the different entities present int the system. Here user and gesture are related by relation Perform which means users is perorm the gestures. Gesture and webcam are related by relation capture by which mean geste is capture by webcam. Then webcam geneate the RGB image frame. RGB frame is used by image preprocessing model for generating the optical image frame. then both RGB frame and optical image frame are taken as input for CNN model and generate the feature vector group and RNN takes the feature vector group and do the classification and classifies the gesture. the classified gesture is then mapped to corresponding navigation operation.And at last the operation is executed.

6.4 UML Diagrams

6.4.1 Use case Diagram

Use case diagrams are a set of use cases, actors, and their relationships. They represent the use case view of a system.A use case represents a particular functionality of a system. Hence, use case diagram is used to describe the relationships among the functionalities and their internal/external controllers. These controllers are known as actors.

In figure 6.8:

- Actor user: user represent the actual person who will be going to use the software.
- Actor system: represent the ML model.
- use case capture gesture: the use case capture the gesture perform by the user using the webcam.
- user case extract features: the use case extract characteristic of the capture gesture and generate the feature vector.
- use case recognize gesture: the use case recognize the gesture using the feature vector.
- use case display result: the sue case display the actual result to the user.

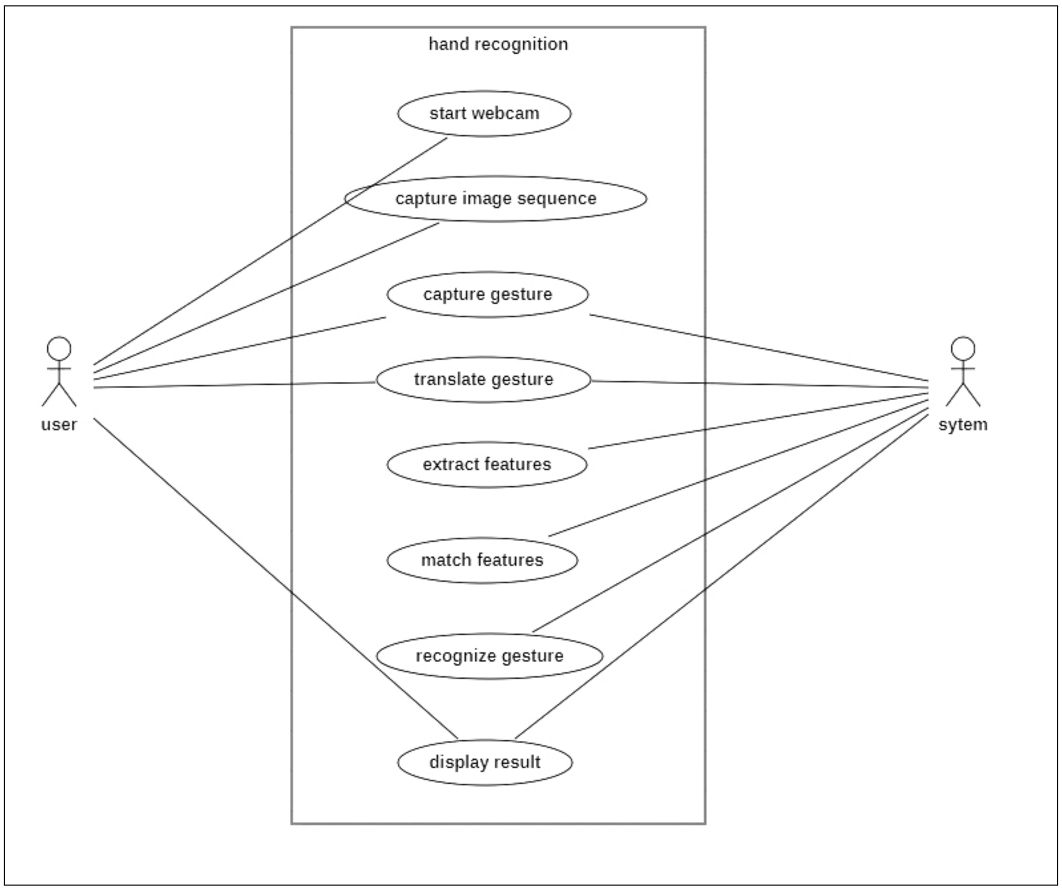

Figure 6.8: Use Case diagram

6.4.2 Class Diagram

Class diagrams are the most common diagrams used in UML. Class diagram consists of classes, interfaces, associations, and collaboration. Class diagrams basically represent the object-oriented view of a system, which is static in nature.

Active class is used in a class diagram to represent the concurrency of the system.

Class diagram represents the object orientation of a system. Hence, it is generally used for development purpose. This is the most widely used diagram at the time of system construction.

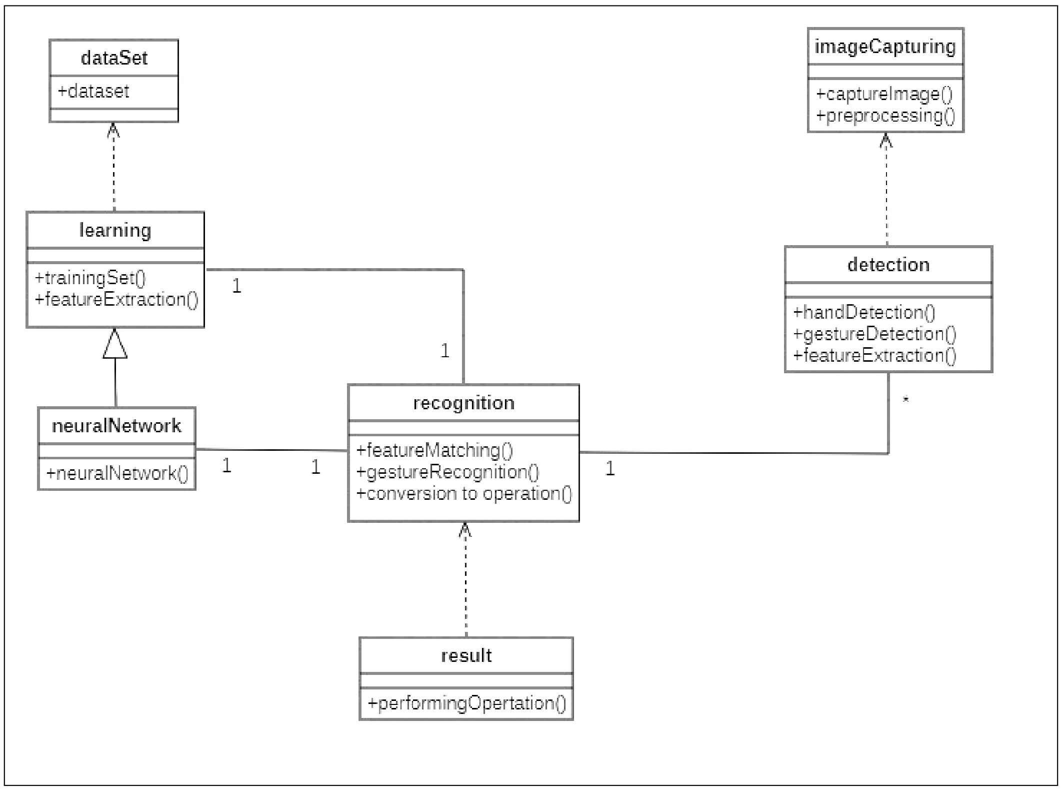

Figure 6.9: Class diagram

6.4.3 State Diagram

Any real-time system is expected to be reacted by some kind of internal/external events. These events are responsible for state change of the system.Statechart diagram is used to represent the event driven state change of a system. It basically describes the state change of a class, interface, etc.State chart diagram is used to visualize the reaction of a system by internal/external factors.

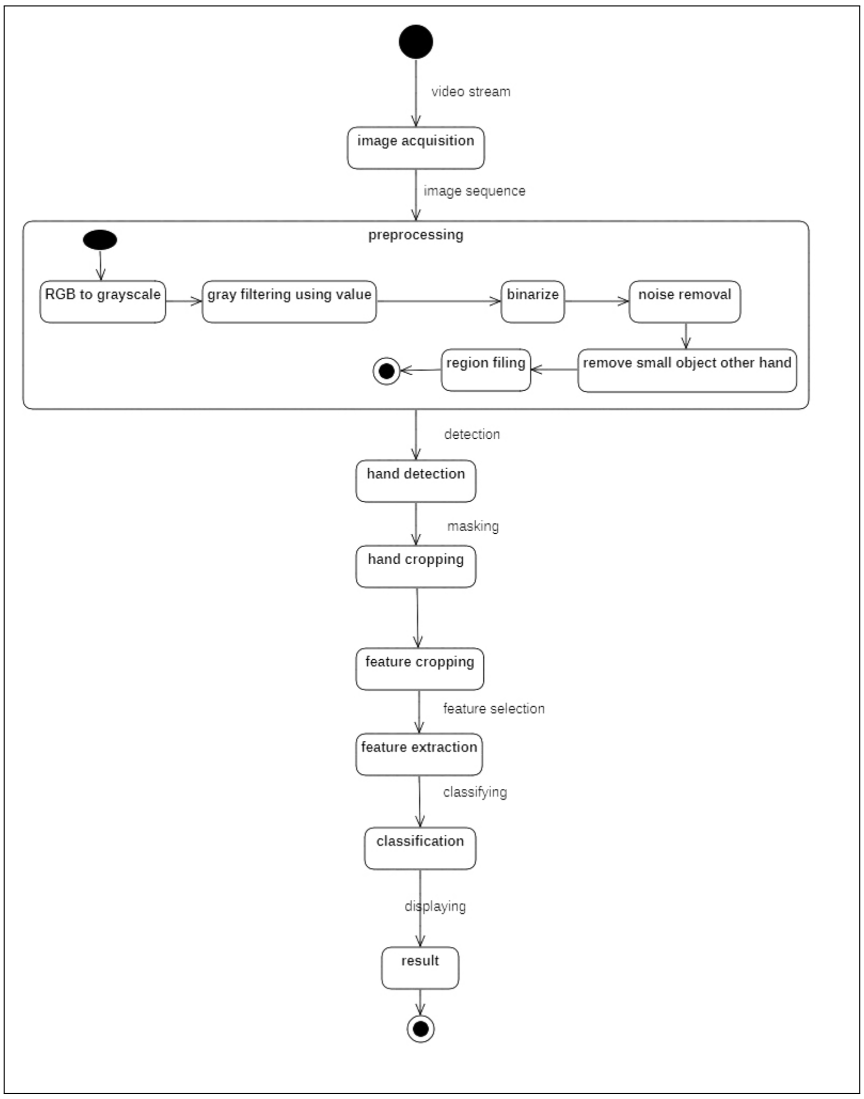

Figure 6.10: State diagram

6.4.4 Activity Diagram

Activity diagram describes the flow of control in a system. It consists of activities and links. The flow can be sequential, concurrent, or branched.Activities are nothing but the functions of a system. Numbers of activity diagrams are prepared to capture the entire flow in a system. Activity diagrams are used to visualize the flow of controls in a system. This is prepared to have an idea of how the system will work when executed.

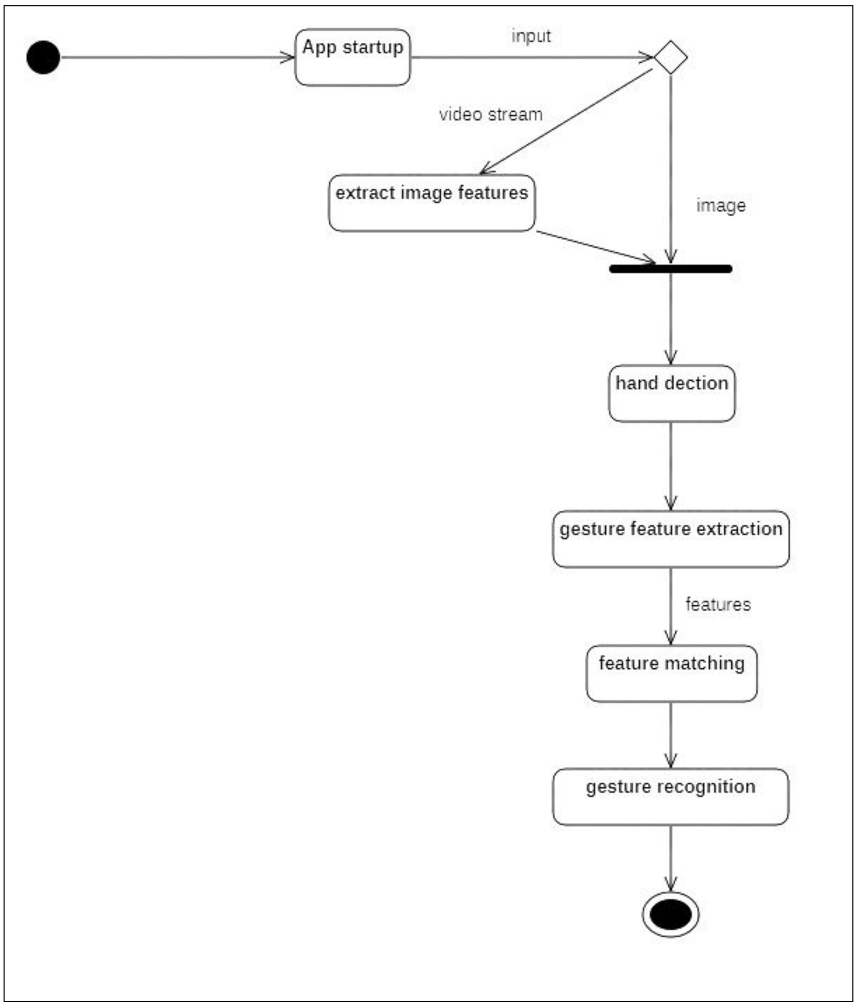

Figure 6.11: Activity diagram

6.4.5 Sequence Diagram

A sequence diagram is an interaction diagram. From the name, it is clear that the diagram deals with some sequences, which are the sequence of messages flowing from one object to another. Interaction among the components of a system is very important from implementation and execution perspective. Sequence diagram is used to visualize the sequence of calls in a system to perform a specific functionality.

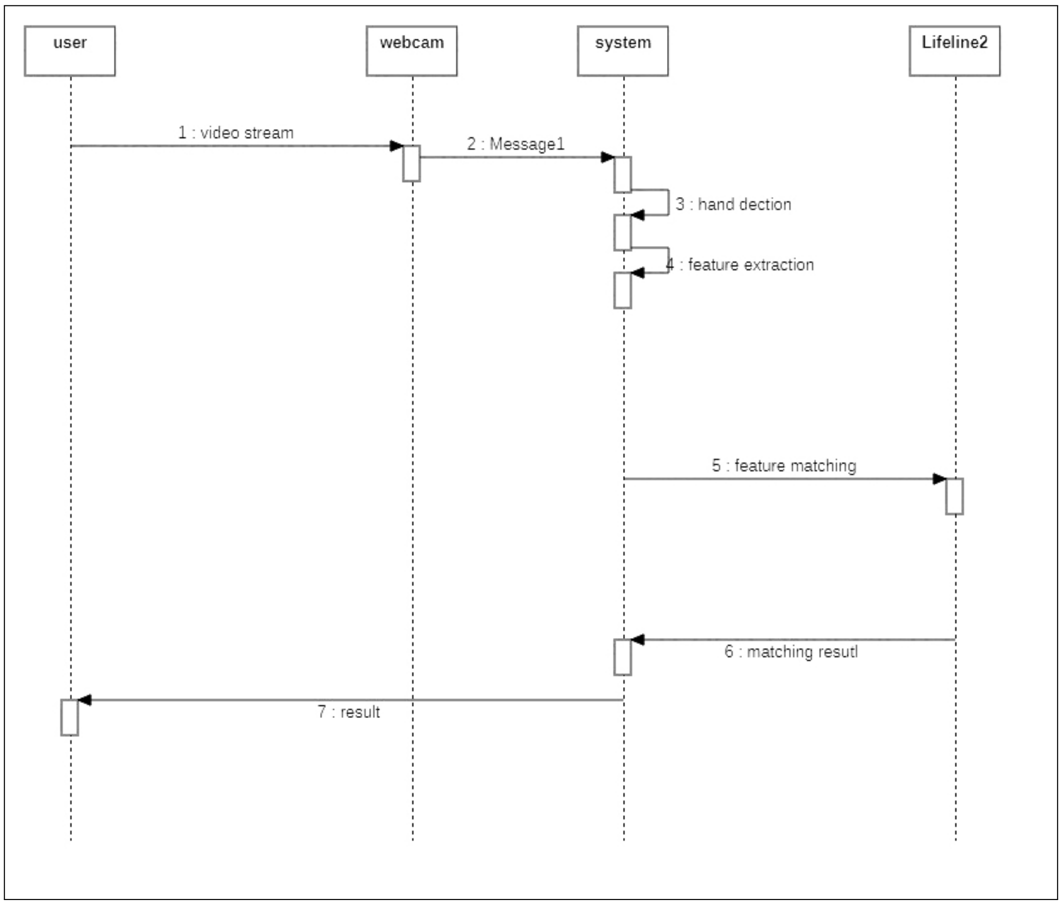

Figure 6.12: Class diagram

6.4.6 Package Diagram

Package diagrams are structural diagrams used to show the organization and arrangement of various model elements in the form of packages. A package is a grouping of related UML elements, such as diagrams, documents, classes, or even other packages. Each element is nested within the package, which is depicted as a file folder within the diagram, then arranged hierarchically within the diagram. Package diagrams are most commonly used to provide a visual organization of the layered architecture within any UML classifier, such as a software system.

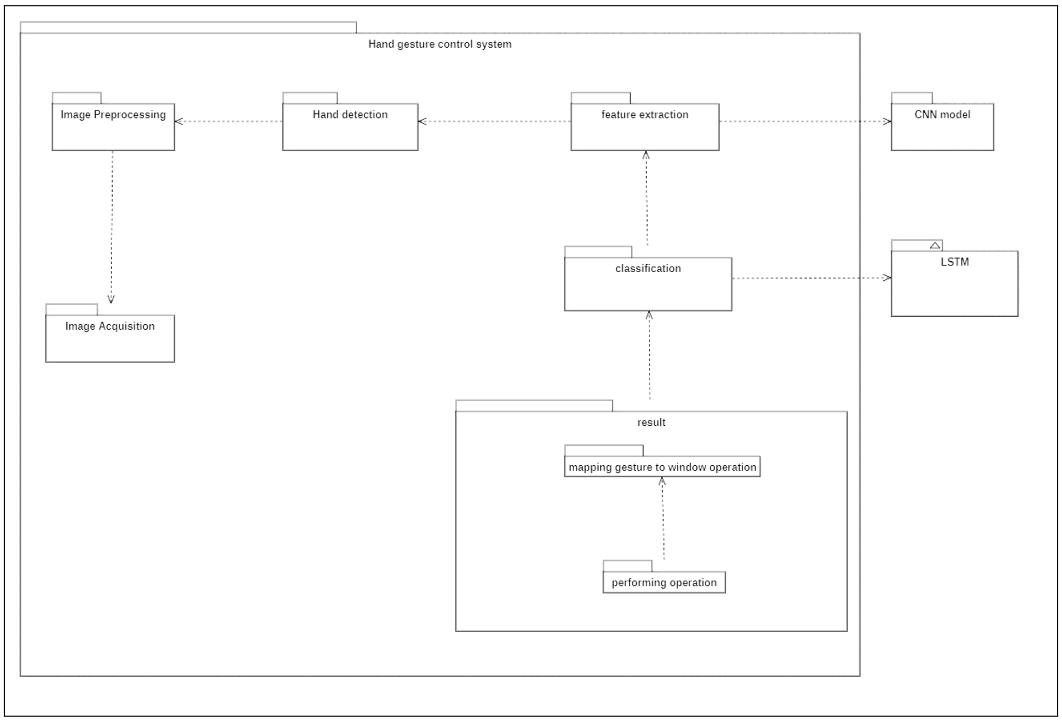

Figure 6.13: Package diagram

6.4.7 Interaction Overview

UML Interaction Overview Diagrams provide a high level of abstraction an interaction model. It is a variant of the Activity Diagram where the nodes are the interactions or interaction occurrences. The Interaction Overview Diagram focuses on the overview of the flow of control of the interactions which can also show the flow of activity between diagrams. In other words, you can link up the "real" diagrams and achieve high degree navigability between diagrams inside an Interaction Overview Diagram.

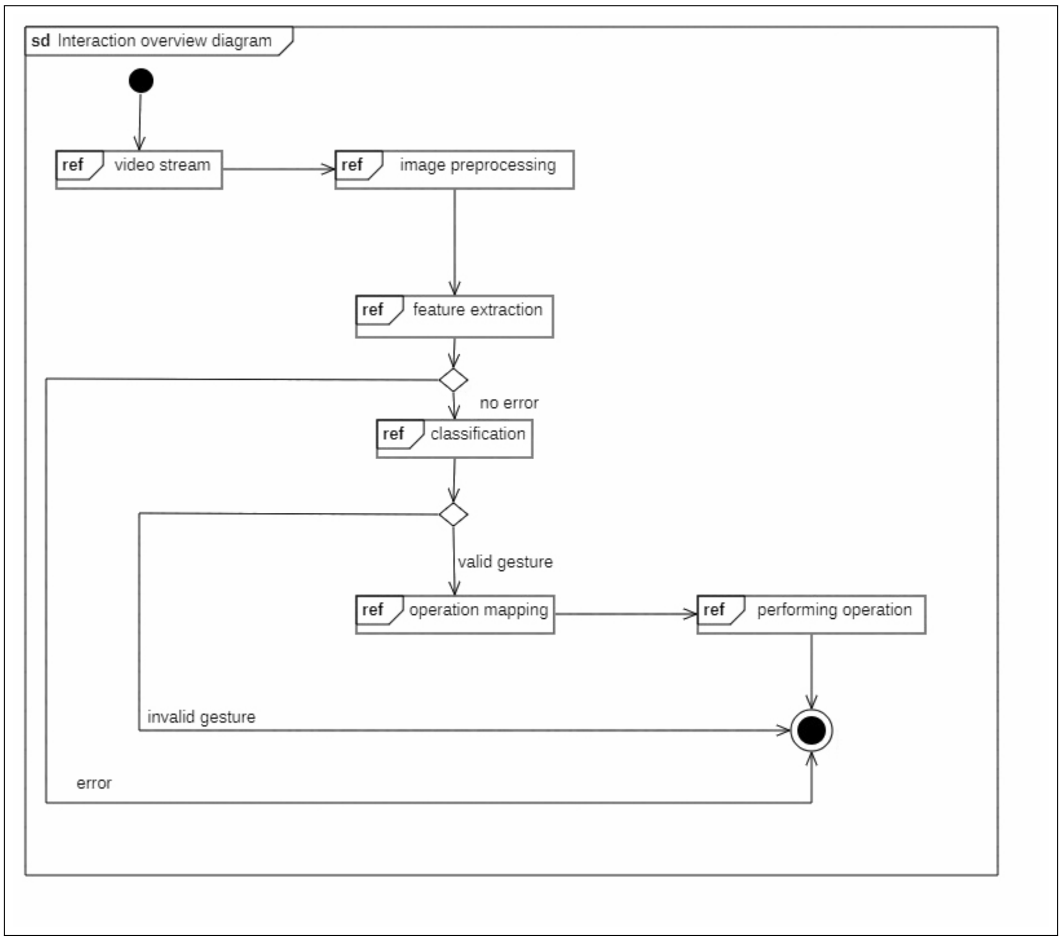

Figure 6.14: Interaction overview diagram

6.4.8 Component Diagram

Component diagrams represent a set of components and their relationships. These components consist of classes, interfaces, or collaborations. Component diagrams represent the implementation view of a system.During the design phase, software artifacts (classes, interfaces, etc.) of a system are arranged in different groups depending upon their relationship. Now, these groups are known as components.Finally, it can be said component diagrams are used to visualize the implementation.

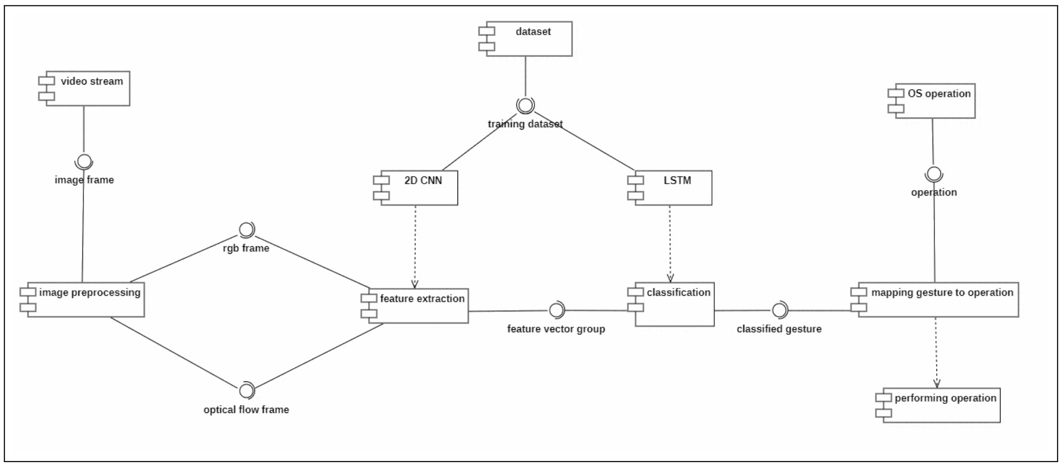

Figure 6.15: Component diagram

6.4.9 Deployment Diagram

Deployment diagrams are a set of nodes and their relationships. These nodes are physical entities where the components are deployed.Deployment diagrams are used for visualizing the deployment view of a system. This is generally used by the deployment team.

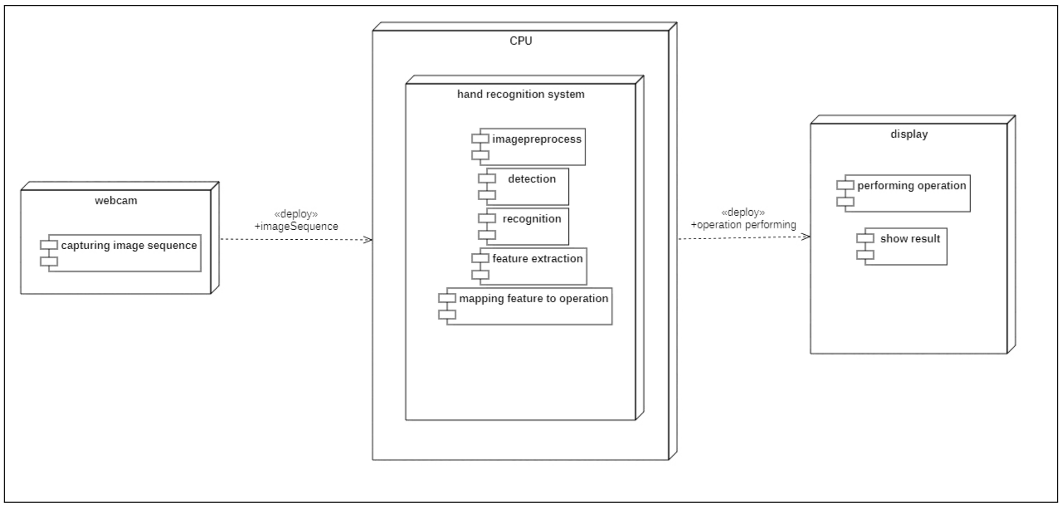

Figure 6.16: Deployment diagram

Chapter 7

OTHERS SPECIFICATION

7.1 Advantages

Through this system we can establish basic system control without actually having to come in contact with the system physically. 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.

7.2 Limitations

. 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. 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. 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.

7.3 Applications

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. 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.

Chapter 8

CONCLUSION AND FUTURE WORKS

The purpose of this research was to identify effective strategies for the contact less navigation system. Based on the analysis we proposed a way to use the hand gesture perform by the user the system will respond to it accordingly. We will be using ML concept like CNN, RNN etc. for the development of the software. We describe all the requirement in SRS section and all the diagram related to the system in system design.

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.

Chapter 9

References

[1] Yuxiao Chen, Long Zhao, Xi Peng, Jianbo Yuan, and Dimitris N Metaxas. Construct dynamic graphs for hand gesture recognition via spatial-temporal attention. *arXiv preprint arXiv:1907.08871*, 2019.

[2] Hung-Yuan Chung, Yao-Liang Chung, and Wei-Feng Tsai. An efficient hand gesture recognition system based on deep cnn. In *2019 IEEE International Conference on Industrial Technology (ICIT)*, pages 853–858. IEEE, 2019.

[3] Yuecong Min, Yanxiao Zhang, Xiujuan Chai, and Xilin Chen. An efficient pointlstm for point clouds based gesture recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5761–5770, 2020.

[4] Md Abdur Rahim, Abu Saleh Musa Miah, Abu Sayeed, and Jungpil Shin. Hand gesture recognition based on optimal segmentation in human-computer interaction. In *2020 3rd IEEE International Conference on Knowledge Innovation and Invention (ICKII)*, pages 163–166. IEEE, 2020.

[5] Debajit Sarma and MK Bhuyan. Hand gesture recognition using deep network through trajectory-to-contour based images. In *2018 15th IEEE India council international conference (INDICON)*, pages 1–6. IEEE, 2018.

[6] Hao Tang, Hong Liu, Wei Xiao, and Nicu Sebe. Fast and robust dynamic hand gesture recognition via key frames extraction and feature fusion. *Neurocomputing*, 331:424–433, 2019.

[7] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks for action recognition in videos. *IEEE transactions on pattern analysis and machine intelligence*, 41(11):2740–2755, 2018.

[8] Jun Xu, Hanchen Wang, Jianrong Zhang, and Linqin Cai. Robust hand gesture recognition based on rgb-d data for natural human-computer interaction. *IEEE Access*, 2022.

[9] Wenjin Zhang and Jiacun Wang. Dynamic hand gesture recognition based on 3d convolutional neural network models. In *2019 IEEE 16th International Conference on Networking, Sensing and Control (ICNSC)*, pages 224–229. IEEE, 2019.

[10] Wenjin Zhang, Jiacun Wang, and Fangping Lan. Dynamic hand gesture recognition based on short-term sampling neural networks. *IEEE/CAA Journal of Automatica Sinica*, 8(1):110–120, 2020.