GESTURE CONTROLLED VIRTUAL MOUSE

a Major Project-1

submitted **during 7th Semester** *for partial fulfillment of the requirements for the award of*

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE & ENGINEERING

by

PANIDAPU HIMA CHANDANA- 19BQ1A05H1

NAVULURI NITISH KUMAR - 19BQ1A05G2

PERUMALLA HEMA SRI - 19BQ1A05H7

NALLAMOTHU HEMANTH - 19BQ1A05F8

Under the guidance of

Dr. K. Lohitha Lakshmi

Associate Professor, Dept of CSE

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

(B.Tech Program is Accredited by NBA)

VASIREDDY VENKATADRI INSTITUTE OF TECHNOLOGY

Permanently Affiliated to JNTU Kakinada, Approved by AICTE

Accredited by NAAC with 'A' Grade, ISO 9001:2008 Certified

NAMBUR(V), PEDAKAKANI(M), GUNTUR-522 508

Tel no: 0863-2118036, url:www.vvitguntur.com

November 2022

DECLARATION

We, PANIDAPU HIMA CHANDANA (19BQ1A05H1), NAVULURI NITISH KUMAR(19BQ1A05G2), PERUMALLA HEMA SRI (19BQ1A05H7), NALLAMOTHU HEMANTH(19BQ1A05F8) hereby declare that the Project Report entitled "GESTURE CONTROLLED VIRTUAL MOUSE" done by us, under the guidance of Dr.K.LOHITHA LAKSHMI ,Associate Professor,Dept of CSE at Vasireddy Venkatadri Institute of Technology is submitted **during 7th Semester**, for partial fulfillment of the requirements for the award of Bachelor of Technology in Computer Science & Engineering. The results embodied in this report have not been submitted **in any semester anywhere else for credits in this or other University.**

DATE :

PLACE :

SIGNATURE OF THE CANDIDATE(S)

(P.Hima Chandana-19BQ1A05H1)

(Navuluri Nitish Kumar-19BQ1A05G2)

(Perumalla Hema Sri-19BQ1A05H7)

(Nallamothu Hemanth-19BQ1A05F8)

VASIREDDY VENKATADRI INSTITUTE OF TECHNOLOGY
Permanently Affiliated to JNTU Kakinada, Approved by AICTE
Accredited by NAAC with 'A' Grade, ISO 9001:2008 Certified
Nambur, Pedakakani (M), Guntur (Dt) - 522508
DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
B.Tech Program is Accredited by NBA

CERTIFICATE

This is to certify that this Major Project-1 Report is the bonafide work of PANIDAPU HIMA CHANDANA,NAVULURI NITISH KUMAR,PERUMALLA HEMA SRI,NALLAMOTHU HEMANTH bearing **Reg. No. 19BQ1A05H1, 19BQ1A05G2, 19BQ1A05H7, 19BQ1A05F8** who had carried out the project entitled "GESTURE CONTROLLED VIRTUAL MOUSE" **under our supervision, during 7th semester.**

Project Guide **Head of the Department**

________________ ________________

Submitted for Viva voce Examination held on ____________________________

External Examiner

ACKNOWLEDGEMENT

I take this opportunity to express my deepest gratitude and appreciation to all those people who made this project work easier with words of encouragement, motivation, discipline, and faith by offering different places to look to expand my ideas and helped me towards the successful completion of this project work.

First and foremost, I express my deep gratitude to **Mr. Vasireddy Vidya Sagar**, Chairman, Vasireddy Venkatadri Institute of Technology for providing necessary facilities throughout the B.Tech programme.

I express my sincere thanks to **Dr. Y. Mallikarjuna Reddy**, Principal, Vasireddy Venkatadri Institute of Technology for his constant support and cooperation throughout the B.Tech programme.

I express my sincere gratitude to **Dr. V. Ramachandran**, Professor & HOD, Computer Science & Engineering, Vasireddy Venkatadri Institute of Technology for his constant encouragement, motivation and faith by offering different places to look to expand my ideas.

I would like to express my sincere gratefulness to my guide **Dr. K. Lohitha Lakshmi,Associate Professor(CSE Dept.)** for her insightful advice, motivating suggestions, invaluable guidance, help and support in successful completion of this project.

I would like to take this opportunity to express my thanks to the **teaching and non- teaching** staff in Department of Computer Science & Engineering, VVIT for their invaluable help and support.

Panidapu Hima Chandana(19BQ1A05H1)

Navuluri Nitish Kumar(19BQ1A05G2)

Perumalla Hema Sri(19BQ1A05H7)

Nallamothu Hemanth (19BQ1A05F8)

INDEX
TABLE OF CONTENTS

List of Figures

ABSTRACT

Gesture Controlled Virtual mouse makes human-computer interaction simple by making use of dragging and gestures. The computer requires almost no contact. All the I/O operations can be virtually controlled with a voice assistant. this project proposes a way to control the position of the cursor with the bare hands without any electronic device. While the operations like clicking and dragging objects will be performed with different hand gestures. The proposed system requires a webcam as an input device. This project makes use ofstate-of-art Machine Learning and Computer Vision algorithms to recognize hand gestures and voice commands, which works smoothly without any hardware requirements. It was implemented by the Media Pipe Python framework which grants access to the core components such as Timestamp, Packet, and CalculatorGraph. The system is used to control the pointer of the mouse by detecting the human hand and moving the pointer in the direction of the human hand respectively. The output of the camera will be displayed on the system's screen so that it can be further calibrated by the user. So this makes a user perform actions quickly and complete the work in time.

CHAPTER 1
INTRODUCTION

Non - verbal communication in the form of gestures is utilized to convey a certain message. The movements of a person's body, hands, or face can be used to send this message. Gestures have the ability to convey something when engaging with other individuals. From simple to incredibly complicated hand movements. For example, we can point to something (an object or people) or utilize a variety of simple gestures or motions that are conveyed in sign language that are integrated with their syntax and dictionary, more often known as sign languages. As a result, humans can communicate more effectively by employing hand motions as a device with the help of computers.Hand gestures have taken control of mouse functions such as controlling the movement of a visual item. The work is supposed to be low-cost, and it makes use of low-cost input devices such as a webcam to capture hand movements as input. Modelling predetermined command-based movements is used to manipulate materials. The conventional human-machine interaction has evolved as a result of virtualization and the longterm shift towards immersive technologies like the metaverse.With the advancement of technology[4], devices are becoming smaller and smaller.Some devices have gone wireless, while others have gone unnoticed. The proposal is to create a virtual mouse that recognizes gestures. The goal is to use a simple camera instead of a classic or regular mouse to control mouse cursor functions. The Virtual Mouse works as a medium of the user and the machine only using a camera.

It helps the user to interact with a machine without any mechanical or physical devices and control mouse functions. In this gesture recognition system, it is very possible to capture & track the fingertip of hand with a webcam or built-in cam and the system track the color and movement of the hand & move cursor with it. The Hand Gesture recognition is moving at tremendous speed for the futuristic products and services and major companies are developing a technology based on the hand gesture system for applications such as:

1. Laptop, Hand held devices,
2. Professional and LED lights.
3. Entertainment
4. Education

The initial setup includes a camera that can be used for providing input to the system. To the extreme, it can also be called as hardware because it uses a camera for tracking hands. The objectives are to accomplish:

1. Replacing cumbersome touchpads.
2. Malfunctioning hardware mouse
3. Reduce cost of hardware.

The work focuses on extracting the features over the human hands and then matching their features to recognize the movement of the hand.

To track fingertips as a movable object, and to utilize it for mouse functions, the camera should be positioned in a way so that it can see the user's hands in the right positions. This can be use space saving situations, for those patients who don't have control over their limbs and for other similar cases. It's a virtual mouse instead of a physical mouse which will work only based on webcam captured frames & tracking colored fingertips. Most of the computer users use a webcam on their computer and most of the laptops have a built-in webcam. The proposed system which is webcam based, might be able to eliminate the need of a mouse partially.

The process of interaction with a computer using hand gesture is a very interesting & effective approach to HCI (Human-Computer Interaction)[2]. There is some really good research on this interest. The hand gesture recognition technology is also popular in sign language recognition. The objective is to develop and implement an alternative system to control a mouse cursor. The alternative method is hand gesture recognition using webcam and cnn model.

CHAPTER 2
AIM & SCOPE

2.1 EXISTING SYSTEM:

The existing system consists of the generic mouse and trackpad system of monitor controlling and the non-availability of a hand gesture system. The remote accessing of monitor screen using the hand gesture is unavailable. Even-though it is largely trying to implement the scope is simply restricted in the field of virtual mouse. The existing virtual mouse control system consists of the simple mouse operations using the hand recognition system, where we could perform the basic mouse operation like mouse pointer control, left click, right click, drag etc. The further use of the hand recognition is not being made use of. Even-though there are a number of systems which are used for hand recognition, the system they made used is the static hand recognition which is simply recognition of the shape made by hand and by defining an action for each shape made, which is limited to a number of defined actions and a large amount of confusion.

2.2 PROPOSED SYSTEM:

Using the current system even-though there are a number of quick access methods available for the hand and mouse gesture for the laptops, using our project we could make use of the laptop or webcam and by recognizing the hand gesture we could control mouse and perform basic operations like mouse pointer controlling, select and deselect using left click, drag and drop functions etc,. The project done is a “Zero Cost” hand recognition system for laptops, which uses simple algorithms to determine the hand, hand movements and by assigning an action for each movement. The system we are implementing which is been written in python code be much more responsive and is easily implemented since python is a simple language and is platform independent with a flexibility and is portable which is desirable in creating a program which is focused in such an aim for creating a Virtual Mouse and Hand Recognition system. The system be much more extendable by defining actions for the hand movement for doing a specific action. It could be further modified to any further extent by implementing such actions for the set of hand gesture.

2.3 FEASIBILITY STUDY:

An important outcome of preliminary investigation is the determination that the system request is feasible.This is possible only if it is feasible within limited resources and time.
The different feasibilities that have to be analyzed are:

- Technical Feasibility
- Operational Feasibility
- Economic Feasibility

2.3.1. Technical Feasibility:

The system is platform dependent and thus requires minimal software to be pre installed on the machine. Python and Anaconda environment are used to develop the system. Media Pipe Hands utilizes an ML pipeline consisting of multiple models working together. A palm detection model that operates on the full image and returns an oriented hand bounding box. A hand landmark model that operates on the cropped image region defined by the palm detector and returns high-fidelity 3D hand key points .

2.3.2. Operational Feasibility:

Operational Feasibility deals with the study of prospects of the system to be developed. This system operationally helps the users to work efficiently and produces faster outcome of actions.

2.3.3. Economic Feasibility:

Economic Feasibility or Cost-benefit is an assessment of the economic justification for a computer based project. As hardware was required for the project is minimal for all the purposes thus the cost on project's hardware is low. Every dependency required for the project can be installed from PyPI() for free of cost and hence this reduces further cost of the project.

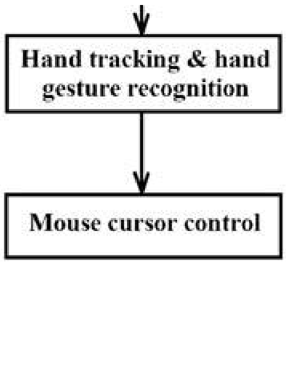
2.4 SCOPE OF ANALYSIS

Virtual Mouse that will soon to be introduced to replace the physical computer[3] mouse to promote convenience while still able to accurately interact and control the computer system. To do that, the software requires to be fast enough to capture and process every image, in order to successfully track the user's gesture. Therefore, this project will develop a software application with the aid of the latest software coding technique and the open-source computer vision library also known as the OpenCV.

The scope of the project is as below:

- Real time application.
- User friendly application.
- Removes the requirement of having a physical mouse.

The process of the application can be started when the user's gesture was captured in real time by the webcam, which the captured image will be processed for segmentation to identify which pixels values. The AI virtual mouse system is useful for many applications; it can be used to reduce the space for using the physical mouse, and it can be used in situations where we cannot use the physical mouse. The system eliminates the usage of devices, and it improves the human-computer interaction.



CHAPTER 3

MACHINE LEARNING

Machine learning (ML) is the study of computer algorithms that improve automatically through experience. It is seen as a subset of artificial intelligence. Machine learning algorithms build a mathematical model based on sample data, known as "training data", in order to make predictions or decisions without being explicitly programmed to do so. Machine learning algorithms are used in a wide variety of applications, such as email filtering and computer vision, where it is difficult or infeasible to develop conventional algorithms to perform the needed tasks.

Machine learning is closely related to computational statistics, which focuses on making predictions using computers. The study of mathematical optimization delivers methods, theory and application domains to the field of machine learning. Data mining is a related field of study, focusing on exploratory data analysis through unsupervised learning. In its application across business problems, machine learning is also referred to as predictive analytics.

3.1 OVERVIEW:

Machine learning involves computers discovering how they can perform tasks without being explicitly programmed to do so. For the early tasks that humans wanted computers to accomplish, it was possible to create algorithms that enabled the machine to execute all the steps needed to solve the problem in hand. So on the computer's part, no learning was needed. For certain advanced tasks, facial recognition[8] for example, it is not easy to create the needed algorithms, partly as it's not easy for humans to precisely define how we recognize faces. Abundant face related data exists however. So far, compared to the difficulty in directly creating the required algorithms, it's turned out in practice to be easier to assist computers to teach themselves how to recognize faces from available data. The discipline of machine learning develops various approaches for computers to learn to accomplish tasks for which no algorithm exists.

3.2 MACHINE LEARNING APPROACHES:

Early classifications for machine learning approaches sometimes divided them into three broad categories, depending on the nature of the "signal" or "feedback" available to the learning system.

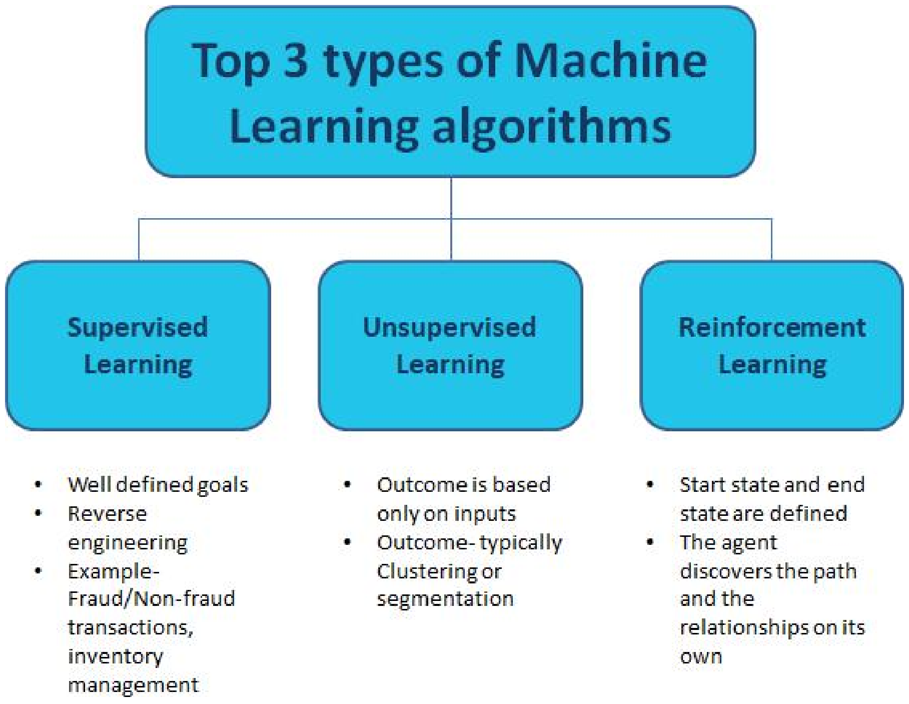

Top 3 types of Machine Learning algorithms
Supervised Learning
Unsupervised Learning
Reinforcement Learning
• Well defined goals
• Reverse engineering
• Example- Fraud/Non-fraud transactions, inventory management
• Outcome is based only on inputs
• Outcome- typically Clustering or segmentation
• Start state and end state are defined
• The agent discovers the path and the relationships on its own

Fig3.1 Types of Machine Learning Algorithms

3.2.1 ***Supervised learning:*** Supervised learning algorithms build a mathematical model of a set of data that contains both the inputs and the desired outputs. The data is known as training data, and consists of a set of training examples. Each training example has one or more inputs and the desired output, also outputs known as a supervisory signal. In the mathematical model, each training example is represented by an array or vector, sometimes called a feature vector, and the training data is represented by a matrix.

3.2.2 ***Unsupervised learning:*** Unsupervised learning algorithms take a set of datathat contains only inputs, and find structure in the data, like grouping or clustering of data points. The algorithms, therefore, learn from test data that has not been labeled, classified or categorized. Instead of responding to feedback, unsupervised learning algorithms identify commonalities in the data and react based on the presence or absence of such commonalities in each new piece of data. A central application of unsupervised learning is in the field of density estimation in statistics, such as finding the probability density function. Though unsupervised learning encompasses other domains involving summarizing and explaining data features.

3.2.3 ***Semi-Supervised Learning:*** Semi-supervised learning falls between unsupervised learning (without any labeled training data) and supervised learning (with completely labeled training data). Some of the training examples are missing training labels, yet many machine-learning researchers have found that unlabeled data, when used in conjunction with a small amount of labeled data, can produce a considerable improvement in learning accuracy. In weakly supervised learning, the training labels are noisy, limited, or imprecise; however,these labels are often cheaper to obtain, resulting in larger effective training sets.

3.2.4 ***Reinforcement learning:*** Reinforcement learning is an area of machine learning concerned with how software agents ought to take actions in an environment so as to maximize some notion of cumulative reward. Due to its generality, the field is studied in many other disciplines, such as game theory, control theory, operations research, information theory, simulationbased optimization, multi-agent systems, swarm intelligence, statistics and genetic algorithms. In machine learning, the environment is typically represented as a Markov Decision Process (MDP). Many reinforcement learning algorithms use dynamic programming techniques. Reinforcement learning algorithms do not assume knowledge of an exact mathematical model of the MDP, and are used when exact models are infeasible.

Reinforcement learning algorithms are used in autonomous vehicles or in learning to play a game against a human opponent.

3.2.5 ***Other Learning:***

- Feature Learning
- Sparse Dictionary Learning
- Artificial Neural Networks
- Decision Trees
- Support Vector Machines
- Regression Analysis
- Genetic Algorithms
- Training models
- Federated Learning

3.3 DEEP LEARNING VS MACHINE LEARNING VS ARTIFICIAL INTELLIGENCE:

AI involves machines that can perform tasks that are characteristic of human intelligence. While this is rather general, it includes things like planning, understanding language, recognizing objects and sounds, learning, and problem solving.

We can put AI in two categories, general and narrow. General AI would have all of the characteristics of human intelligence, including the capacities mentioned above. Narrow AI exhibits some facet(s) of human intelligence, and can do that facet extremely well, but is lacking in other areas. A machine that's great at recognizing images, but nothing else, would be an example of narrow AI.

At its core, machine learning is simply a way of achieving AI.

Arthur Samuel coined the phrase not too long after AI, in 1959, defining it as, "the ability to learn without being explicitly programmed." You see, you can get AI without using machine learning, but this would require building millions of lines of codes with complex rules and decision-trees.

So instead of hard coding software routines with specific instructions to accomplish a particular task, machine learning is a way of "training" an algorithm so that it can learn how. "Training" involves feeding huge amounts of data to the algorithm and allowing the algorithm to adjust itself and improve.

To give an example, machine learning has been used to make drastic improvements to computer vision (the ability of a machine to recognize an object in an image or video). You gather hundreds of thousands or even millions of pictures and then have humans tag them. For example, the humans might tag pictures that have a cat in them versus those that do not. Then, the algorithm tries to build a model that can accurately tag a picture as containing a cat or not as well as a human.

Once the accuracy level is high enough, the machine has now "learned" what a cat looks like. Deep learning is one of many approaches to machine learning.

Other approaches include decision tree learning, inductive logic programming, clustering, reinforcement learning, and Bayesian networks, among others.

Deep learning was inspired by the structure and function of the brain, namely the Inter connecting of many neurons. Artificial Neural Networks (ANNs) are algorithms that mimic the biological structure of the brain.

In ANNs, there are “neurons” which have discrete layers and connections to other “neurons”.
Each layer picks out a specific feature to learn, such as curves/edges in image recognition. It’s this layering that gives deep learning its name, depth is created by using multiple layers as opposed to a single layer.

Machine learning and deep learning are both types of AI. In short, **machine learning** is AI that can automatically adapt with minimal human interference. **Deep learning** is a subset of machine learning that uses artificial neural networks to mimic the learning process of the human brain.

CHAPTER 4
LITERATURE SURVEY

OBJECTIVES

Gesture control is the ability to recognize and interpret movements of the human body in order to interact with and control a computer system without direct physical contact. The term “natural user interface” is becoming commonly used to describe these interface systems, reflecting the general lack of any intermediate devices between the user and the system. The main objective of the AI virtual mouse system is to control the mouse cursor functions by using hand gestures[5] instead of using a physical mouse. The proposed system can be achieved by using a webcam or a built-in camera that detects the hand gestures and hand tip processes these frames to perform the particular mouse functions.

BACKGROUND

The virtual mouse system is useful for many applications. it can be used to reduce the space for using the physical mouse, and it can be used in situations where we cannot use the physical mouse. The system eliminates the usage of devices, and it improves the human-computer interaction. There are generally two approaches for hand gesture recognition, which are hardware based, where the user must wear a tool , and therefore the other is vision based which uses image processing techniques with inputs from a camera. The proposed system is vision based, which uses image processing techniques and inputs from a computer webcam. Vision based gesture recognition tracking and gesture recognition. Hand tracking would be wont to navigate the pc cursor and hand gestures would be wont to perform mouse functions. The scope of the project would therefore be to design a vision based CC system, which can perform the mouse function previously stated.

CONVOLUTIONAL NEURAL NETWORK (CNN)

Convolutional Neural Network (CNN) is a discriminative (supervised) learning data labeled to classify different patterns. CNN improves the connections between DNN layers. CNNs train multiple layers with nonlinear, fully connected networks. The hidden layers of a CNN consist of complex layers that convolve with multiplication or other product. The input CNN requires is numeric. Furthermore, CNN is used to extract dealing with more complex features to perform the task with better accuracy.

CHAPTER 5
DEEP LEARNING

5.1 INTRODUCTION

Deep learning drives many artificial intelligence (AI) applications and services that improve automation, performing analytical and physical tasks without human intervention[7]. Deep learning technology lies behind everyday[6] products and services (such as digital assistants, voice-enabled TV remotes, and credit card fraud detection) as well as emerging technologies (such as self-driving cars)

Deep learning is a set of algorithms in machine learning that attempt to model high-level abstractions in data by using architectures composed of multiple non-linear transformations. Since 2006, deep structured learning, or more commonly called deep learning or hierarchical learning, has emerged as a new area of machine learning research. During the past several years, the techniques developed from deep learning research have already been impacting a wide range of signal and information processing work within the traditional and the new, widened scopes including key aspects of machine learning and artificial intelligence.

Artificial intelligence (AI) is the intelligence exhibited by machines or software, and the branch that develops machines and software with human-like intelligence. The goal of AI is to invent a machine which can sense, remember, learn, and recognize like a real human being.

Perceptron is the first machine which can sense and learn but has fundamentally limited learning abilities. The later neural network with multiple hidden layers can learn more complicated functions but it lacks a good learning algorithm. The appearance of SVM enlightens people within a short time since it facilitates the learning procedures and performs well in many practical problems, but SVM also encounters its bottlenecks due to its shallow architectures.

Deep learning is a learning method with the deep architecture and the good learning algorithms, which can perform the intellectual learning like learning the features. This ability, together with the efficient learning algorithms that can ensure this ability, point out a new direction toward. The development of machine learning is an integral part of the development of artificial intelligence.

In the early days of AI, people were interested in building machines that mimic human brains. The perceptron model was invented in 1957, and it generated over optimistic view for AI during 1960s. After Marvin Minsky pointed out the limitation of this model in expressing complex functions, researchers stopped pursuing this model for the next decade.

In 1970s, the machine learning field was dormant, when expert systems became the mainstream approach in AI. The revival of machine learning came in mid-1980s, when the decision tree model was invented and distributed as software. The model can be viewed by a 16human and is easy to explain. It is also very versatile and can adapt to widely different problems. It is also in mid 1980s multi-layer neural networks were invented, with enough hidden layers; a neural network can express any function, thus overcoming the limitation of perceptron. Deep learning is one of many approaches to machine learning. Other approaches include decision tree learning, inductive logic programming, clustering, reinforcement learning, and Bayesian networks, among others.

Deep learning was inspired by the structure and function of the brain, namely the interconnecting of many neurons. Artificial Neural Networks (ANNs) are algorithms that mimic the biological structure of the brain.

In ANNs, there are “neurons” which have discrete layers and connections to other “neurons”. Each layer picks out a specific feature to learn, such as curves/edges in image recognition. It’s this layering that gives deep learning its name, depth is created by using multiple layers as opposed to a single layer.

A series of workshops, tutorials, and special issues or conference special sessions in recent years have been devoted exclusively to deep learning and its applications to various signal and information processing areas. These include:

1. 2008 NIPS Deep Learning Workshop;
2. 2009 NIPS Workshop on Deep Learning for Speech Recognition and Related Applications;
3. 2009 ICML Workshop on Learning Feature Hierarchies;
4. 2011 ICML Workshop on Learning Architectures, Representations, and Optimization for Speech and Visual Information Processing;
5. 2012 ICASSP Tutorial on Deep Learning for Signal and Information Processing;
6. 2012 ICML Workshop on Representation Learning;
7. 2012 Special Section on Deep Learning for Speech and Language Processing in IEEE Transactions on Audio, Speech, and Language Processing (T-ASLP, January);

8. 2010, 2011, and 2012 NIPS Workshops on Deep Learning and Unsupervised Feature Learning;
9. 2013 NIPS Workshops on Deep Learning and on Output Representation Learning. 1710. 2013 Special Issue on Learning Deep Architectures in IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI, September).
11. 2013 International Conference on Learning Representations;
12. 2013 ICML Workshop on Representation Learning Challenges;
13. 2013 ICML Workshop on Deep Learning for Audio, Speech, and Language Processing;
14. 2013 ICASSP Special Session on New Types of Deep Neural Network Learning for Speech Recognition and Related Applications.

5.2 DEEP LEARNING-MODELS

Deep learning models are widely used in extracting high-level abstract features, providing improved performance over the traditional models, increasing interpretability and also for understanding and processing biological data.

CNN MODEL

- A convolutional neural network, or CNN, is a deep learning neural network sketched for processing structured arrays of data such as portrayals.

- CNN are very satisfactory at picking up on design in the input image, such as lines, gradients, circles, or even eyes and faces. This characteristic that makes a convolutional neural networks so robust for computer vision.

- CNN can run directly on a underdone image and do not need any pre-processing.

- A convolutional neural network is a feed forward neural network, seldom with up to 20.

- The strength of a convolutional neural network comes from a particular kind of layer called the convolutional layer.

- CNN contains many convolutional layers assembled on top of each other, each one competent of recognizing more sophisticated shapes.

- With three or four convolutional layers it is viable to recognize handwritten digits and with 25 layers it is possible to differentiate human faces.

- The agenda for this sphere is to activate machines to view the world as humans do, perceive it in a alike fashion and even use the knowledge for a multitude of duty such as image and video recognition, image inspection and classification, media recreation, recommendation systems, natural language processing, etc.

5.3 ALGORTIHMS OF DEEP LEARNING:

5.3.1 CONVOLUTIONAL NEURAL NETWORKS

CNN's, also known as ConvNets, consist of multiple layers and are mainly used for image processing and object detection. Yann LeCun developed the first CNN in 1988 when it was called LeNet. It was used for recognizing characters like ZIP codes and digits. CNN's are widely used to identify satellite images, process medical images, forecast time
series, and detect anomalies.

CNN's have multiple layers that process and extract features from data:

Convolution Layer

CNN has a convolution layer that has several filters to perform the convolution operation.

Rectified Linear Unit (ReLU)

CNN's have a ReLU layer to perform operations on elements. The output is a rectified feature map.

Pooling Layer

The rectified feature map next feeds into a pooling layer. Pooling is a down-sampling operation that reduces the dimensions of the feature map. The pooling layer then converts the resulting two-dimensional arrays from the pooled feature map into a single, long, continuous, linear vector by flattening it.

Fully Connected Layer

A fully connected layer forms when the flattened matrix from the pooling layer is fed as an input, which classifies and identifies the images.

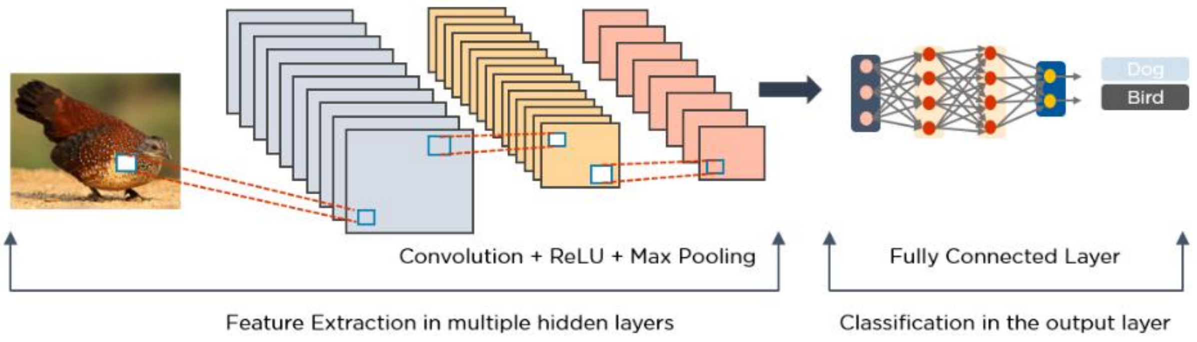

Dog
Bird
Convolution + ReLU + Max Pooling
Fully Connected Layer
Feature Extraction in multiple hidden layers
Classification in the output layer

Fig 5.3.1 Fully Connected Convolution Layer

5.3.2 MULTILAYER PERCEPTRONS(MLPS)

MLPs are an excellent place to start learning about deep learning technology.

MLPs belong to the class of feedforward neural networks with multiple layers of perceptrons that have activation functions. MLPs consist of an input layer and an output layer that are fully connected. They have the same number of input and output layers but may have multiple hidden layers and can be used to build speech-recognition, image-recognition, and machine-translation software.

MLPs feed the data to the input layer of the network. The layers of neurons connect in a graph so that the signal passes in one direction.

MLPs compute the input with the weights that exist between the input layer and the hidden layers.

MLPs use activation functions to determine which nodes to fire. Activation functions include ReLUs, sigmoid functions, and tanh.MLPs train the model to understand the correlation and learn the dependencies between the independent and the target variables from a training data set.

CHAPTER 6
ALGORTIHMS AND FLOWCHARTS

6.1 MACHINE LEARNING MODEL

A machine learning model is a file that has been trained to recognise specific types of patterns. A model is trained over a set of data, providing it with an algorithm that it can use to reason about and learn from that data.

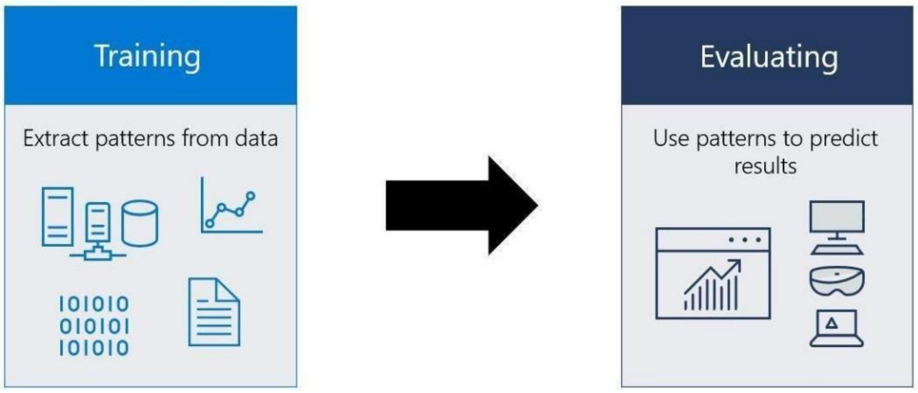


Fig 6.1 Overview of Machine Learning Process

Once trained, the model can be used to reason over new data and make predictions about it. For example., assume we want to create an application that can detect a user's emotions based on their facial expressions. A model can be trained by feeding it images of faces that have each been labeled with a different emotion, and then that model can be used in an application that can recognize any user's emotion.

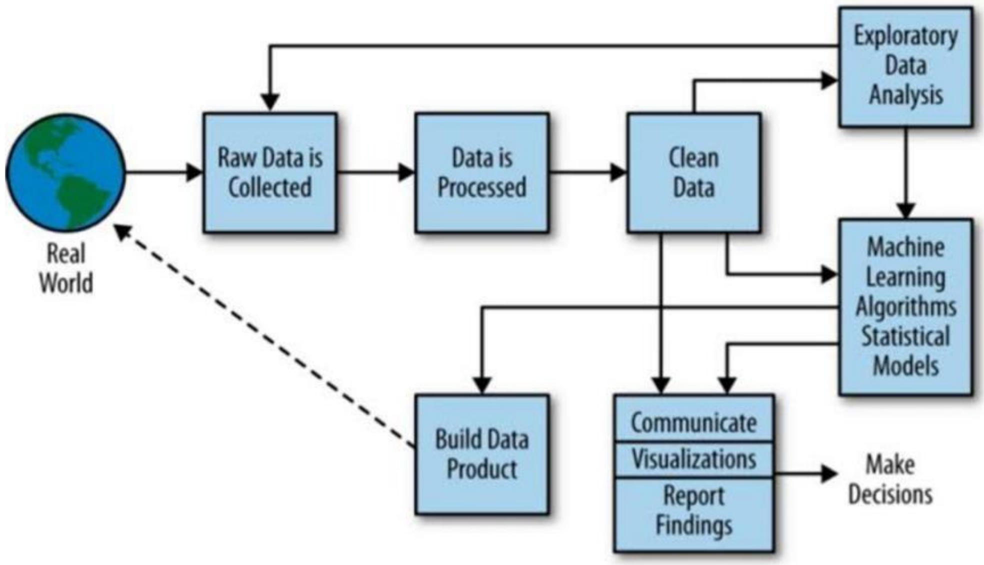

Real World
Raw Data is Collected
Data is Processed
Clean Data
Exploratory Data Analysis
Machine Learning Algorithms Statistical Models
Build Data Product
Communicate Visualizations Report Findings
Make Decisions

Fig 6.2 Machine learning process Flow

When to use Machine Learning?

Good machine learning scenarios often have the following common properties:

1. They involve a repeated decision or evaluation which you want to automate and need consistent results.
2. It is difficult or impossible to explicitly describe the solution or criteria behind a decision.
3. You have labeled data, or existing examples where you can describe the situation and map it to the correct result.

6.2 EXPLORATORY DATAANALYSIS:

In statistics, exploratory data analysis is an approach of analyzing data sets to summarize their main characteristics, often using statistical graphics and other data visualization methods.

Steps in EDA:

- First step toward building a model.
- It's a method of systematically going through the data,
- Plotting distributions of all variables (using box plots),
- Plotting time series of data,
- Transforming variables,
- Looking at all pairwise relationships between variables using scatterplot matrices, and Generating summary statistics for all of them.
- Computing their mean, minimum, maximum, the upper and lower Quartiles,identifying outliers.

Why EDA?

- To gain intuition about the data;
- To make comparisons between distributions;
- For sanity checking (making sure the data is on the scale you expect, in the format you thought it should be);
- To find out where data is missing or if there are outliers; and
- EDA helps you make sure the product is performing as intended.

6.3 DEEP LEARNING CLASSIFIFCATION ALGORTIHMS

Deep learning has gained massive popularity in scientific computing, and its algorithms are widely used by industries that solve complex problems[1]. All deep learning algorithms use different types of neural networks to perform specific tasks. Deep learning uses artificial neural networks to perform sophisticated computations on large amounts of data. It is a type of machine learning that works based on the structure and function of the human brain.

While deep learning algorithms feature self-learning representations, they depend upon ANNs that mirror the way the brain computes information. During the training process, algorithms use unknown elements in the input distribution to extract features, group objects, and discover useful data patterns. Much like training machines for self-learning, this occurs at multiple levels, using the algorithms to build the models. Deep learning models make use of several algorithms. While no one network is considered perfect, some algorithms are better suited to perform specific tasks. To choose the right ones, it’s good to gain a solid understanding of all primary algorithms.

Types of Algorithms used in Deep Learning

Here is the list of top 10 most popular deep learning algorithms:

1. Convolutional Neural Networks (CNNs)
2. Long Short Term Memory Networks (LSTMs)
3. Recurrent Neural Networks (RNNs)
4. Generative Adversarial Networks (GANs)
5. Radial Basis Function Networks (RBFNs)
6. Multilayer Perceptrons (MLPs)
7. Self Organizing Maps (SOMs)
8. Deep Belief Networks (DBNs)
9. Restricted Boltzmann Machines(RBMs)
10. Autoencoders.

Deep learning algorithms work with almost any kind of data and require large amounts of computing power and information to solve complicated issues.

6.4 IMPORTANCE OF CNN MODEL:

Convolutional neural network (CNN), a class of artificial neural networks that has become dominant in various computer vision tasks, is attracting internet across a variety of domains, of features through backpropogation by using multiple building blocks such as convolution layers,pooling layers,and fully connected layers.

1. CNN is a type of deep learning model for processing data that has a grid pattern, such as images, which is inspired by the organization of animal visual cortex and designed to automatically and adaptively learn spatial hierarchies of features,from low- to high-level patterns.
2. CNN is a mathematical construct that is typically composed of three types of layers (or building blocks): convolution, pooling, and fully connected layers.
3. The first two, convolution and pooling layers, perform feature extraction, whereas the third, a fully connected layer, maps the extracted features into final output, such as classification.
4. A convolution layer plays a key role in CNN, which is composed of a stack of mathematical operations, such as convolution, a specialized type of linear operation.
5. In digital images, pixel values are stored in a two-dimensional (2D) grid, i.e., an array of numbers and a small grid of parameters called kernel, an optimizable feature extractor, is applied at each image position, which makes CNNs highly efficient for image processing, since a feature may occur anywhere in the image.
6. The process of optimizing parameters such as kernels is called training, which is performed so as to minimize the difference between outputs and ground truth labels through an optimization algorithm called backpropagation and gradient descent, among others.

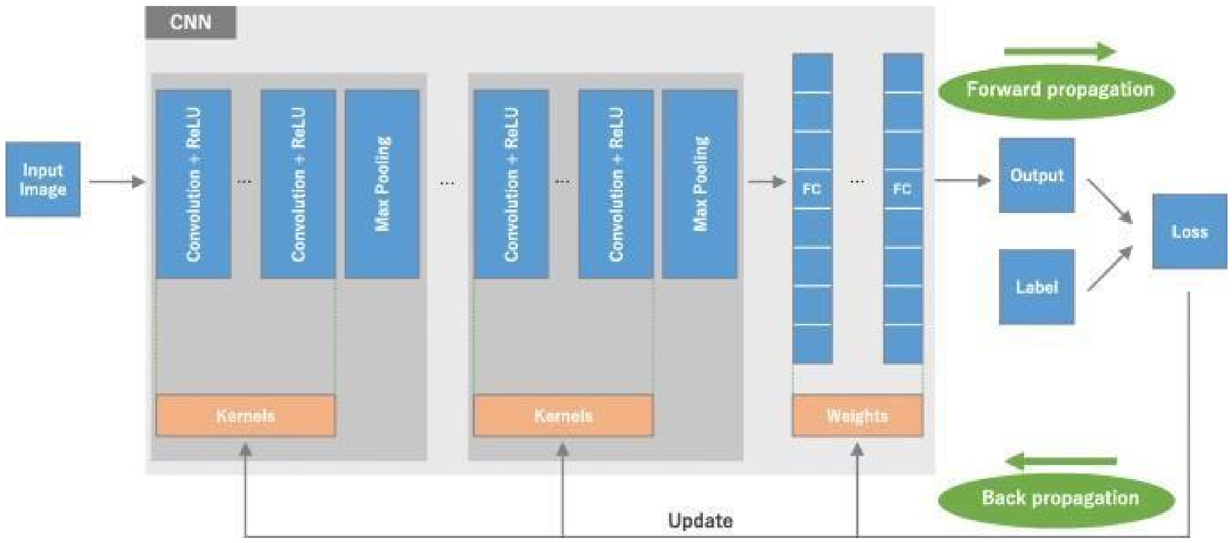


Fig 6.4 An Overview of CNN Model

An overview of a convolutional neural network (CNN) architecture and the training process. A CNN is composed of a stacking ofseveral building blocks: convolution layers, pooling layers (e.g., max pooling), and fully connected (FC) layers. A model's performance under particular kernels and weights[9] is calculated with a loss function through forward propagation on a training dataset, and learnable parameters, i.e., kernels and weights, are updated according to the loss value through backpropagation with gradient descent optimization algorithm.

Most recent radiomics studies use hand-crafted feature extraction techniques, such as texture analysis, followed by conventional machine learning classifiers, such as random forests and support vector machines. There are several differences to note between such methods and CNN. First, CNN does not require hand-crafted feature extraction. Second, CNN architectures do not necessarily require segmentation of tumors or organs by human experts.Third, CNN is far more data hungry because of its millions of learnable parameters to estimate, and, thus, is more computationally expensive, resulting in requiring graphical processing-units.

6.5 CNN ARCHITECTURE:

There are two main parts to a CNN architecture :

- A convolution tool that separates and identifies the various features of the image for analysis in a process called as Feature Extraction.
- The network of feature extraction consists of many pairs of convolutional or pooling layers.
- A fully connected layer that utilizes the output from the convolution process and predicts the class of the image based on the features extracted in previous stages.
- This CNN model of feature extraction aims to reduce the number of features present in a dataset.
- It creates new features which summarises the existing features contained in an original set of features.There are many CNN layers as shown in the CNN architecture diagram.

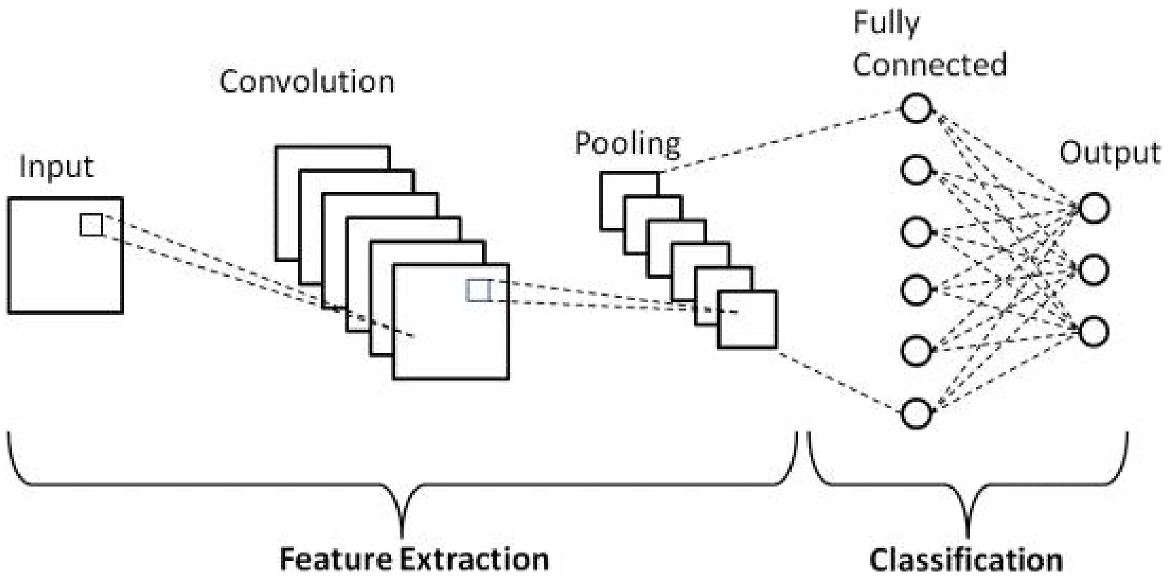


Fig 6.5 Convolutional Layers

1. ***Convolutional Layer***

- This layer is the first layer that is used to extract the various features from the input images.
- In this layer, the mathematical operation of convolution is performed between the input image and a filter of a particular size MxM.
- By sliding the filter over the input image, the dot product is taken between the filter and the parts of the input image with respect to the size of the filter (MxM).
- The output is termed as the Feature map which gives us information about the image such as the corners and edges. Later, this feature map is fed to other layers to learn several other features of the input image.

2. ***Pooling Layer***

- In most cases, a Convolutional Layer is followed by a Pooling Layer.
- The primary aim of this layer is to decrease the size of the convolved feature map to reduce the computational costs.This is performed by decreasing the connections between layers and independently operates on each feature map.
- Depending upon method used, there are several types of Pooling operations. It basically summarises the features generated by a convolution layer.
- In Max Pooling, the largest element is taken from feature map. Average Pooling calculates the average of the elements in a predefined sized Image section.
- The total sum of the elements in the predefined section is computed in Sum Pooling.
- The Pooling Layer usually serves as a bridge between the Convolutional Layer and the FC Layer.

- This CNN model generalises the features extracted by the convolution layer, and helps the networks to recognise the features independently. With the help of this, the computations are also reduced in a network.

3. Fully Connected Layer

- The Fully Connected (FC) layer consists of the weights and biases along with theneurons and is used to connect the neurons between two different layers.
- These layers are usually placed before the output layer and form the last few layers of a CNN Architecture.
- In this, the input image from the previous layers are flattened and fed to the FC layer.
- The flattened vector then undergoes few more FC layers where the mathematical functions operations usually take place.
- In this stage, the classification process begins to take place.
- The reason two layers are connected is that two fully connected layers will perform better than a single connected layer.
- These layers in CNN reduce the human supervision.

4. Dropout

- Usually, when all the features are connected to the FC layer, it can cause overfitting in the training dataset.
- Overfitting occurs when a particular model works so well on the training data causing a negative impact in the model's performance when used on a new data.
- To overcome this problem, a dropout layer is utilised wherein a few neurons are dropped from the neural network during training process resulting in reduced size of the model.

- On passing a dropout of 0.3, 30% of the nodes are dropped out randomly from the neural network.
- Dropout results in improving the performance of a machine learning model as it prevents overfitting by making the network simpler.
- It drops neurons from the neural networks during training.

5.Activation Functions

- Finally, one of the most important parameters of the CNN model is the activation function.
- They are used to learn and approximate any kind of continuous and complex relationship between variables of the network.
- In simple words, it decides which information of the model should fire in the forward direction and which ones should not at the end of the network.
- It adds non-linearity to the network. There are several commonly used activation functions such as the ReLU, Softmax, tanH and the Sigmoid functions. Each of these functions have a specific usage.
- For a binary classification CNN model, sigmoid and softmax functions are preferred an for a multi-class classification, generally softmax us used. In simple terms, activation functions in a CNN model determine whether a neuron should be activated or not.
- It decides whether the input to the work is important or not to predict using mathematical operations.

CHAPTER 7

IMPLEMENTATION CODE

7.1 Importing Libraries

```
import cv2

import mediapipe as mp

import pyautogui

import math

from enum import IntEnum

from ctypes import cast, POINTER

from comtypes import CLSCTX_ALL

from pycaw.pycaw import AudioUtilities, IAudioEndpointVolume

from google.protobuf.json_format import MessageToDict

import screen_brightness_control as sbcontrol



pyautogui.FAILSAFE = False

mp_drawing = mp.solutions.drawing_utils

mp_hands = mp.solutions.hands
```

7.2 Class Controller and variables declared.

```
class Controller:



    tx_old = 0

    ty_old = 0

    trial = True

    flag = False

    grabflag = False

    pinchmajorflag = False

    pinchminorflag = False

    pinchstartxcoord = None

    pinchstartycoord = None

    pinchdirectionflag = None

    prevpinchlv = 0

    pinchlv = 0

    framecount = 0

    prev_hand = None

    pinch_threshold = 0.3
```

7.3 Gesture classes

```
class Gest(IntEnum):

   FIST = 0

   PINKY = 1

   RING = 2

   MID = 4

   LAST3 = 7

   INDEX = 8

   FIRST2 = 12

   LAST4 = 15

   THUMB = 16

   PALM = 31




   V_GEST = 33

   TWO_FINGER_CLOSED = 34

   PINCH_MAJOR = 35

   PINCH_MINOR = 36
```

```
class HLabel(IntEnum):

    MINOR = 0

    MAJOR = 1




class HandRecog:



    def _init_(self, hand_label):




        self.finger = 0

        self.ori_gesture = Gest.PALM

        self.prev_gesture = Gest.PALM

        self.frame_count = 0

        self.hand_result = None

        self.hand_label = hand_label
```

7.4 Distance functions

```
def get_signed_dist(self, point):


    sign = -1

    if self.hand_result.landmark[point[0]].y < self.hand_result.landmark[point[1]].y:

        sign = 1

    dist = (self.hand_result.landmark[point[0]].x -
    self.hand_result.landmark[point[1]].x)**2

    dist += (self.hand_result.landmark[point[0]].y -
    self.hand_result.landmark[point[1]].y)**2

    dist = math.sqrt(dist)

    return dist*sign


  def get_dist(self, point):


    dist = (self.hand_result.landmark[point[0]].x -
    self.hand_result.landmark[point[1]].x)**2

    dist += (self.hand_result.landmark[point[0]].y -
    self.hand_result.landmark[point[1]].y)**2

    dist = math.sqrt(dist)

    return dist


  def get_dz(self,point):


    return abs(self.hand_result.landmark[point[0]].z - self.hand_result.landmark[point[1]].z)
```

7.5 Setting state of the finger

```
def set_finger_state(self):

    if self.hand_result == None:
        return

    points = [[8,5,0],[12,9,0],[16,13,0],[20,17,0]]
    self.finger = 0
    self.finger = self.finger | 0 #thumb
    for idx,point in enumerate(points):

        dist = self.get_signed_dist(point[:2])
        dist2 = self.get_signed_dist(point[1:])

        try:
            ratio = round(dist/dist2,1)
        except:
            ratio = round(dist1/0.01,1)

        self.finger = self.finger << 1
        if ratio > 0.5 :

            self.finger = self.finger | 1
```

7.6 Obtaining gesture

```
def get_gesture(self):


    if self.hand_result == None:

        return Gest.PALM


    current_gesture = Gest.PALM

    if self.finger in [Gest.LAST3,Gest.LAST4] and self.get_dist([8,4]) < 0.05:

        if self.hand_label == HLabel.MINOR :

            current_gesture = Gest.PINCH_MINOR

        else:

            current_gesture = Gest.PINCH_MAJOR


    elif Gest.FIRST2 == self.finger :

        point = [[8,12],[5,9]]

        dist1 = self.get_dist(point[0])

        dist2 = self.get_dist(point[1])

        ratio = dist1/dist2

        if ratio > 1.7:

            current_gesture = Gest.V_GEST

        else:

            if self.get_dz([8,12]) < 0.1:
            current_gesture =  Gest.TWO_FINGER_CLOSED
            else:
            current_gesture =  Gest.MID


    else:
        current_gesture =  self.finger


    if current_gesture == self.prev_gesture:
        self.frame_count += 1
    else:
        self.frame_count = 0


    self.prev_gesture = current_gesture


    if self.frame_count > 4 :
        self.ori_gesture = current_gesture
    return self.ori_gesture
```

7.7 System functions are defined

```
def changesystembrightness():


        currentBrightnessLv = sbcontrol.get_brightness(display=0)/100.0
        currentBrightnessLv += Controller.pinchlv/50.0

        if currentBrightnessLv > 1.0:

            currentBrightnessLv = 1.0

        elif currentBrightnessLv < 0.0:

            currentBrightnessLv = 0.0

        sbcontrol.fade_brightness(int(100*currentBrightnessLv) , start = 
sbcontrol.get_brightness(display=0))



    def changesystemvolume():



        devices = AudioUtilities.GetSpeakers()

        interface = devices.Activate(IAudioEndpointVolume.iid, CLSCTX_ALL, None)

        volume = cast(interface, POINTER(IAudioEndpointVolume))

        currentVolumeLv = volume.GetMasterVolumeLevelScalar()

        currentVolumeLv += Controller.pinchlv/50.0

        if currentVolumeLv > 1.0:

            currentVolumeLv = 1.0

        elif currentVolumeLv < 0.0:

            currentVolumeLv = 0.0

        volume.SetMasterVolumeLevelScalar(currentVolumeLv, None)
```

```
def scrollVertical():



    pyautogui.scroll(120 if Controller.pinchlv>0.0 else -120)



def scrollHorizontal():

    pyautogui.keyDown('shift')

    pyautogui.keyDown('ctrl')

    pyautogui.scroll(-120 if Controller.pinchlv>0.0 else 120)

    pyautogui.keyUp('ctrl')

    pyautogui.keyUp('shift')
```

7.8 Function for detection of the position of the finger and for pinch control

```
def get_position(hand_result):



    point = 9

    position = [hand_result.landmark[point].x ,hand_result.landmark[point].y]

    sx,sy = pyautogui.size()

    x_old,y_old = pyautogui.position()

    x = int(position[0]*sx)

    y = int(position[1]*sy)

    if Controller.prev_hand is None:

        Controller.prev_hand = x,y

    delta_x = x - Controller.prev_hand[0]

    delta_y = y - Controller.prev_hand[1]



    distsq = delta_x*2 + delta_y*2

    ratio = 1

    Controller.prev_hand = [x,y]



    if distsq <= 25:

        ratio = 0
    elif distsq <= 900:
        ratio = 0.07 * (distsq ** (1/2))
    else:
        ratio = 2.1
    x , y = x_old + delta_x*ratio , y_old + delta_y*ratio
    return (x,y)


def pinch_control_init(hand_result):
    Controller.pinchstartxcoord = hand_result.landmark[8].x
    Controller.pinchstartycoord = hand_result.landmark[8].y
    Controller.pinchlv = 0
    Controller.prevpinchlv = 0
    Controller.framecount = 0


# Hold final position for 5 frames to change status
def pinch_control(hand_result, controlHorizontal, controlVertical):

    if Controller.framecount == 5:
        Controller.framecount = 0
        Controller.pinchlv = Controller.prevpinchlv
        if Controller.pinchdirectionflag == True:

            controlHorizontal() #x



        elif Controller.pinchdirectionflag == False:

            controlVertical() #y



    lvx =  Controller.getpinchxlv(hand_result)

    lvy =  Controller.getpinchylv(hand_result)



    if abs(lvy) > abs(lvx) and abs(lvy) > Controller.pinch_threshold:

        Controller.pinchdirectionflag = False

        if abs(Controller.prevpinchlv - lvy) < Controller.pinch_threshold:

            Controller.framecount += 1

        else:

            Controller.prevpinchlv = lvy

            Controller.framecount = 0



    elif abs(lvx) > Controller.pinch_threshold:

        Controller.pinchdirectionflag = True
        if abs(Controller.prevpinchlv - lvx) < Controller.pinch_threshold:

            Controller.framecount += 1

        else:

            Controller.prevpinchlv = lvx

            Controller.framecount = 0
```

7.9 Function for handling controlling of gesture

```
def handle_controls(gesture, hand_result):

        x,y = None,None

        if gesture != Gest.PALM :

            x,y = Controller.get_position(hand_result)



        # flag reset

        if gesture != Gest.FIST and Controller.grabflag:

            Controller.grabflag = False

            pyautogui.mouseUp(button = "left")



        if gesture != Gest.PINCH_MAJOR and Controller.pinchmajorflag:

            Controller.pinchmajorflag = False
```

```
if gesture != Gest.PINCH_MINOR and Controller.pinchminorflag:

    Controller.pinchminorflag = False



# implementation

if gesture == Gest.V_GEST:

    Controller.flag = True

    pyautogui.moveTo(x, y, duration = 0.1)



elif gesture == Gest.FIST:

    if not Controller.grabflag :

        Controller.grabflag = True

        pyautogui.mouseDown(button = "left")

    pyautogui.moveTo(x, y, duration = 0.1)



elif gesture == Gest.MID and Controller.flag:

    pyautogui.click()

    Controller.flag = False



elif gesture == Gest.INDEX and Controller.flag:

    pyautogui.click(button='right')
```

```
            Controller.flag = False



        elif gesture == Gest.TWO_FINGER_CLOSED and Controller.flag:

            pyautogui.doubleClick()

            Controller.flag = False



        elif gesture == Gest.PINCH_MINOR:

            if Controller.pinchminorflag == False:

                Controller.pinch_control_init(hand_result)

            Controller.pinchminorflag = True

            Controller.pinch_control(hand_result,Controller.scrollHorizontal,
Controller.scrollVertical)

       elif gesture == Gest.PINCH_MAJOR:

            if Controller.pinchmajorflag == False:

                Controller.pinch_control_init(hand_result)

                Controller.pinchmajorflag = True

            Controller.pinch_control(hand_result,Controller.changesystembrightness,
Controller.changesystemvolume)
```

CHAPTER 8

RESULTS AND DISCUSSION

8.1 THEORY ABOUT USED LIBRARIES

1.OPEN CV

OpenCV is a huge open-source library for computer vision, machine learning, and image processing. OpenCV supports a wide variety of programming languages like Python, C++, Java, etc. It can process images and videos to identify objects, faces, or even the handwriting of a human. When it is integrated with various libraries, such as Numpy which is a highly optimized library for numerical operations, then the number of weapons increases in yourArsenal i.e whatever operations one can do in Numpy can be combined with OpenCV.

The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects,produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc.

Open CV performs Image processing basically includes the following three steps:

1.Importing the image

2.Analysing and manipulating the image.

3.Output in which result can be altered image or report that is based on image

analysis.

2.NUMPY

NumPy is a Python library used for working with arrays. It also has functions for working in the domain of linear algebra, Fourier transform, and matrices. NumPy was created in 2005 by Travis Oliphant. It is an open source project and you can use it freely. NumPy stands for Numerical Python. It is written partially in Python, but most of the parts that require fast computation are written in C or C++.

3.MEDIAPIPE

- **Mediapipe** is a cross-platform library developed by Google that provides amazing ready to-use ML solutions for computer vision tasks.

- MediaPipe powers revolutionary products and services we use daily. Unlike power-hungry machine learning Frameworks, MediaPipe requires minimal resources. It is so tiny and efficient that even embedded IoT devices can run it.

- This cross-platform Framework works in Desktop/Server, Android, iOS, and embedded devices like Raspberry Pi and Jetson Nano.

4.PyPI

The Python Package Index (PyPI) is a repository ofsoftware for the Python programming language. PyPI as an index allows users to search for packages by keywords or by filters against their metadata, such as free software license or compatibility with POSIX. [10] A single entry on PyPI is able to store, aside from just a package and its metadata, previous releases of the package, precompiled wheels (e.g. containing DLLs on Windows), as well as different forms for different operating systems and Python versions.

8.2 RESULT DISCUSSION:

Features:

- Neutral Gesture
- Right Click
- Left Click
- Double Click
- Drag and Drop
- Volume Control
- Brightness Control
- Multiple Item Selection

WORKING ENVIRONMENT:

Anaconda supports multiple versions of Python and associated packages. An environment generally includes one version of Python or R language and some packages.The ability to have a custom environment for your project is one of the most powerful features of Anaconda Enterprise Notebooks.

Conda is an open-source package and environment management system that runs onWindows, macOS, and Linux. Conda quickly installs, runs, and updates packages and theirdependencies. It also easily creates, saves, loads, and switches between environments on your local computer. It was created for Python programs, but it can package and distribute software for any language.

8.3 OUTPUTS

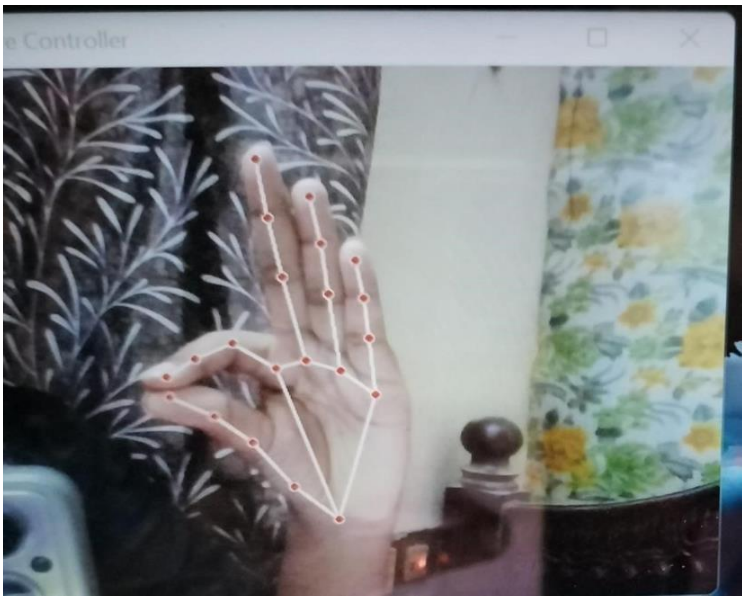


Fig 8.3.1 Gesture showing movement of the cursor

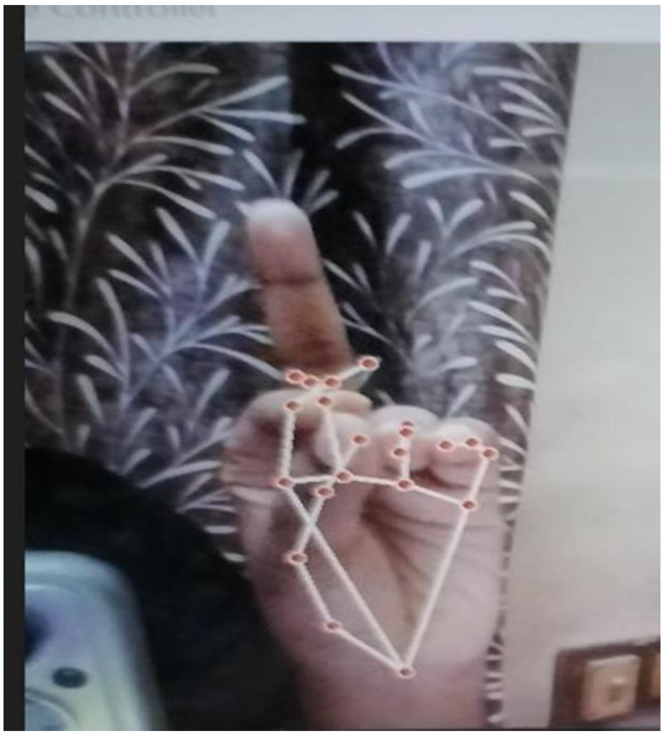


Fig 8.3.2 Gesture of left click

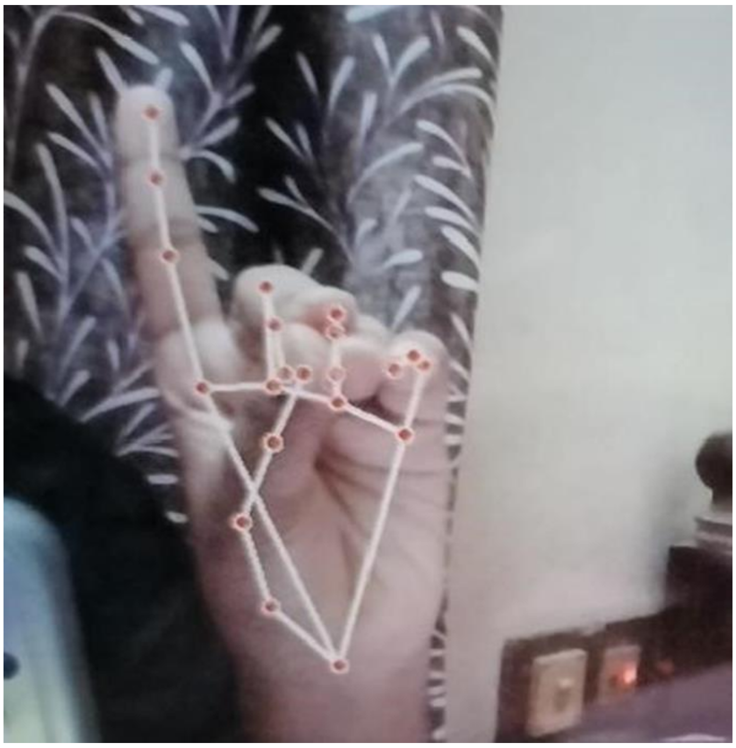


Fig 8.3.3 Gesture of Right Click

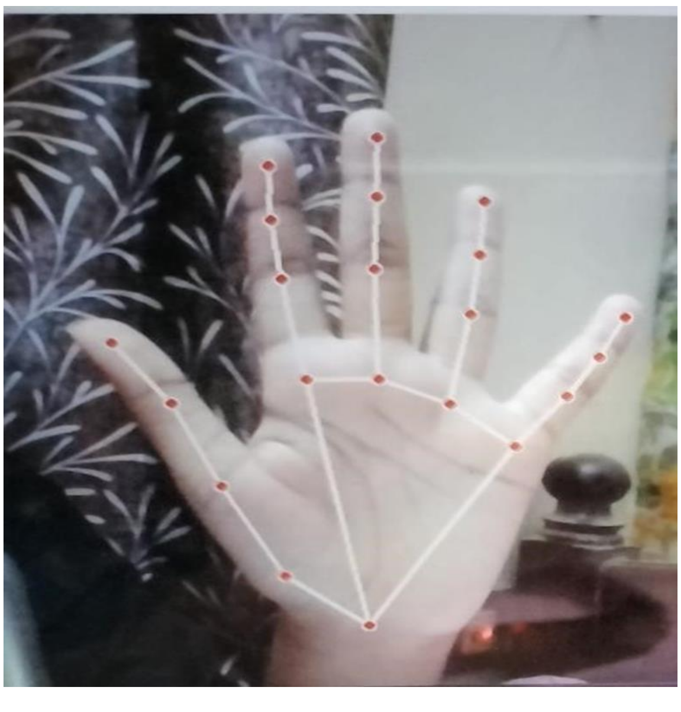


Fig 8.3.4 Neutral Gesture

CHAPTER 9

CONCLUSION

In this modern world, where technologies are at the peak, there are many facilities available for offering input to any applications running on the computer systems, some of the inputs can be offered using physical touch and some of them without using physical touch (likespeech, hand gestures, head gestures etc.). Using hand gestures many users can handle applications from distance without even touching it. But there are many applications which cannot be controlled using hand gestures as an input. This technique can be very helpful forphysically challenged people because they can define the gesture according to their need.We need to improve our system and try to build more robust algorithm for both recognition and detection even i the cluttered background and a
normal lighting condition. The main goal of the visual mouse system is to allow the user to control the mouse pointer function with a hand touch instead of manipulating things. Keyboard and mouse actually form an integral part of the computer system. Our system architecture can facilitate the use of computer for the paralyzed people. The use of object detection and image processing in OpenCV for the implementation of our work has proved to be practically successful and the task of keyboard and mouse is achieved with good precision. Most of the applications require additional hardware which are often very expensive. The motive of this work is to create this technology as cheaply as possible and to create it under a standardized operating system as well.

CHAPTER 10

REFERNCES

[1] G. R. S. Murthy, R. S. Jadon. (2009). "A Review of Vision Based Hand Gestures Recognition,"International Journal of Information Technology and Knowledge Management, vol. 2(2), pp. 405.

[2] P. Garg, N. Aggarwal and S. Sofat. (2009). "Vision Based Hand Gesture Recognition," WorldAcademy of Science, Engineering and Technology, Vol. 49, pp. 972-977.[3] Fakhreddine Karray, Milad Alemzadeh, Jamil Abou Saleh, Mo Nours Arab, (2008) ."Human-Computer Interaction: Overview on State of the Art", International Journal on Smart Sensing andIntelligent Systems,

[3] Rafiqul Z. Khan, Noor A. Ibraheem, (2012). "Survey on Gesture Recognition for Hand ImagePostures", International Journal of Computer And Information Science, Vol. 5(3), Doi:10.5539/cis.v5n3p110

[4] N. A. Ibraheem., R. Z. Khan, (2012). "Vision Based Gesture Recognition Using Neural NetworksApproaches: A Review", International Journal of Human Computer Interaction (IJHCI), Malaysia,Vol. 3(1).[25]

[5] Min B., Yoon, H., Soh, J., Yangc, Y., & Ejima, T. (1997). "Hand Gesture Recognition Using HiddenMarkov Models". IEEE International Conference on computational cybernetics and simulation. Vol.5, Doi: 10.1109/ICSMC.1997.637364.

[6] Minghai Y., Xinyu Q., Qinlong G., Taotao R., Zhongwang L., (2010). "Online PCA with AdaptiveSubspace Method for Real-Time Hand Gesture Learning and Recognition", journal World Scientificand Engineering Academy and SocietWSEAN, Vol. 9(6).

[7]Shuying Zhao, Wenjun Tan, Shiguang Wen, and Yuanyuan Liu, (2008). "An Improved Algorithm ofHand Gesture Recognition under Intricate Background", Springer the First International Conferenceon Intelligent Robotics and Applications (ICIRA 2008),: Part I. pp. 786–794, 2008.Doi:10.1007/978-3-540-88513-9_85

[8]Rafiqul Z. Khan, Noor A. Ibraheem, (2012). "Survey on Gesture Recognition for Hand ImagePostures", International Journal of Computer And Information Science, Vol. 5(3), Doi: 10.5539/cis.v5n3p110

[9]Verma, R., Dev A. (2009)."Vision based hand gesture recognition using finite state machines and fuzzy logic". IEEE International Conference on Ultra-Modern Telecommunications&Workshops(ICUMT '09), pp. 1-6. doi: 10.1109/ICUMT.2009.5345425.