A
Major Project report on

OBSTACLE AVOIDANCE ALGORITHM USING DEEP LEARNING

Submitted To
Jawaharlal Nehru Technological University Hyderabad
In Partial Fulfilment of the Requirements for the Award of Degree Of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted By

ABHIRAV KULKARNI	177Z1A0501
A.SAI VAMSHI	177Z1A0503
M.SAMUEL JAYAKUMAR	177Z1A0552

Under the Guidance of
Mr. V. VEERABHADRAM
Associate Professor

SCHOOL OF ENGINEERING
Department of Computer Science and Engineering

NALLA NARASIMHA REDDY
EDUCATION SOCIETY'S GROUP OF INSTITUTIONS
(Approved by AICTE, New Delhi, Affiliated to JNTU-Hyderabad)
Chowdariguda (Vil.) Korremula 'x' Roads, Via Narapally, Ghatkesar (Mandal)
Medchal (Dist.), Telangana-500088

2020-2021

Nalla Narasimha Reddy Group of Institutions
NNRG Integrated Campus
NALLA NARASIMHA REDDY
Education Society's Group of Institutions - Integrated Campus
Approved by AICTE, New Delhi, Affiliated to JNTU - Hyderabad
SCHOOL OF ENGINEERING
DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that the project report titled **"Obstacle Avoidance algorithm using Deep learning"** is being submitted by **Abhirav Kulkarni** (**177Z1A0501**), **A.Sai Vamshi** (**177Z1A0503**), and **M.Samuel Jayakumar** (**177Z1A0552**) in Partial fulfilment for the award of **Bachelor of Technology in Computer Science & Engineering** is a record bonafide work carried out by them. The results embodied in this report have not been submitted to any other University for the award of any degree.

Internal Guide
(Mr. V.Veerabhadram)

Head of the Department
(Dr. K.Rameshwaraiah)

Director
(Dr..C.V.Krishna Reddy)

DECLARATION

We Abhirav Kulkarni, A.Sai Vamshi and M.Samuel Jayakumar, are students of **Bachelor of Technology in Computer Science and Engineering, 2020-2021** Institute of **Nalla Narasimha Reddy Education Society's Group of Institutions**, Hyderabad, Telangana State, hereby declare that the work presented in this project work entitled **Obstacle avoidance algorithm using Deep Learning** is the outcome of our own bonafide work and is correct to the best of our knowledge and this work has been undertaken taking care of engineering ethics. It contains no material previously published or written by another person nor material which has been accepted for the award of any other degree or diploma of the university or other institute of higher learning.

Abhirav Kulkarni 177Z1A0501

A.Sai Vamshi 177Z1A0503

M.Samuel Jayakumar 177Z1A0552

Date:

ACKNOWLEDGEMENT

We express our sincere gratitude to our guide **Mr. V.Veerabhadram,** Associate Professor in Computer Science and Engineering Department who motivated throughout the period of the project and also for his valuable and intellectual suggestions apart from his adequate guidance, constant encouragement right throughout our work.

We wish to record our deep sense of gratitude to our Project Co-ordinator **Mr.R.Ramesh**, Assistant Professor in Computer Science and Engineering Department, who has guided the project with scholarly advice, deep interest and warm personal affection. He has been a constant source of encouragement and inspired us in completing the project and offered valuable guidance from time to time, we are very thankful to him.

We profoundly express thanks to **Dr.K.Rameshwaraiah,** Head of Computer Science and Engineering Department, for his cooperation and encouragement in completing the project successfully.

We wish to express our sincere thanks to **Dr.G.Janardhana Raju,** Dean School of Engineering for providing the facilities for completion of the project.

We wish to express our sincere thanks to **Dr**. **C.V.Krishna Reddy,** Director NNRESGI for providing the facilities for completion of the project.

Finally, we would like to thank all the faculty members, supporting staff of the Department of Computer Science and Engineering for extending their help in all circumstances.

By

Abhirav Kulkarni **177Z1A0501**

A.Sai Vamshi **177Z1A0503**

M.Samuel Jayakumar **177Z1A0552**

ABSTRACT

Inspired by the advantages of the hierarchical feature extraction of deep learning. This work investigates the development of a Convolutional Neural Network (CNN) algorithm to solve the problem of the mobile robot obstacle avoidance in an indoor environment.

The algorithm takes raw images as input and generates control commands as network output. Control commands include go straight-forward, turn-full-left, turn-half-left, turnfull-right, and turn-half-right.

CONTENTS

	Page No.
List of Figures	**i**
List of Tables	**ii**
1. INTRODUCTION	**1**
1.1 Motivation	1
1.2 Problem Definition	1
1.3 Objective of project	2
1.4 Limitation of project	2
2. LITERATURE SURVEY	**3**
2.1 Introduction	3
2.2 Existing System	3
2.3 Proposed System	4
3. SYSTEM ANALYSIS	**5**
3.1 Introduction	5
3.2 Software requirements	5
3.3 Hardware requirements	6
3.4 Content Diagram of Project	6
4. SYSTEM DESIGN	**7**
4.1 Introduction	7
4.2 DFD/ER/UML Diagrams	8
5. IMPLEMENTATION & RESULTS	**19**
5.1 Introduction	19
5.2 Method of Implementation	19
5.3 Explanation of key functions	40

5.4 Output Screens 44

6. TESTING 47

6.1 Introduction 47

6.2 Design of test cases and scenerios 58

7.CONCLUSION 62

6.3 Project Conclusion 62

6.4 Future Enhancement 62

8.REFERENCES 63

6.5 Text Books 63

6.6 Websites 64

6.7 Paper References 64

List of Figures

Name of the figure		Page no.
3.1	Content diagram of the project	6
4. 1	Use case diagram	12
4.2	Class diagram	14
4.3	Sequence diagram	16
4.4	Collaboration diagram	18
5.1	Output Screen-1	44
5.2	Output Screen-2	44
5.3	Output Screen-3	45
5.4	Output Screen-4	45
5.5	Output Screen-5	46
6.1	Levels of Testing	51

List of Tables

Name of the table		Page no.
6.1	Advantages and Disadvantages of White box testing	59
6.2	Advantages and Disadvantages of Black box testing	60
6.3	Scenarios	61

1. INTRODUCTION

Navigation is one of the most important tasks of mobile robots, where staying operational while avoiding collisions and maintaining safety standard are priorities in mobile robots. To develop an autonomous mobile robot, we need to build a system that can grasp environments, react to unexpected events, and plan dynamically in order to achieve the mission.

This work is concerned with the problem of obstacle avoidance in indoor environments for vision-based mobile robots. The main goal is to develop a deep learning algorithm for obstacle avoidance using raw images of robot- environment as input and generate control commands as network output, to solve the problem of real-time robot navigation.

In order to achieve the main objective, a new dataset is compiled using depth images (RGBD) and robot orientation data obtained by an Inertial Measurement Unit (IMU). RGBD is an image channel in which each pixel relates to a distance between the image plane and the corresponding object in the RGB image.

1.1 MOTIVATION

This work combines the perception stage and the control stage with a single deep network. The network structure fuses CNN with the decision-making process. The adopted CNN algorithm contains three convolutional stages; each stage has three layers “convolution + activation + pooling”, followed by one fully connected layer. The CNN structure is used to detect and understand visual features and the fully connected layers are for decision-making.

1.2 PROBLEM DEFINITION

The motion planning and control problem is a well-known problem in the field of robotics. The objective is to find collision-free trajectories for a robot, in static environment containing some obstacles, between a start and a goal configuration. It has attracted much research in recent years. In this context the term control has a broad meaning that includes

many different controls, such as low-level motor control, and behaviour control, where behaviour represents many complicated tasks, like obstacle avoidance and goal seeking.

1.3 OBJECTIVE OF PROJECT

Inspired by the advantages of the hierarchical feature extraction of deep learning, this work investigates the development of a Convolutional Neural Network (CNN) algorithm to solve the problem of the mobile robot obstacle voidance in an indoor environment

1.4 LIMITATIONS

- It is painfully slow to train.
- The network architecture weights themselves are quite large.

2. LITERATURE SURVEY

A literature survey or a literature review in a project report is that section which shows the various analyses and research made in the field of your interest and the results already published, taking into account the various parameters of the project and the extent of the project.

It is the most important part of your report as it gives you a direction in the area of your research. It helps you set a goal for your analysis - thus giving you your problem statement.

Literature survey is something when you look at a literature (publications) in a surface level, or an Ariel view. It incorporates the study of place people and productions are setting of research. It is phase where the analyst tries to know about what is all the literature related with one range of interest. Also, the relevant literature works are short-listed. Moreover, literature survey guides or helps the researcher to define/find out/identify a problem.

2.1 INTRODUCTION

The main purpose of the literature review work was to survey previous studies on knowledge sharing and intranets. In this, we look into the details about the existing system and try to reduce the disadvantages of the existing system. We try to improve the performance and the efficiency of the new proposed system and also learn the advantages of proposed system.

2.2 EXISTING SYSTEM

Classical perception methods extract information from the raw sensors readings based on artificially designed complex features.

These methods are designed to adopt to generic environments.

These methods are prone to errors when they encounter unstructured dynamic environments wide illuminations differenced and different terrain types.

2.2.1 DISADVANTAGES

There are few disadvantages identified in the existing system and are defined below:

i. Use of database
ii. Inaccurate results
iii. Time complexity is more
iv. Accuracy of algorithms is less

2.3 PROPOSED SYSTEM

The objective of the robot's motion planning and the control is to find Collison free paths between two positions. To achieve the main objective a new dataset is compiled using depth images (RGBD) .The baseline control algorithm chosen in this work is the VGG16(convolution neural network model for large scale image recognition)

2.3.1 ADVANTAGES

There are advantages in the proposed system which could overcome the drawbacks of the existing system and are defined below:

i. Accurate results
ii. Better Accuracy
iii. Robustness
iv. Less errors

3. SYSTEM ANALYSIS

It is a process of collecting and interpreting facts, identifying the problems, and decomposition of a system into its components.

System analysis is conducted for the purpose of studying a system or its parts in order to identify its objectives. It is a problem solving technique that improves the system and ensures that all the components of the system work efficiently to accomplish their purpose. Analysis specifies what the system should do.

3.1 INTRODUCTION

In this phase the requirements are gathered and analysed. User's requirements are gathered in this phase. This phase is the main focus of the users and their interaction with the system.

There are few questions raised:

- Who is going to use the system?
- How will they use the system?
- What data should be input into the system?
- What data should be output by the system?

These general questions are answered during a requirement gathering phase. After requirement gathering these requirements are analysed for their validity and the possibility of incorporating the requirements in the system to be development is also studied.

Finally, a Requirement Specification document is created which serves the purpose of guideline for the next phase of the model.

3.2 SOFTWARE REQUIREMENTS

It deals with defining software resource requirements and prerequisites that need to be installed on a computer to provide optimal functioning of an application. These requirements or prerequisites are generally not included in the software installation package and need to be installed separately before the software is installed. The software requirements are description of features and functionalities of the target system. Requirements convey the expectations of users from the software product. The requirements can be obvious or hidden, known or

unknown, expected or unexpected from client's point of view. We should try to understand what sort of requirements may arise in the requirement elicitation phase and what kinds of requirements are expected from the software system.

The software requirements that are required for this project are as follows:

- Operating System : Windows 7/8/10
- Technology : TensorFlow, NumPy, open cv,Keras
- Coding Language : Python 2
- UML's : Star Uml
- Editor : Python Idle-3

3.3 HARDWARE REQUIREMENTS

The most common set of requirements defined by any operating system or software application is the physical computer resources, also known as hardware, a hardware requirements list is often accompanied by a hardware compatibility list (HCL), especially in case of operating systems. An HCL lists tested, compatible, and sometimes incompatible hardware devices for a particular operating system or application.

The hardware requirements that are required for this project are as follows:

- Processor : minimum intel i5 or more
- RAM : minimum 4 GB
- Hard Disk : minimum 250 GB

3.4 CONTENT DIAGRAM OF PROJECT

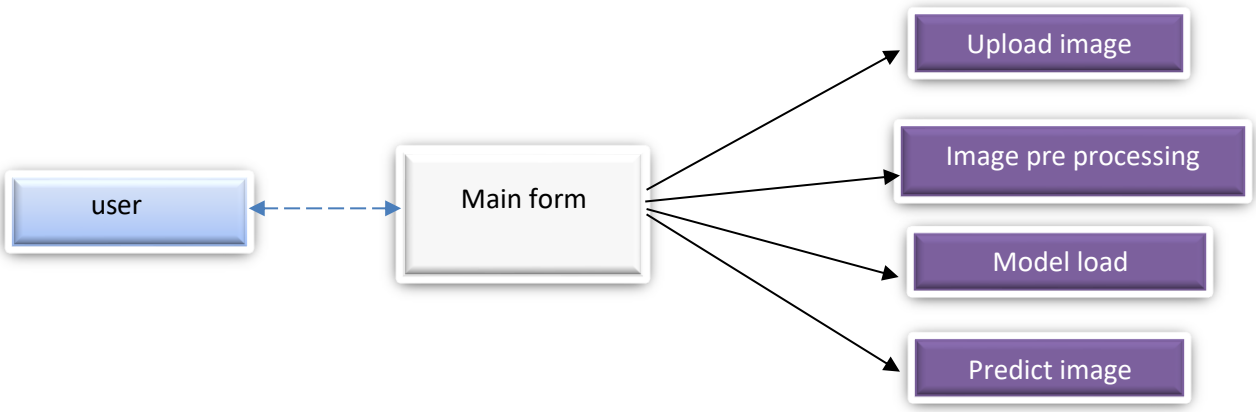

Figure 3.1: Content diagram of the project

4. SYSTEM DESIGN

The process of design involves “conceiving and planning out in mind and making a drawing, pattern or a sketch”. The system design transforms a logical representation of what a given system is required to do into the physical reality during development. Important design factors such as reliability, response time, throughput of the system, maintainability, expandability etc., should be taken into account. Design constraints like cost, hardware limitations, standard compliance etc should also be dealt with. The task of system design is to take the description and associate with it a specific set of facilities-men, machines (computing and other), accommodation, etc., to provide complete specifications of a workable system. This new system must provide for all of the essential data processing and it may also do some of those tasks identified during the work of analysis as optional extras. It must work within the imposed constraints and show improvement over the existing system.. At the outset of design a choice must be made between the main approaches. Talks of ‘preliminary design’ concerned with identification analysis and selections of the major design options are available for development and implementation of a system. These options are most readily distinguished in terms of the physical facilities to be used for the processing who or what does the work.

4.1 INTRODUCTION

Software design is the process by which an agent creates a specification of a software artifact, intended to accomplish goals, using a set of primitive components and subject to constraints. Software design may refer to either "all the activity involved in conceptualizing, framing, implementing, commissioning, and ultimately modifying complex systems" or "the activity following requirements specification and before programming, as in a stylized software engineering process." Software design usually involves problem solving and planning a software solution. This includes both a low-level component design and a high-level, architecture design.

Design is the first step in the development phase for any techniques and principles for the purpose of defining a device, a process or system in sufficient detail to permit its physical realization.

Once the software requirements have been analyzed and specified the software design involves four technical activities – design, coding, implementation and testing that are required to build and verify the software.

The design activities are of main importance in this phase, because in this activity, decisions ultimately affecting the success of the software implementation and its ease of maintenance are made. These decisions have the final bearing upon reliability and maintainability of the system. Design is the only way to accurately translate the customer's requirements into finished software or a system.

4.2 DFD/ER/UML DIAGRAMS

UML stands for Unified Modelling Language which is used in object oriented software engineering. It is a standard language for specifying, visualizing, constructing, and documenting the artefacts of the software systems. UML is different from other common programming languages like C++, Java, and COBOL etc. It is pictorial language used to make software blueprints.

Although typically used in software engineering it is a rich language that can be used to model an application structures, behaviour and even business processes. There are 8 UML diagram types to help us model this behaviour.

There are two types of UML modelling:

- Structural Modelling
- Behavioural Modelling

Structural Modelling:

Structural model represents the framework for the system and this framework is the place where all other components exist. Hence, the class diagram, component diagram and deployment diagrams are part of structural modelling. They all represent the elements and the mechanism to assemble them.

The structural model never describes the dynamic behaviour of the system. Class diagram is the most widely used structural diagram.

Structural Modelling captures the static features of a system. They consist of the following:

i. Classes diagrams
ii. Objects diagrams
iii. Deployment diagrams
iv. Package diagrams
v. Composite structure diagram

vi. Component diagram

Behavioural Modelling:

Behavioural model describes the interaction in the system. It represents the interaction among the structural diagrams. Behavioural modelling shows the dynamic nature of the system. They consist of the following:

i. Activity diagrams
ii. Interaction diagrams
iii. Use case diagrams

All the above show the dynamic sequence of flow in a system.

4.2.1 USE CASE DIAGRAM:

A use case diagram is a dynamic or behaviour diagram in UML. Use case diagrams model the functionality of a system using actors and use cases. Use cases are a set of actions, services, and functions that the system needs to perform. The "actors" are people or entities operating under defined roles within the system.

As the most known diagram type of the behavioural UML diagrams, use-case diagrams gives a graphic overview of the characters involved in a system, different functions needed by those characters and how these different functions are interacted.

Use case diagrams are valuable for visualizing the functional requirements of a system that will translate into design choices and development priorities. They also help identify any internal or external factors that may influence the system and should be taken into consideration. They provide a good high level analysis from outside the system. Use case diagrams specify how the system interacts with actors without worrying about the details of how that functionality is implemented.

A key concept of use case modeling is that it helps us design a system from the end user's perspective. It is an effective technique for communicating system behavior in the user's terms by specifying all externally visible system behavior.

Purpose of Use Case Diagram:

Use case diagrams are typically developed in the early stage of development and are used to gather the requirements of a system including internal and external influences. These

requirements are mostly design requirements. Hence, when a system is analyzed to gather its functionalities, use cases are prepared and actors are identified.
When the initial task is complete, use case diagrams are modelled to present the outside view.
In brief, the purposes of use case diagrams can be said to be as follows −

- Specify the context of a system
- Capture the requirements of a system
- Validate a systems architecture
- Drive implementation and generate test cases
- Developed by analysts together with domain experts

Guidelines to draw a Use Case Diagram:

Use case diagrams are considered for high level requirement analysis of a system. When the requirements of a system are analyzed, the functionalities are captured in use cases.
We can say that use cases are nothing but the system functionalities written in an organized manner. The second thing which is relevant to use cases are the actors. Actors can be defined as something that interacts with the system.

Actors can be a human user, some internal applications, or may be some external applications. When we are planning to draw a use case diagram, we should have the following items identified.

- Functionalities to be represented as use case
- Actors
- Relationships among the use cases and actors.

Use case diagrams are drawn to capture the functional requirements of a system. After identifying the above items, we have to use the following guidelines to draw an efficient use case diagram

- The name of a use case is very important. The name should be chosen in such a way so that it can identify the functionalities performed.
- Give a suitable name for actors.
- Show relationships and dependencies clearly in the diagram.
- Do not try to include all types of relationships, as the main purpose of the diagram is to identify the requirements.

- Use notes whenever required to clarify some important points.

Usage of Use Case Diagrams:

Use case diagrams specify the events of a system and their flows. But use case diagram never describes how they are implemented. Use case diagram can be imagined as a black box where only the input, output, and the function of the black box is known.

These diagrams are used at a very high level of design. This high level design is refined again and again to get a complete and practical picture of the system. A well-structured use case also describes the pre-condition, post condition, and exceptions. These extra elements are used to make test cases when performing the testing.

Although use case is not a good candidate for forward and reverse engineering, still they are used in a slightly different way to make forward and reverse engineering. The same is true for reverse engineering. Use case diagram is used differently to make it suitable for reverse engineering.

In forward engineering, use case diagrams are used to make test cases and in reverse engineering use cases are used to prepare the requirement details from the existing application. Use case diagrams can be used for −

- Requirement analysis and high level design.
- Model the context of a system.
- Reverse engineering.
- Forward engineering.

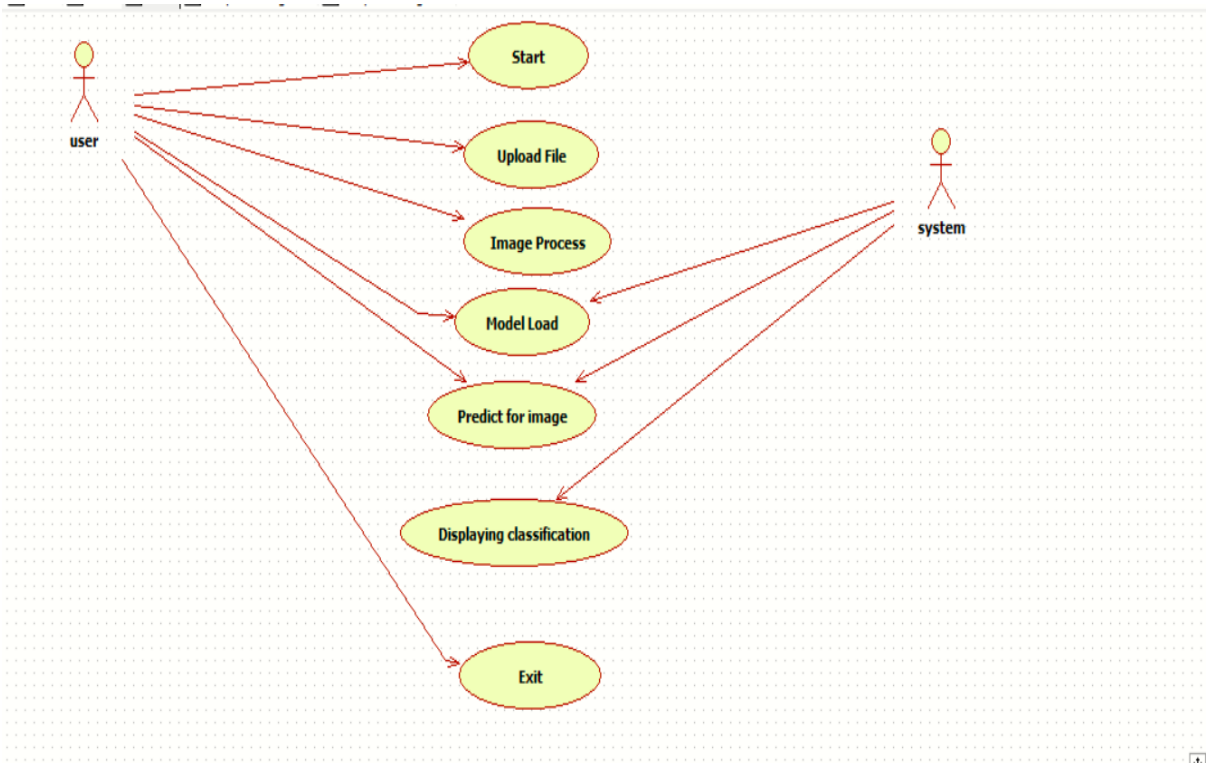

Figure 4.1: Use case diagram

4.2.2 CLASS DIAGRAM

Class diagrams are the main building blocks of every object oriented methods. It is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application.

Class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modeling of object oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages.

Class diagram shows a collection of classes, interfaces, associations, collaborations, and constraints. It is also known as a structural diagram.

Purpose of Class Diagram:

The purpose of class diagram is to model the static view of an application. Class diagrams are the only diagrams which can be directly mapped with object-oriented languages and thus widely used at the time of construction.

UML diagrams like activity diagram, sequence diagram can only give the sequence flow of the application; however class diagram is a bit different. It is the most popular UML diagram in the coder community.

The purpose of the class diagram can be summarized as –

- This is the only UML which can appropriately depict various aspects of OOPs concept.
- Analysis and design of the static view of an application.
- Describe responsibilities of a system.
- Base for component and deployment diagrams.
- Forward and reverse engineering.

Guidelines to draw a Class diagram:

Class diagrams are the most popular UML diagrams used for construction of software applications. It is very important to learn the drawing procedure of class diagram.

Class diagrams have a lot of properties to consider while drawing but here the diagram will be considered from a top level view.

Class diagram is basically a graphical representation of the static view of the system and represents different aspects of the application. A collection of class diagrams represent the whole system.

The following points should be remembered while drawing a class diagram −

- The name of the class diagram should be meaningful to describe the aspect of the system.
- Each element and their relationships should be identified in advance.
- Responsibility (attributes and methods) of each class should be clearly identified
- For each class, minimum number of properties should be specified, as unnecessary properties will make the diagram complicated.
- Use notes whenever required to describe some aspect of the diagram. At the end of the drawing it should be understandable to the developer/coder.

- Finally, before making the final version, the diagram should be drawn on plain paper and reworked as many times as possible to make it correct.

Usage of Class Diagram:

Class diagram is a static diagram and it is used to model the static view of a system. The static view describes the vocabulary of the system.

Class diagram is also considered as the foundation for component and deployment diagrams. Class diagrams are not only used to visualize the static view of the system but they are also used to construct the executable code for forward and reverse engineering of any system.

Generally, UML diagrams are not directly mapped with any object-oriented programming languages but the class diagram is an exception.
Class diagram clearly shows the mapping with object-oriented languages such as Java, C++, etc. From practical experience, class diagram is generally used for construction purpose.

Class diagrams are used for −

- Describing the static view of the system.
- Showing the collaboration among the elements of the static view.
- Describing the functionalities performed by the system.
- Construction of software applications using object oriented languages.

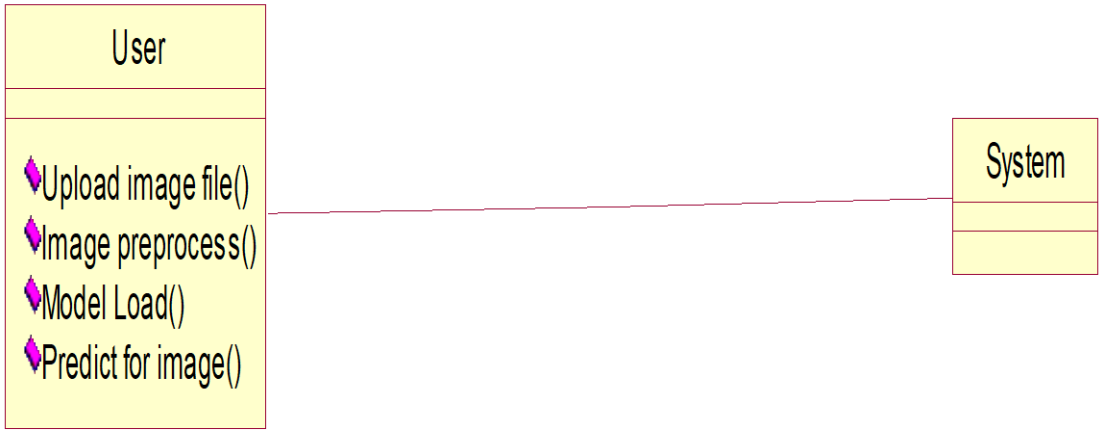

Figure 4.2: Class diagram

4.2.3 SEQUENCE DIAGRAM

A sequence diagram simply depicts interaction between objects in a sequential order i.e. the order in which these interactions take place. We can also use the terms event diagrams or event scenarios to refer to a sequence diagram. Sequence diagrams describe how and in what order the objects in a system function. These diagrams are widely used by businessmen and software developers to document and understand requirements for new and existing systems. Sequence diagrams emphasize on time sequence of messages and are typically associated with use case realizations in the logical view of the system under development. Sequence diagrams are sometimes called **event diagrams** or **event scenarios**.

Purpose of Sequence Diagram:

The purpose of sequence diagrams is to visualize the interactive behavior of the system. Visualizing the interaction is a difficult task. Hence, the solution is to use different types of models to capture the different aspects of the interaction.
Sequence diagrams are used to capture the dynamic nature but from a different angle.

The purpose of sequence diagram is −

- To capture the dynamic behaviour of a system.
- To describe the message flow in the system.
- To describe the structural organization of the objects.
- To describe the interaction among objects.

Guidelines to draw Sequence diagram:

The purpose of sequence diagrams is to capture the dynamic aspect of a system. So to capture the dynamic aspect, we need to understand what a dynamic aspect is and how it is visualized. Dynamic aspect can be defined as the snapshot of the running system at a particular moment.

The sequence diagram captures the time sequence of the message flow from one object to another.
Following things are to be identified clearly before drawing the sequence diagram --

- Objects taking part in the interaction.
- Message flows among the objects.

- The sequence in which the messages are flowing.

Usage of Sequence Diagram:

To understand the practical application, we need to understand the basic nature of sequence diagram.

The main purpose of sequence diagrams is they are used to capture the dynamic behavior of a system. However, the specific purpose is more important to clarify and understand.

Sequence diagrams are used to capture the order of messages flowing from one object to another. A single diagram is not sufficient to describe the dynamic aspect of an entire system, so a set of diagrams are used to capture it as a whole.

Sequence diagrams are used when we want to understand the message flow. Message flow means the sequence of control flow from one object to another.

Sequence diagrams can be used −

- To model the flow of control by time sequence.
- For forward engineering.
- For reverse engineering.

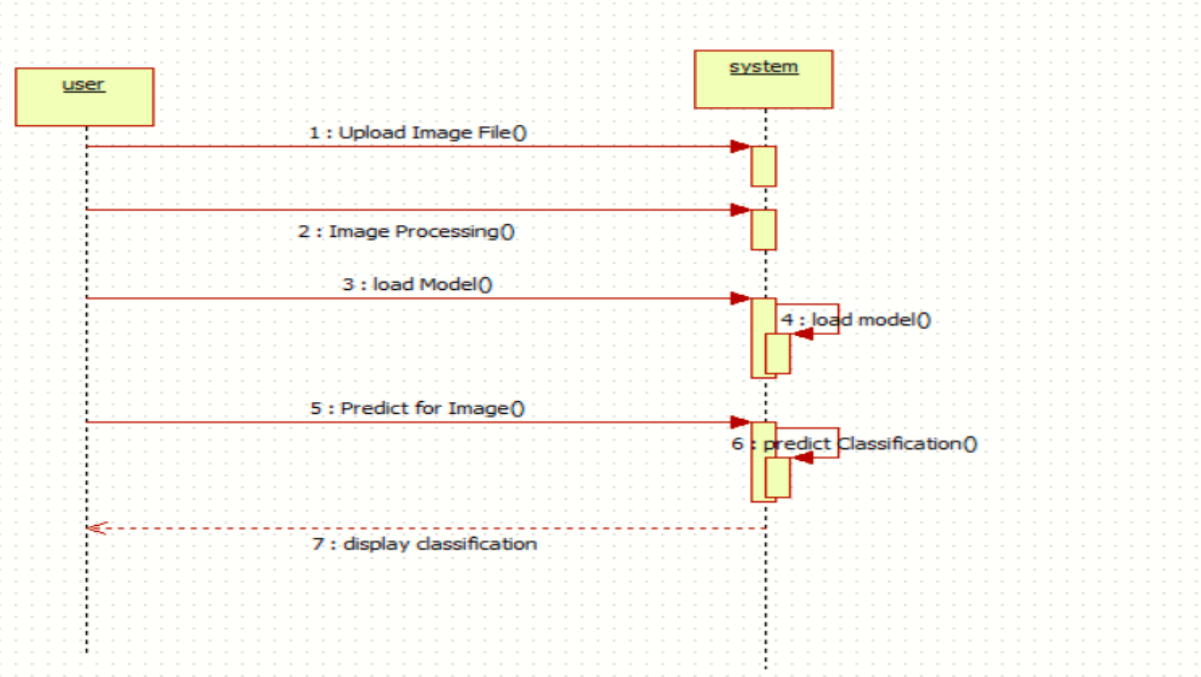

Figure 4.3: Sequence diagram

4.2.4 COLLABORATION DIAGRAM

A collaboration diagram, also known as a communication diagram, is an illustration of the relationships and interactions among software objects in the Unified Modelling Language (UML). These diagrams can be used to portray the dynamic behaviour of a particular use case and define the role of each object.

Collaboration diagrams are created by first identifying the structural elements required to carry out the functionality of an interaction. A model is then built using the relationships between those elements.

In the collaboration diagram, the method call sequence is indicated by some numbering technique. The number indicates how the methods are called one after another. Collaboration diagram emphasizes on the structural organization of the objects that send and receive messages.

Purpose of Collaboration Diagram:

The purpose of collaboration diagrams is to visualize the interactive behavior of the system. Visualizing the interaction is a difficult task. Hence, the solution is to use different types of models to capture the different aspects of the interaction.
Collaboration diagrams are used to capture the dynamic nature but from a different angle.

The purpose of collaboration diagram is −

- To capture the dynamic behaviour of a system.
- To describe the message flow in the system.
- To describe the structural organization of the objects.
- To describe the interaction among objects.

Guidelines to draw Collaboration diagram:

The purpose of collaboration diagrams is to capture the dynamic aspect of a system. So to capture the dynamic aspect, we need to understand what a dynamic aspect is and how it is visualized. Dynamic aspect can be defined as the snapshot of the running system at a particular moment.

The collaboration diagram describes the organization of objects in a system taking part in the message flow.

Following things are to be identified clearly before drawing the sequence diagram --

- Objects taking part in the interaction.
- Message flows among the objects.
- Object organization.

Usage of Collaboration Diagram:

To understand the practical application, we need to understand the basic nature of collaboration diagram.

The main purpose of the collaboration diagrams is they are used to capture the dynamic behavior of a system. However, the specific purpose is more important to clarify and understand.

Collaboration diagrams are used to describe the structural organization of the objects taking part in the interaction. A single diagram is not sufficient to describe the dynamic aspect of an entire system, so a set of diagrams are used to capture it as a whole.

Collaboration diagrams are used when we want to understand the structural organization. Structural organization means the visual organization of the elements in a system.

Collaboration diagrams can be used −

- To model the flow of control by structural organizations.
- For forward engineering.
- For reverse engineering.

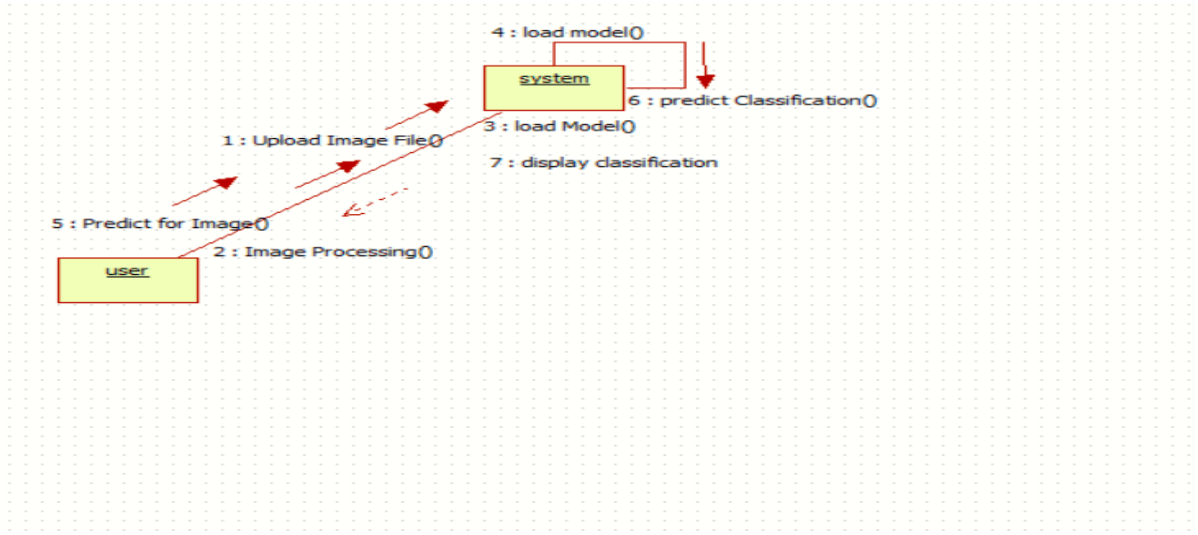

Figure 4.4: Collaboration diagram

5. IMPLEMENTATION AND RESULTS

5.1 INTRODUCTION

Functions are used for placing or storing the code which is to be repeated several times. For example, if we need same code, then we must have to write that code again and again. So in order to remove this we use functions.

Implementation is the stage where the theoretical design is turned into a working system. The most crucial stage in achieving a new successful system is giving confidence on the new system for the users that it will work efficiently and effectively.
The system can be implemented only after thorough testing is done and if it is found to work according to the specification.

It involves careful planning, investigation of the current system and its constraints on implementation, design of methods to achieve the change over an evaluation of change over methods apart from planning. Two major tasks for preparing the implementation are education and training of the users and testing of the system.

5.2 METHOD OF IMPLEMENTATION

The more complex the system being implemented, the more involved will be the systems analysis and design efforts required for implementation.
The implementation phase comprises of several activities. The required hardware and software acquisition is carried out. The system may require some software to be developed. For this, programs are written and tested. The user then change over to his new fully tested system and the old system is discontinued.

5.2.1 TECHNOLOGIES USED

The technologies that are used in the project are:

i. Deep Learning

Deep learning is an artificial intelligence (AI) function that imitates the workings of the human brain in processing data and creating patterns for use in decision making. Deep learning

is a subset of machine learning in artificial intelligence that has networks capable of learning unsupervised from data that is unstructured or unlabelled. Also known as deep neural learning or deep neural network.

Deep learning has evolved hand-in-hand with the digital era, which has brought about an explosion of data in all forms and from every region of the world. This data, known simply as big data, is drawn from sources like social media, internet search engines, e-commerce platforms, and online cinemas, among others. This enormous amount of data is readily accessible and can be shared through fintech applications like cloud computing.
However, the data, which normally is unstructured, is so vast that it could take decades for humans to comprehend it and extract relevant information. Companies realize the incredible potential that can result from unravelling this wealth of information and are increasingly adapting to AI systems for automated support.

Deep Learning vs. Machine Learning

One of the most common AI techniques used for processing big data is machine learning, a self-adaptive algorithm that gets increasingly better analysis and patterns with experience or with newly added data.

If a digital payments company wanted to detect the occurrence or potential for fraud in its system, it could employ machine learning tools for this purpose. The computational algorithm built into a computer model will process all transactions happening on the digital platform, find patterns in the data set, and point out any anomaly detected by the pattern.

Deep learning, a subset of machine learning, utilizes a hierarchical level of artificial neural networks to carry out the process of machine learning. The artificial neural networks are built like the human brain, with neuron nodes connected together like a web. While traditional programs build analysis with data in a linear way, the hierarchical function of deep learning systems enables machines to process data with a nonlinear approach.

VGG16(Visual Geometry Group)

VGG16 is a convolutional neural network model proposed by K. Simonyan and A. Zisserman from the University of Oxford in the paper “Very Deep Convolutional Networks for Large-Scale Image Recognition”.

It was one of the famous model submitted to ILSVRC-2014. It makes the improvement over AlexNet by replacing large kernel-sized filters (11 and 5 in the first and second

convolutional layer, respectively) with multiple 3×3 kernel-sized filters one after another. VGG16 was trained for weeks and was using NVIDIA Titan Black GPU's.

The input to cov1 layer is of fixed size 224 x 224 RGB image. The image is passed through a stack of convolutional (conv.) layers, where the filters were used with a very small receptive field: 3×3 (which is the smallest size to capture the notion of left/right, up/down, center). In one of the configurations, it also utilizes 1×1 convolution filters, which can be seen as a linear transformation of the input channels (followed by non-linearity). The convolution stride is fixed to 1 pixel; the spatial padding of conv. layer input is such that the spatial resolution is preserved after convolution, i.e. the padding is 1-pixel for 3×3 conv. layers. Spatial pooling is carried out by five max-pooling layers, which follow some of the conv. layers (not all the conv. layers are followed by max-pooling). Max-pooling is performed over a 2×2 pixel window, with stride 2

Examples of Deep Learning:

1. Customer experience

Machine learning is already used by many businesses to enhance the customer experience. Just a couple of examples include online self-service solutions and to create reliable workflows. There are already deep-learning models being used for chatbots, and as deep learning continues to mature, we can expect this to be an area deep learning will be used for many businesses.

2. Translations

Although automatic machine translation isn't new, deep learning is helping enhance automatic translation of text by using stacked networks of neural networks and allowing translations from images.

3. Adding color to black-and-white images and videos

What used to be a very time-consuming process where humans had to add color to black-and-white images and videos by hand can now be automatically done with deep-learning models.

4. Language recognition

Deep learning machines are beginning to differentiate dialects of a language. A machine decides that someone is speaking English and then engages an AI that is learning to tell the differences between dialects. Once the dialect is determined, another AI will step in

that specializes in that particular dialect. All of this happens without involvement from a human.

5. Autonomous vehicles

There's not just one AI model at work as an autonomous vehicle drives down the street. Some deep-learning models specialize in streets signs while others are trained to recognize pedestrians. As a car navigates down the road, it can be informed by up to millions of individual AI models that allow the car to act.

ii. Python:

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. It was created by Guido van Rossum during 1985- 1990. Like Perl, Python source code is also available under the GNU General Public License (GPL).

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

Some of the key advantages of learning Python:

- **Python is Interpreted** − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- **Python is Interactive** − You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
- **Python is Object-Oriented** − Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
- **Python is a Beginner's Language** − Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

Characteristics of Python:

Following are important characteristics of Python Programming −

- It supports functional and structured programming methods as well as OOP.

- It can be used as a scripting language or can be compiled to byte-code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- It supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

Applications of Python:

As mentioned before, Python is one of the most widely used language over the web. I'm going to list few of them here:

- **Easy-to-learn** − Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- **Easy-to-read** − Python code is more clearly defined and visible to the eyes.
- **Easy-to-maintain** − Python's source code is fairly easy-to-maintain.
- **A broad standard library** − Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- **Interactive Mode** − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- **Portable** − Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- **Extendable** − You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- **Databases** − Python provides interfaces to all major commercial databases.
- **GUI Programming** − Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
- **Scalable** − Python provides a better structure and support for large programs than shell scripting.

i. Python GUI - Tkinter:

Python offers multiple options for developing GUI (Graphical User Interface). Out of all the GUI methods, tkinter is the most commonly used method. It is a standard Python interface to the Tk GUI toolkit shipped with Python. Python with tkinter is the fastest and easiest way to create the GUI applications. Creating a GUI using tkinter is an easy task.

To create a Tkinter app:

1. Importing the module – Tkinter
2. Create the main window (container)
3. Add any number of widgets to the main window
4. Apply the event Trigger on the widgets.

Python Tkinter Geometry:

The Tkinter geometry specifies the method by using which, the widgets are represented on display.

The python Tkinter provides the following geometry methods.

1. The pack() method
2. The grid() method
3. The place() method

1. Python Tkinter pack() method:

The pack () widget is used to organize widget in the block. The positions widgets added to the python application using the pack () method can be controlled by using the various options specified in the method call.

However, the controls are less and widgets are generally added in the less organized manner. The syntax to use the pack() is given below.

Syntax:

widget.pack(options)

A list of possible options that can be passed in pack() method is given below:

- expand: If the expand is set to true, the widget expands to fill any space.
- fill: By default, the fill is set to NONE. However, we can set it to X or Y to determine whether the widget contains any extra space.
- size: it represents the side of the parent to which the widget is to be placed on the window.

2. Python Tkinter grid() method:

The grid() geometry manager organizes the widgets in the tabular form. We can specify the rows and columns as the options in the method call. We can also specify the column span (width) or rowspan(height) of a widget.

This is a more organized way to place the widgets to the python application. The syntax to use the grid() is given below.

Syntax:

widget.grid(options)

A list of possible options that can be passed inside the grid() method is given below:

- column: The column number in which the widget is to be placed. The leftmost column is represented by 0.
- columnspan: The width of the widget. It represents the number of columns up to which, the column is expanded.
- ipadx, ipady: It represents the number of pixels to pad the widget inside the widget and border.
- padx, pady: It represents the number of pixels to pad the widget outside the widget and border.
- row: The row number in which the widget is to be placed. The topmost row is represented by 0.
- rowspan: The height of the widget, i.e. the number of the row up to which the widget is expanded.

3. Python Tkinter place() method:

The place() geometry manager organizes the widgets to the specific x and y coordinates.

Syntax:

widget.place(options)

A list of possible options that can be passed inside the place() method is given below:

- anchor: It represents the exact position of the widget within the container. The default value (direction) is NW (the upper left corner)

- bordermode: The default value of the border type is INSIDE that refers to ignore the parent and inside the border. The other option is OUTSIDE.
- height, width: It refers to the height and width in pixels.
- relheight, relwidth: It is represented as the float between 0.0 and 1.0 indicating the fraction of the parent's height and width.
- relx, rely: It is represented as the float between 0.0 and 1.0 that is the offset in the horizontal and vertical direction.
- x, y: It refers to the horizontal and vertical offset in the pixels.

ii. Numpy:

NumPy is a module for Python. The name is an acronym for "Numeric Python" or "Numerical Python". It is pronounced ‘NUM-py’ or less often ‘NUM-pee’. It is an extension module for Python, mostly written in C. This makes sure that the precompiled mathematical and numerical functions and functionalities of Numpy guarantee great execution speed.

Numeric, the ancestor of NumPy, was developed by Jim Hugunin. Another package Numarray was also developed, having some additional functionalities. In 2005, Travis Oliphant created NumPy package by incorporating the features of Numarray into Numeric package. There are many contributors to this open source project.

Furthermore, NumPy enriches the programming language Python with powerful data structures, implementing multi-dimensional arrays and matrices. These data structures guarantee efficient calculations with matrices and arrays. The implementation is even aiming at huge matrices and arrays, better know under the heading of "big data". Besides that the module supplies a large library of high-level mathematical functions to operate on these matrices and arrays.

NumPy is based on two earlier Python modules dealing with arrays. One of these is Numeric. Numeric is like NumPy a Python module for high-performance, numeric computing, but it is obsolete nowadays. Another predecessor of NumPy is Numarray, which is a complete rewrite of Numeric but is deprecated as well. NumPy is a merger of those two, i.e. it is build on the code of Numeric and the features of Numarray.

NumPy is often used along with packages like SciPy (Scientific Python) and Mat−plotlib (plotting library). This combination is widely used as a replacement for MatLab, a popular platform for technical computing. However, Python alternative to MatLab is now seen as a more modern and complete programming language.

SciPy needs Numpy, as it is based on the data structures of Numpy and furthermore its basic creation and manipulation functions. It extends the capabilities of NumPy with further useful functions for minimization, regression, Fourier-transformation and many others.Both NumPy and SciPy are not part of a basic Python installation. They have to be installed after the Python installation.

Syntax:

import numpy as np

Operations of NumPy:

Using NumPy, a developer can perform the following operations −

- Mathematical and logical operations on arrays.
- Fourier transforms and routines for shape manipulation.
- Operations related to linear algebra. NumPy has in-built functions for linear algebra and random number generation.

Advantages of using NumPy:

- Open Source
- Array oriented computing
- Efficiently implemented multi-dimensional arrays
- Designed for scientific computation

NumPy Array:

Numpy array is a powerful N-dimensional array object which is in the form of rows and columns. We can initialize numpy arrays from nested Python lists and access it elements.

We use python numpy array instead of a list because of the below three reasons:

1. Less Memory
2. Fast
3. Convenient

Python NumPy Operations:

- **ndim:**

You can find the dimension of the array, whether it is a two-dimensional array or a single dimensional array.

- **itemsize:**

You can calculate the byte size of each element.

- **dtype:**

You can find the data type of the elements that are stored in an array. So, if you want to know the data type of a particular element, you can use 'dtype' function which will print the datatype along with the size.

- **reshape:**

Reshape is when you change the number of rows and columns which gives a new view to an object.

- **slicing:**

Slicing is basically extracting particular set of elements from an array. This slicing operation is pretty much similar to the one which is there in the list as well.

- **linspace:**

This is another operation in python numpy which returns evenly spaced numbers over a specified interval.

- **max/min:**

Next, we have some more operations in numpy such as to find the minimum, maximum as well the sum of the numpy array.

- **Square Root & Standard Deviation:**

There are various mathematical functions that can be performed using python numpy. You can find the square root, standard deviation of the array.

- **Addition Operation:**

You can perform more operations on numpy array i.e. addition, subtraction, multiplication and division of the two matrices.

- **Vertical & Horizontal Stacking:**

If you want to concatenate two arrays and not just add them, you can perform it using two ways – *vertical stacking* and *horizontal stacking*.

- **ravel:**

There is one more operation where you can convert one numpy array into a single column i.e. *ravel*.

iii. Matplotlib:

Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack. It was introduced by John Hunter in the year 2002.

Matplotlib was originally written by John D. Hunter, has an active development community, and is distributed under a BSD-style license. Michael Droettboom was nominated as matplotlib's lead developer shortly before John Hunter's death in August 2012, and further joined by Thomas Caswell.

One of the greatest benefits of visualization is that it allows us visual access to huge amounts of data in easily digestible visuals. Matplotlib consists of several plots like line, bar, scatter, histogram etc.

It is a very powerful plotting library useful for those working with Python and NumPy. And for making statistical interference, it becomes very necessary to visualize our data and

Matplotlib is the tool that can be very helpful for this purpose. It provides MATLAB like interface only difference is that it uses Python and is open source.

Matplotlib 2.0.x supports Python versions 2.7 through 3.6. Python 3 support started with Matplotlib 1.2. Matplotlib 1.4 is the last version to support Python 2.6. Matplotlib has pledged to not support Python 2 past 2020 by signing the Python 3 Statement.

There are several toolkits which are available that extend python matplotlib functionality. Some of them are separate downloads, others can be shipped with the matplotlib source code but have external dependencies.

- **Basemap**: It is a map plotting toolkit with various map projections, coastlines and political boundaries.
- **Cartopy**: It is a mapping library featuring object-oriented map projection definitions, and arbitrary point, line, polygon and image transformation capabilities.
- **Excel tools**: Matplotlib provides utilities for exchanging data with Microsoft Excel.
- **Mplot3d**: It is used for 3-D plots.
- **Natgrid**: It is an interface to the natgrid library for irregular gridding of the spaced data.

Basics:

- **Figure –** It is a whole figure which contains one or more than one axes or plots.
- **Axes –** A figure contains usually more than one axes (plots) and it may contain two or three in case of three-dimensional structure or objects. Each Axes has a title, an X – label, and Y –label.
- **Axis-** It takes care of generating graph limits
- **Artists-** Mostly they are tied to the axes, and whatever we see on the figure like text objects, line 2D objects, and collection objects.

Types of Plots:

There are various plots which can be created using python matplotlib. Some of them are listed below:

1. Bar Graph
2. Histogram
3. Scatter Plot

4. Area Plot
5. Pie Chart

1. Python Matplotlib: Bar Graph

First, let us understand why we need a bar graph. A bar graph uses bars to compare data among different categories. It is well suited when you want to measure the changes over a period of time. It can be represented horizontally or vertically. Also, the important thing to keep in mind is that longer the bar, greater is the value.

2. Python Matplotlib – Histogram

The main difference between a bar graph and a histogram is that the histograms are used to show a distribution whereas a bar chart is used to compare different entities. Histograms are useful when you have arrays or a very long list.

3. Python Matplotlib : Scatter Plot

Usually we need scatter plots in order to compare variables, for example, how much one variable is affected by another variable to build a relation out of it. The data is displayed as a collection of points, each having the value of one variable which determines the position on the horizontal axis and the value of other variable determines the position on the vertical axis.

4. Python Matplotlib : Area Plot

Area plots are pretty much similar to the line plot. They are also known as stack plots. These plots can be used to track changes over time for two or more related groups that make up one whole category.

5. Python Matplotlib : Pie Chart

A pie chart refers to a circular graph which is broken down into segments i.e. slices of pie. It is basically used to show the percentage or proportional data where each slice of pie represents a category.

Disadvantages:

Apart from these, python matplotlib has some disadvantages. Some of them are listed below:

- They are heavily reliant on other packages, such as NumPy.

- It only works for python, so it is hard or impossible to be used in languages other than python. (But it can be used from Julia via PyPlot package).

Advantages:

There are few advantages of Matplotlib when compared to Matlab and are as follows:

- Ability to use Python
- Free and open-source.

iv. Pandas:

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. The name Pandas is derived from the word Panel Data – an Econometrics from Multidimensional data.

Prior to Pandas, Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data — load, prepare, manipulate, model, and analyze.

Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

History:

Pandas was initially developed by Wes McKinney in 2008 while he was working at AQR Capital Management. He convinced the AQR to allow him to open source the Pandas. Another AQR employee, Chang She, joined as the second major contributor to the library in 2012. Over the time many versions of pandas have been released. The latest version of the pandas is 1.0.3.

Key Features of Pandas:

- Fast and efficient Data Frame object with default and customized indexing.
- Tools for loading data into in-memory data objects from different file formats.
- Data alignment and integrated handling of missing data.
- Reshaping and pivoting of date sets.
- Label-based slicing, indexing and sub setting of large data sets.

- Columns from a data structure can be deleted or inserted.
- Group by data for aggregation and transformations.
- High performance merging and joining of data.
- Time Series functionality.

Uses of Pandas:

Pandas has so many uses that it might make sense to list the things it can't do instead of what it can do.This tool is essentially your data's home. Through pandas, you get acquainted with your data by cleaning, transforming, and analyzing it.

For example, say you want to explore a dataset stored in a CSV on your computer. Pandas will extract the data from that CSV into a DataFrame — a table, basically — then let you do things like:

- Calculate statistics and answer questions about the data, like
    - What's the average, median, max, or min of each column?
    - Does column A correlate with column B?
    - What does the distribution of data in column C look like?
- Clean the data by doing things like removing missing values and filtering rows or columns by some criteria.
- Visualize the data with help from Matplotlib. Plot bars, lines, histograms, bubbles, and more.
- Store the cleaned, transformed data back into a CSV, other file or database.

Core components of pandas:

Pandas generally provide two data structure for manipulating data, they are:

i. Series
ii. DataFrame

i. Series:

Pandas Series is a one-dimensional labeled array capable of holding data of any type (integer, string, float, python objects, etc.). The axis labels are collectively called index. Pandas Series is nothing but a column in an excel sheet. Labels need not be unique but must be a hashable type. The object supports both integer and label-based indexing and provides a host of methods for performing operations involving the index.

Creating a Series:

In the real world, a Pandas Series will be created by loading the datasets from existing storage, storage can be SQL Database, CSV file, and Excel file. Pandas Series can be created from the lists, dictionary, and from a scalar value etc.

ii. DataFrame:

Pandas DataFrame is two-dimensional size-mutable, potentially heterogeneous tabular data structure with labeled axes (rows and columns). A Data frame is a two-dimensional data structure, i.e., data is aligned in a tabular fashion in rows and columns. Pandas DataFrame consists of three principal components, the data, rows, and columns.

Creating a DataFrame:

In the real world, a Pandas DataFrame will be created by loading the datasets from existing storage, storage can be SQL Database, CSV file, and Excel file. Pandas DataFrame can be created from the lists, dictionary, and from a list of dictionary etc.

Advantages:

- Fast and efficient for manipulating and analyzing data.
- Data from different file objects can be loaded.
- Easy handling of missing data (represented as NaN) in floating point as well as non-floating point data
- Size mutability: columns can be inserted and deleted from DataFrame and higher dimensional objects
- Data set merging and joining.
- Flexible reshaping and pivoting of data sets
- Provides time-series functionality.
- Powerful group by functionality for performing split-apply-combine operations on data sets.

Pandas in Data Science:

Pandas is generally used for data science because pandas is used in conjunction with other libraries that are used for data science. It is built on the top of the NumPy library which

means that a lot of structures of NumPy are used or replicated in Pandas. The data produced by Pandas is often used as input for plotting functions of Matplotlib, statistical analysis in SciPy, machine learning algorithm in Scikit-learn.

Pandas program can be run from any text editor but it is recommended to use Jupyter Notebook for this as Jupyter given the ability to execute code in a particular cell rather than executing the entire file. Jupyter also provides an easy way to visualize pandas dataframe and plots.

v. Sklearn:

Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistence interface in Python. This library, which is largely written in Python, is built upon NumPy, SciPy and Matplotlib.

Origin of Scikit-Learn:

It was originally called ***scikits.learn*** and was initially developed by David Cournapeau as a Google summer of code project in 2007. Later, in 2010, Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, and Vincent Michel, from FIRCA (French Institute for Research in Computer Science and Automation), took this project at another level and made the first public release (v0.1 beta) on 1st Feb. 2010.

Prerequisites:

Before we start using scikit-learn latest release, we require the following −

- Python (>=3.5)
- NumPy (>= 1.11.0)
- Scipy (>= 0.17.0)
- Joblib (>= 0.11)
- Matplotlib (>= 1.5.1) is required for Sklearn plotting capabilities.
- Pandas (>= 0.18.0) is required for some of the scikit-learn examples using data structure and analysis.

Features:

Rather than focusing on loading, manipulating and summarising data, Scikit-learn library is focused on modeling the data. Some of the most popular groups of models provided by Sklearn are as follows −

i. **Supervised Learning algorithms** − Almost all the popular supervised learning algorithms, like Linear Regression, Support Vector Machine (SVM), Decision Tree etc., are the part of scikit-learn.

ii. **Unsupervised Learning algorithms** − On the other hand, it also has all the popular unsupervised learning algorithms from clustering, factor analysis, PCA (Principal Component Analysis) to unsupervised neural networks.

iii. **Clustering** − This model is used for grouping unlabeled data.

iv. **Cross Validation** − It is used to check the accuracy of supervised models on unseen data.

v. **Dimensionality Reduction** − It is used for reducing the number of attributes in data which can be further used for summarisation, visualisation and feature selection.

vi. **Ensemble methods** − As name suggest, it is used for combining the predictions of multiple supervised models.

vii. **Feature extraction** − It is used to extract the features from data to define the attributes in image and text data.

viii. **Feature selection** − It is used to identify useful attributes to create supervised models.

ix. **Open Source** − It is open source library and also commercially usable under BSD license.

x. **Datasets**: for test datasets and for generating datasets with specific properties for investigating model behavior.

xi. **Parameter Tuning**: for getting the most out of supervised models.

xii. **Manifold Learning**: For summarizing and depicting complex multi-dimensional data.

Benefits:

The main benefits of scikit-learn are its free usage, ease of use, versatility, international online community support, and proper API documentation. Here are more details:

i. **Free:** Since scikit-learn is distributed under BSD license, it is free to use for anyone. In addition, with the minimal restrictions of its license, users wouldn't have to worry about legal limitations when designing their platforms and applications.

ii. **Easy to use:** Many research organizations and commercial industries have used scikit-learn in their operations and they all agree about how the module is easy to use. Because of that, they don't run to any problem when performing a variety of processes.

iii. **Versatile use** The system is a handy tool that can do a multitude of things such as creating neuroimages, identifying abusive cloud actions, predicting consumer behavior, etc. It is widely used by research and commercial organizations throughout the world, proof of its versatility and ease of use.

iv. **Backed by international community** Since its one-man mission origin, scikit-learn has come a long way and is being developed by numerous authors in INRIA headed by Fabian Pedregosa as well as by various independent contributors. Because of this, the module is always updated, with several releases each year. scikit-learn is also backed by an international online community that users can count on if ever they run into troubles or have queries.

v. **Properly documented** To help ensure that new and old users both get the assistance they require with regards to integrating scikit-learn with their platforms, as well as extensive, detailed API documentation accessible from their website, is provided. With this, users can seamlessly integrate the learning algorithms in their own platforms.

Steps to build a machine learning model using Sklearn:

Step 1: Load a dataset

A dataset is nothing but a collection of data. A dataset generally has two main components:

- **Features**: (also known as predictors, inputs, or attributes) they are simply the variables of our data. They can be more than one and hence represented by a **feature matrix** ('X' is a common notation to represent feature matrix). A list of all the feature names is termed as **feature names**.
- **Response**: (also known as the target, label, or output) This is the output variable depending on the feature variables. We generally have a single response column and it is represented by a **response vector** ('y' is a common notation to represent response vector). All the possible values taken by a response vector is termed as **target names**.

Step 2: Splitting the dataset

One important aspect of all machine learning models is to determine their accuracy. Now, in order to determine their accuracy, one can train the model using the given dataset and then predict the response values for the same dataset using that model and hence, find the accuracy of the model.

But this method has several flaws in it, like:

- Goal is to estimate likely performance of a model on an out-of-sample data.
- Maximizing training accuracy rewards overly complex models that won't necessarily generalize our model.
- Unnecessarily complex models may over-fit the training data.
- A better option is to split our data into two parts: first one for training our machine learning model, and second one for testing our model.

To summarize:

- Split the dataset into two pieces: a training set and a testing set.
- Train the model on the training set.
- Test the model on the testing set, and evaluate how well our model did.

Advantages of train/test split:

- Model can be trained and tested on different data than the one used for training.
- Response values are known for the test dataset, hence predictions can be evaluated.
- Testing accuracy is a better estimate than training accuracy of out-of-sample performance.

The **train_test_split** function takes several arguments which are explained below:

- **x, y**: These are the feature matrix and response vector which need to be splitted.
- **test_size**: It is the ratio of test data to the given data. For example, setting test_size = 0.4 for 150 rows of X produces test data of 150 x 0.4 = 60 rows.
- **random_state**: If you use random_state = some_number, then you can guarantee that your split will be always the same. This is useful if you want reproducible results, for

example in testing for consistency in the documentation (so that everybody can see the same numbers).

Step 3: Training the model

Now, it's time to train some prediction-model using our dataset. Scikit-learn provide a wide range of machine learning algorithms which have a unified/consistent interface for fitting, predicting accuracy, etc.

Advantages:

- Scikit Learn provides a bunch of genuinely useful utilities for splitting data, computing common statistics, and doing even not-so-common matrix operations.
- Scikit Learn has good documentation, and a clean, mostly consistent API.
- Integrates well with Numpy.

Disadvantages:

- The model API provided by Scikit Learn doesn't have a lot of flexibility, and some models such as Random Forests are known to have a non-standard or buggy implementations (biasing the feature importances in this case).
- Scikit Learn doesn't use hardware acceleration making it slow at times; especially for training models.

5.3 EXPLANATIONS OF KEY FUNCTIONS

Source Code:

```
from tkinter import messagebox
from tkinter import *
from tkinter import simpledialog
import tkinter
from tkinter import filedialog
from tkinter.filedialog import askopenfilename
from keras.models import load_model
import cv2
import numpy as np
from PIL import Image

main = tkinter.Tk()
main.title("Obstacle Detection")
main.geometry("1300x1200")
iFlag = False
pre = False
def upload():

    global filename
    text.delete('1.0', END)
    filename = askopenfilename(initialdir = "D:\DataSet")
    pathlabel.config(text=filename)
    if filename.lower().endswith(('.png', '.jpg', '.jpeg')):
        text.insert(END,"Dataset loaded\n\n")
    else :
        iFlag = True;
        text.insert(END,"Incorrect Input Format")

def imagePreprocess():
    if not iFlag :
        global pre
        pre = True
        global filename
        global img4
        image=cv2.imread(filename)
        cv2.imshow("Image", image)
        img3 = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
        img3 = cv2.resize(img3,(224,224))
        img4= np.reshape(img3,[1,224,224,3])
        text.insert(END,"Image Preporcess Done\n\n")
    else :
        text.insert(END,"Incorrect Input Format")

def loadmodel():
    if not iFlag :
```

```
        if pre == True:
            global model
            model=load_model('mymodel_2.h5')
            text.insert(END,"trained Model loaded")
        else :
            text.insert(END,"Preprocess image before uploading")
    else :
        text.insert(END,"Incorrect Input Format")

def predict():
    if not iFlag:
        if pre == True:
            global img4
            global model
            classes=['reg_-45', 'rgb_-90', 'rgb_0', 'rgb_45', 'rgb_90']
            pred = model.predict_classes(img4)
            text.insert(END," Predicted class for Image: "+str(classes[pred[0]])+"\n\n")
        else :
            text.insert(END,"Preprocess image before uploading")
    else :
        text.insert(END,"Incorrect Input Format")

font = ('times', 16, 'bold')
title = Label(main, text='Obstacle-Avoidance Algorithm Using Deep Learning.')
title.config(bg='sky blue', fg='black')
title.config(font=font)
title.config(height=3, width=120)
title.place(x=0,y=5)

font1 = ('times', 14, 'bold')
upload = Button(main, text="Upload Image File", command=upload)
upload.place(x=250,y=690)
upload.config(font=font1)

pathlabel = Label(main)
pathlabel.config(bg='dark orchid', fg='white')
pathlabel.config(font=font1)
pathlabel.place(x=50,y=750)

imp= Button(main, text="Image Preprocess", command=imagePreprocess)
imp.place(x=440,y=690)
imp.config(font=font1)

ml = Button(main, text="Model Load", command=loadmodel)
ml.place(x=650,y=690)
ml.config(font=font1)

pt = Button(main, text="Predict For Image", command=predict)
pt.place(x=800,y=690)
pt.config(font=font1)
```

```
font1 = ('times', 12, 'bold')
text=Text(main,height=30,width=80)
scroll=Scrollbar(text)
text.configure(yscrollcommand=scroll.set)
text.place(x=300,y=100)
text.config(font=font1)

main.config(bg='snow')
main.mainloop()

from keras.applications import VGG16

vgg_conv = VGG16(weights='imagenet', include_top=False, input_shape=(224, 224, 3))
from keras.preprocessing.image import ImageDataGenerator
for layer in vgg_conv.layers:
  layer.trainable = False
from keras import models
from keras import layers
from keras import optimizers

# Create the model
model = models.Sequential()

# Add the vgg convolutional base model
model.add(vgg_conv)
model.add(layers.Flatten())
model.add(layers.Dense(5, activation='softmax'))
model.summary()
train_dir = 'D:\DataSet'
validation_dir = 'D:\DataSet'
train_datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=20,
    height_shift_range=0.2,
    brightness_range=[0.3,1.0],
    fill_mode='nearest',
    validation_split=0.2
)

train_generator = train_datagen.flow_from_directory(
    train_dir,
    target_size=(224, 224),
    batch_size=32,
    subset='training', # set as training data
    class_mode='categorical')

validation_generator = train_datagen.flow_from_directory(
    train_dir, # same directory as training data
    target_size=(224, 224),
```

```
    batch_size=32,
    subset='validation',
    class_mode='categorical')
print(train_generator.class_indices)

from keras import optimizers

adam = optimizers.Adam()
model.compile(loss='categorical_crossentropy',
              optimizer=adam,
              metrics=['accuracy'])
from datetime import datetime
from keras.callbacks import ModelCheckpoint

metric = 'val_accuracy'
checkpoint = ModelCheckpoint(filepath='mymodel_2.h5',monitor='accuracy',
                             verbose=2, save_best_only=True)

callbacks = [checkpoint]

start = datetime.now()

history = model.fit_generator(
      train_generator,
      steps_per_epoch=train_generator.samples/train_generator.batch_size ,
      epochs=30,callbacks=callbacks,
      verbose=2)

duration = datetime.now() - start
print("Training completed in time: ", duration)
```

5.4 OUTPUT SCREENS

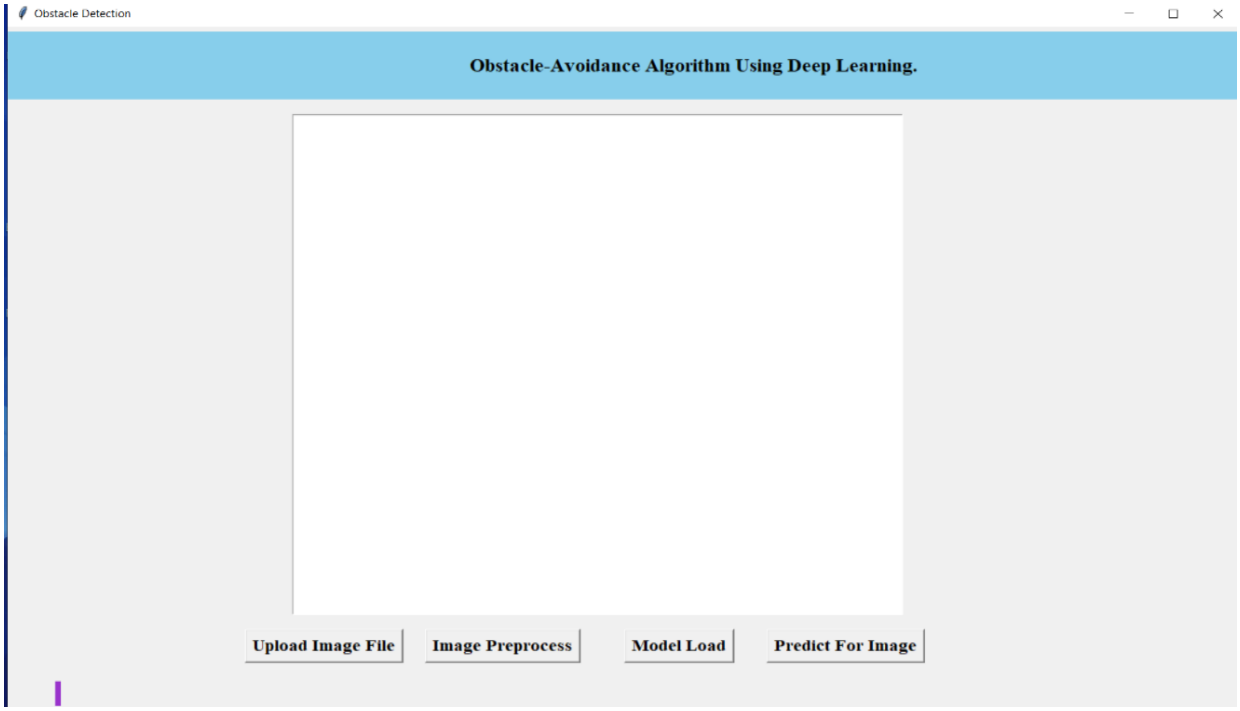

Figure 5.1: opening screen

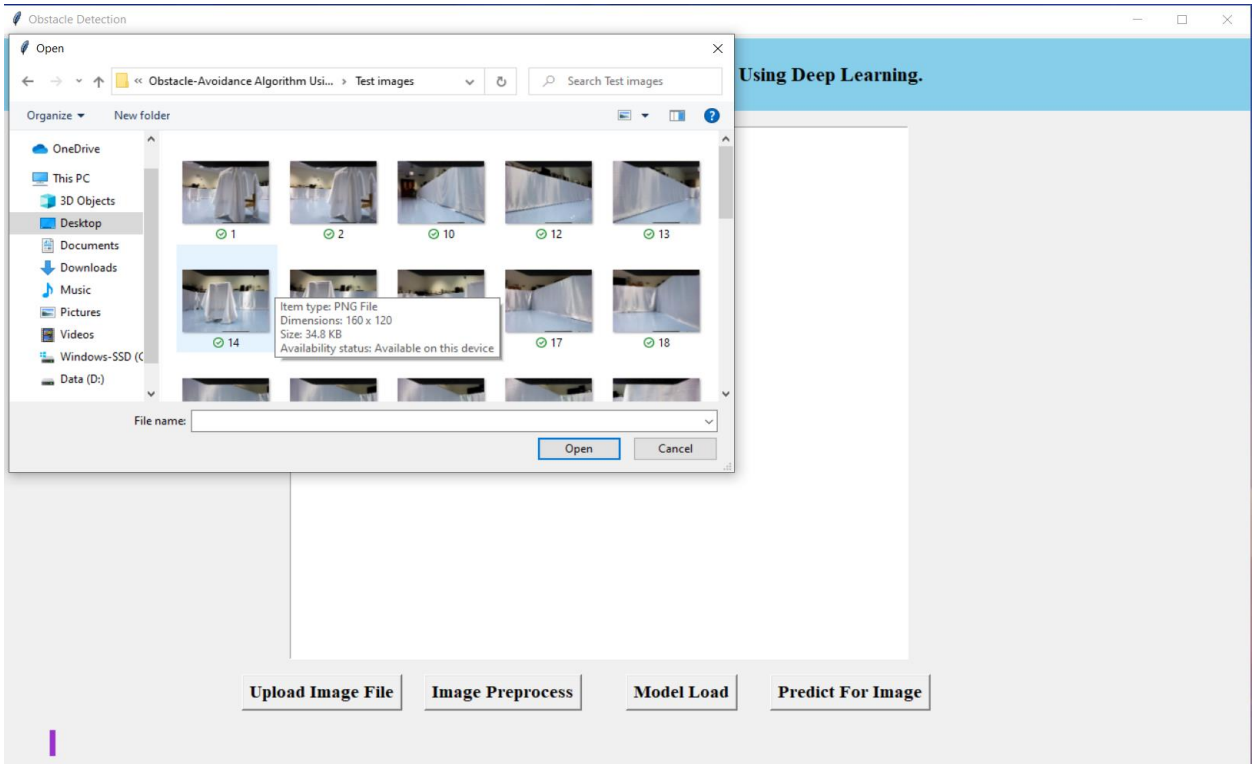

Figure 5.2:loading image

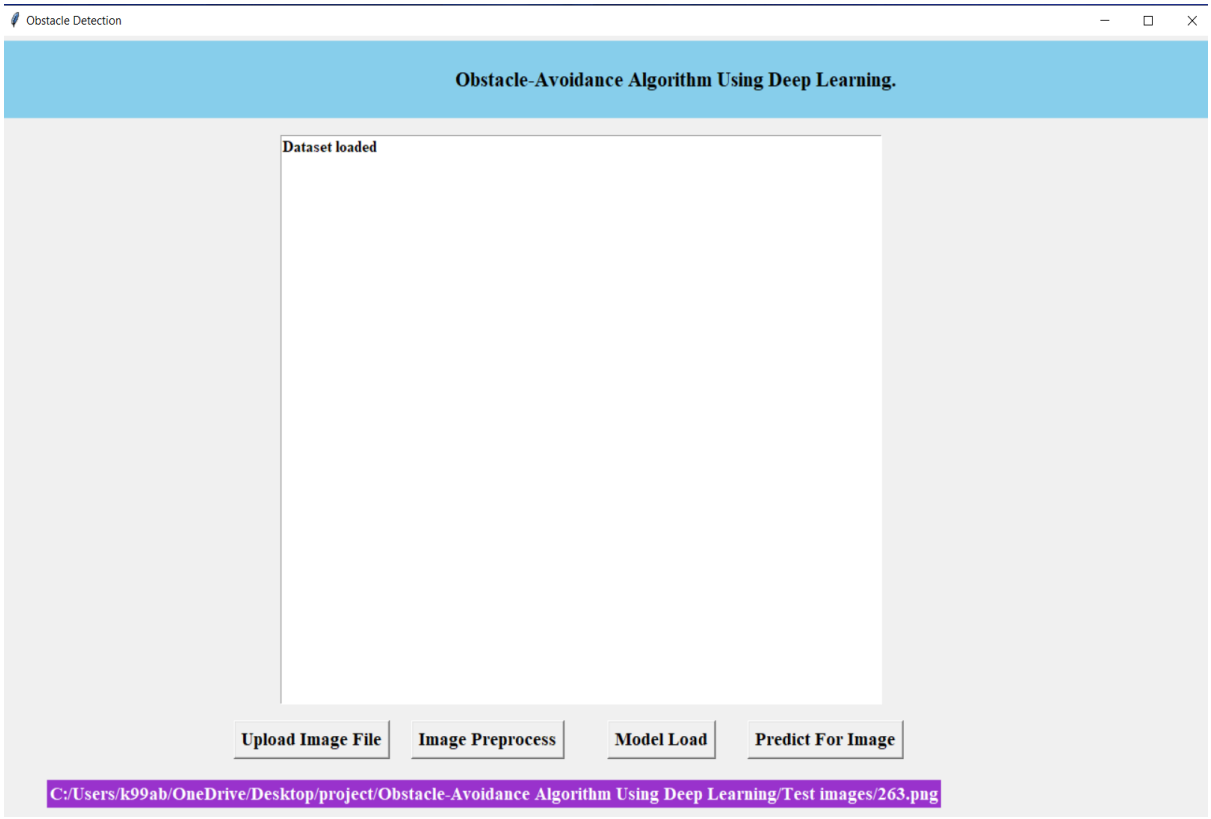

Figure 5.3: Data set loaded

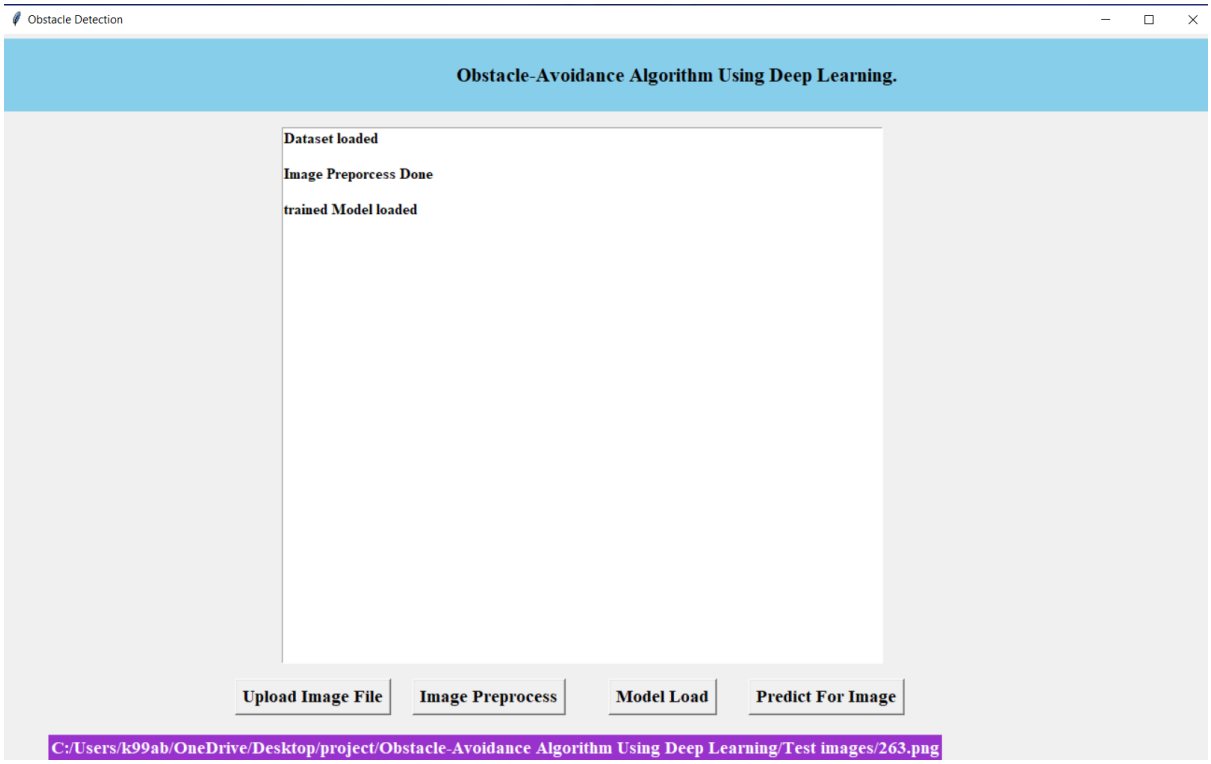

Figure 5.4: Model trained

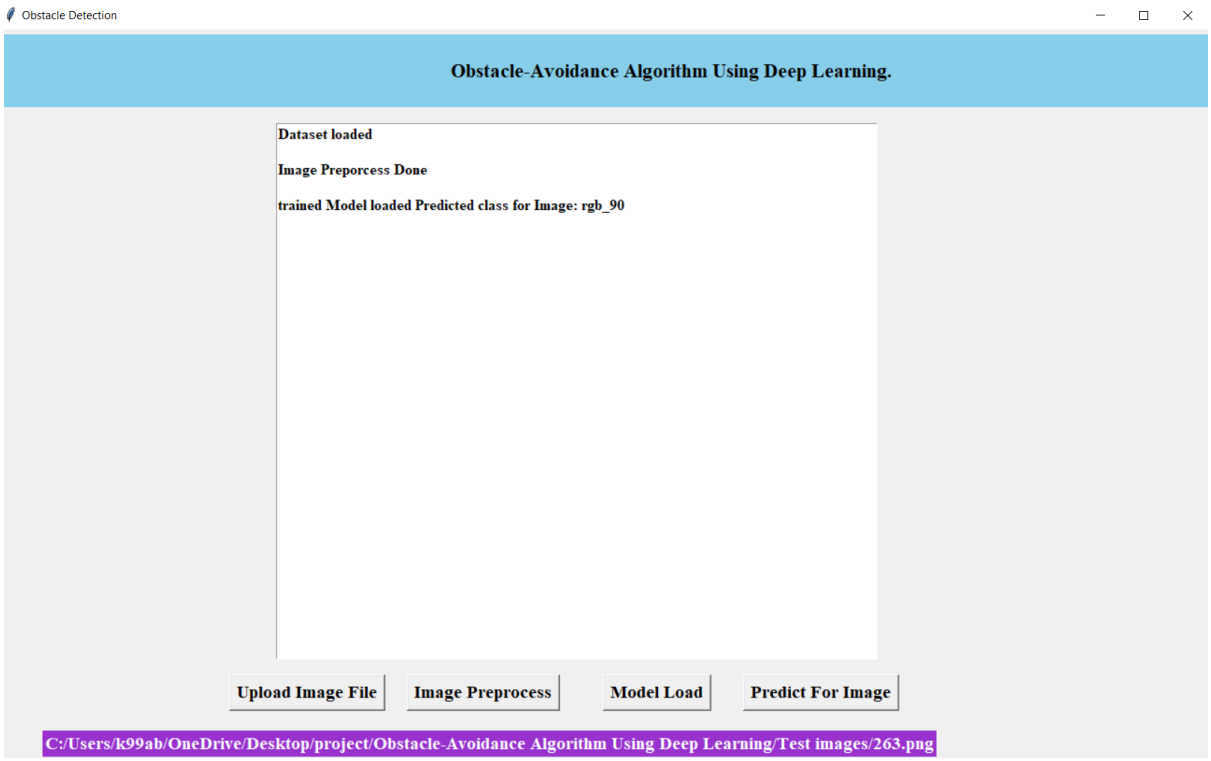

Figure 5.5: Final output

6.TESTING

6.1 INTRODUCTION

Software Testing is defined as an activity to check whether the actual results match the expected results and to ensure that the software system is Defect free. It involves execution of a software component or system component to evaluate one or more properties of interest. Software testing also helps to identify errors, gaps or missing requirements in contrary to the actual requirements. It can be either done manually or using automated tools. Some prefer saying Software testing as a White Box and Black Box Testing.

In simple terms, Software Testing means Verification of Application Under Test (AUT). Software testing is a critical element of software quality and assurance and represents ultimate review of specifications, design and coding. Testing is an exposure of the system to trial input to see whether it produces correct output.

The process of software testing aims not only at finding faults in the existing software but also at finding measures to improve the software in terms of efficiency, accuracy and usability. It mainly aims at measuring specification, functionality and performance of a software program or application.

Software Testing can be done in two ways:

1. **Verification:** It refers to the set of tasks that ensure that software correctly implements a specific function.
2. **Validation:** It refers to a different set of tasks that ensure that the software that has been built is traceable to customer requirements.

Verification: "Are we building the product right?"
Validation: "Are we building the right product?"

Importance of Software Testing:

The importance of software testing is imperative. Software Testing is important because of the following reasons:

1. Software Testing points out the defects and errors that were made during the development phases. It looks for any mistake made by the programmer during the implementation phase of the software.
2. It ensures that the customer finds the organization reliable and their satisfaction in the application is maintained. Sometimes contracts include monetary penalties with respect to the timeline and quality of the product and software testing prevent monetary losses.
3. It also ensures the Quality of the product. Quality product delivered to the customers helps in gaining their confidence. It makes sure that the software application requires lower maintenance cost and results in more accurate, consistent and reliable results.
4. Users are not inclined to use software that has bugs. They may not adopt software if they are not happy with the stability of the application. Testing is important for the product to stay in business.
5. It's important to ensure that the application should not result in any failures because it can be very expensive in the future or in the later stages of the development.

Applications of Software Testing:

- **Cost Effective Development** - Early testing saves both time and cost in many aspects, however reducing the cost without testing may result in improper design of a software application rendering the product useless.
- **Product Improvement** - During the SDLC phases, testing is never a time-consuming process. However diagnosing and fixing the errors identified during proper testing is a time-consuming but productive activity.
- **Test Automation** - Test Automation reduces the testing time, but it is not possible to start test automation at any time during software development. Test automaton should be started when the software has been manually tested and is stable to some extent. Moreover, test automation can never be used if requirements keep changing.
- **Quality Check** - Software testing helps in determining following set of properties of any software such as
    - Functionality
    - Reliability
    - Usability
    - Efficiency
    - Maintainability
    - Portability

i.TYPES OF SOFTWARE TESTING:

Software Testing can be broadly classified into two types:

i. Manual Testing:

Manual testing is a software testing process in which test cases are executed manually without using any automated tool. All test cases executed by the tester manually according to the end user's perspective. It ensures whether the application is working, as mentioned in the requirement document or not. Test cases are planned and implemented to complete almost 100 percent of the software application. Test case reports are also generated manually.

Manual Testing is one of the most fundamental testing processes as it can find both visible and hidden defects of the software. The difference between expected output and output, given by the software, is defined as a defect. The developer fixed the defects and handed it to the tester for retesting.

Manual testing is mandatory for every newly developed software before automated testing. This testing requires great efforts and time, but it gives the surety of bug-free software. Manual Testing requires knowledge of manual testing techniques but not of any automated testing tool.

Types of Manual Testing:

There are various methods used for manual testing. Each technique is used according to its testing criteria. Types of manual testing are given below:

- White Box Testing
- Black Box Testing

Advantages of Manual Testing:

- It does not require programming knowledge while using the Black box method
- It is used to test dynamically changing GUI design.
- Tester interacts with software as a real user so that they are able to discover usability and user interface issues.
- It ensures that the software is a hundred percent bug-free.
- It is cost-effective.

- Easy to learn for new testers.

Disadvantages of Manual Testing:

- It requires a large number of human resources.
- It is very time-consuming.
- Tester develops test cases based on their skills and experience. There is no evidence that they have covered all functions or not.
- Test cases cannot be used again. Need to develop separate test cases for each new software.
- It does not provide testing on all aspects of testing.
- Since two teams work together, sometimes it is difficult to understand each other's motives, it can mislead the process.

ii. Automation Testing:

Automation testing, which is also known as Test Automation, is when the tester writes scripts and uses another software to test the product. This process involves automation of a manual process. Automation Testing is used to re-run the test scenarios that were performed manually, quickly, and repeatedly.

Apart from regression testing, automation testing is also used to test the application from load, performance, and stress point of view. It increases the test coverage, improves accuracy, and saves time and money in comparison to manual testing.

Advantages of Automation Testing:

- Automation testing takes less time than manual testing.
- A tester can test the response of the software if the execution of the same operation is repeated several times.
- Automation Testing provides re-usability of test cases on testing of different versions of the same software.
- Automation testing is reliable as it eliminates hidden errors by executing test cases again in the same way.
- Automation Testing is comprehensive as test cases cover each and every feature of the application.

- It does not require many human resources, instead of writing test cases and testing them manually, they need an automation testing engineer to run them.
- The cost of automation testing is less than manual testing because it requires a few human resources.

Disadvantages of Automation Testing:

- Automation Testing requires high-level skilled testers.
- It requires high-quality testing tools.
- When it encounters an unsuccessful test case, the analysis of the whole event is complicated.
- Test maintenance is expensive because high fee license testing equipment is necessary.
- Debugging is mandatory if a less effective error has not been solved, it can lead to fatal results.

ii.TESTING ACTIVITIES:

Software level testing can be majorly classified into 4 levels:

1. Unit Testing
2. Integration Testing
3. System Testing
4. Acceptance Testing

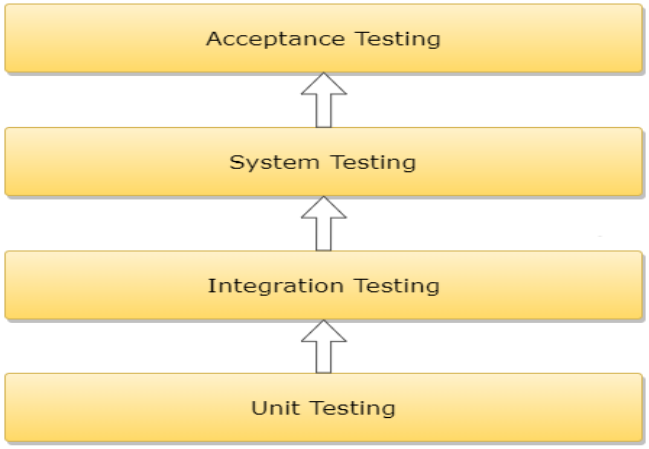

Figure 6.1: Levels of Testing

i. Unit Testing:

Unit Testing is a software testing technique by means of which individual units of software i.e. group of computer program modules, usage procedures and operating procedures are tested

to determine whether they are suitable for use or not. It is a testing method using which every independent module is tested to determine if there are any issue by the developer himself. It is correlated with functional correctness of the independent modules.

Unit Testing is defined as a type of software testing where individual components of a software are tested. Unit Testing of software product is carried out during the development of an application. An individual component may be either an individual function or a procedure. Unit Testing is typically performed by the developer.

Unit testing focuses on the building blocks of the software system, that is, objects and subsystems. They are three motivations behind focusing on components. First, unit testing reduces the complexity of the overall test activities, allowing us to focus on smaller units of the system. Unit testing makes it easier to pinpoint and correct faults given that few computers are involved in this test. Unit testing allows parallelism in the testing activities; that is each component can be tested independently of one another.

The specific candidates for unit testing are chosen from the object model and the system decomposition of the system. In principle, all the objects developed during the development process should be tested.

Objective of Unit testing:

The objective of Unit Testing is:

1. To isolate a section of code.
2. To verify the correctness of code.
3. To test every function and procedure.
4. To fix bug early in development cycle and to save costs.
5. To help the developers to understand the code base and enable them to make changes quickly.
6. To help for code reuse.

Advantages:

- Reduces Cost of Testing as defects are captured in very early phase.
- Unit Tests, when integrated with build gives the quality of the build as well

- Unit Testing allows developers to learn what functionality is provided by a unit and how to use it to gain a basic understanding of the unit API.
- Unit testing allows the programmer to refine code and make sure the module works properly.
- Unit testing enables to test parts of the project without waiting for others to be completed.

ii. Integration Testing:

Integration testing is the second level of the software testing process comes after unit testing. In this testing, units or individual components of the software are tested in a group. The focus of the integration testing level is to expose defects at the time of interaction between integrated components or units.

Unit testing uses modules for testing purpose, and these modules are combined and tested in integration testing. The Software is developed with a number of software modules that are coded by different coders or programmers. The goal of integration testing is to check the correctness of communication among all the modules.

In integration testing, testers test the interfaces between the different modules. These modules combine together to form a bigger component or the system. Hence, it becomes very crucial to validate their behavior when they work together. Apart from the interfaces, they also test the integrated components. Integration testing is the next level of testing after unit testing. Testers do it after completion of the unit testing phase. Integration testing techniques can be a white box or black box depending on the project requirements.

Objectives of Integration Testing:

Integration testing reduces the risk of finding the defects in integrated components in the System testing phase. Integration defects can be complex to fix and they can be time-consuming as well. Finding them early in the cycle eliminates the risk of making too many changes at the System testing phase. As each of the integrating components has been tested in the integration phase, the System testing can focus on end to end journeys and user-specific flows.

- Reducing risk by testing integrating components as they become available.

- Verify whether the functional and non-functional behaviours of the interfaces are designed as per the specification.
- To build confidence in the quality of the interfaces.
- To find defects in the components, system or in the interfaces.
- Prevents defects from escaping to higher test levels of testing i.e System testing.

Guidelines for Integration Testing:

- First, determine the test case strategy through which executable test cases can be prepared according to test data.
- Examine the structure and architecture of the application and identify the crucial modules to test them first.
- Design test cases to verify each interface in detail.
- Choose input data for test case execution. Input data plays a significant role in testing.
- Fix defects and retest.

Advantages of Integration testing:

- It helps to find defects from links and interfaces between the modules.
- It boosts the confidence level of the team in the product as it validates the group of modules together.
- Integration tests run faster than the end to end test scenarios.
- It results in higher code coverage.
- It starts in the early stages when the entire module may not be ready. Hence, it avoids the bugs getting into the system.

iii. System Testing:

System Testing is a type of software testing that is performed on a complete integrated system to evaluate the compliance of the system with the corresponding requirements.

In other words, System Testing means testing the system as a whole. All the modules/components are integrated in order to verify if the system works as expected or not. System Testing is done after Integration Testing. This plays an important role in delivering a high-quality product.

The purpose of a system test is to evaluate the end-to-end system specifications. Usually, the software is only one element of a larger computer-based system. Ultimately, the software is interfaced with other software/hardware systems. System Testing is actually a series of different tests whose sole purpose is to exercise the full computer-based system.

In system testing, integration testing passed components are taken as input. The goal of integration testing is to detect any irregularity between the units that are integrated together. System testing detects within both the integrated units and the whole system. The result of system testing is the observed behavior of a component or a system when it is tested.

System Testing is carried out on the whole system in the context of either system requirement specifications or functional requirement specifications or in the context of both. System testing tests the design and behavior of the system and also the expectations of the customer.

Objectives of System Testing:

The primary objectives of System testing are as below:

- One of the primary objectives of System testing is to reduce risk. Even after individual testing of components, risk of how they will all come together to form a complete System still exists. System testing eliminates this risk by ensuring that it will function as per customer requirements.
- System testing must verify whether the design of the functional and non-functional behaviors of the system is as per the customer's specifications.
- Validate that the system is complete and will work as expected.
- System testing aims to build confidence in the quality of the system as a whole.
- System testing also aims to find defects and to prevent defects from escaping to higher test levels or production. Additionally, it is the only phase that occurs on the full System just before the User Acceptance testing. So it's critical to find all the possible defects at this stage, and they don't leak to production.
- System Testing results are used by stakeholders to make release decisions. The Entry criteria for User Acceptance testing is the basis completion of System Testing. System testing may also adhere to legal or regulatory requirements or standards.

Guidelines for System Testing:

- The very first step is to create a Test Plan.
- Create System Test Cases and test scripts.
- Prepare the test data required for this testing.
- Execute the system test cases and script.
- Report the bugs. Re-testing the bugs once fixed.
- Regression testing to verify the impact of the change in the code.
- Repetition of the testing cycle until the system is ready to be deployed.
- Sign off from the testing team.

Advantages of System Testing:

- It covers a complete end to end software testing.
- The business requirements and system software architecture are both tested in system testing.
- Appropriate system testing help in relieving after production goes live issues and bugs.
- System testing is led in a situation like a production condition or some of the time it is finished with production parallel test condition where the same data input is feed to the exiting framework and new framework to look at the differences in functionalities removed and added. This causes the client to understand the new framework better and feel great with new functionalities included or existing functionalities revised or removed.

iv. Acceptance Testing:

Acceptance Testing is a method of software testing where a system is tested for acceptability. The major aim of this test is to evaluate the compliance of the system with the business requirements and assess whether it is acceptable for delivery or not.

The standard definition of Acceptance testing is given as, “It is a formal testing according to user needs, requirements and business processes conducted to determine whether a system satisfies the acceptance criteria or not and to enable the users, customers or other authorized entities to determine whether to accept the system or not.”

Acceptance Testing is the last phase of software testing performed after System Testing and before making the system available for actual use.

The main goal behind acceptance testing is to check whether the developed software product passes the acceptance norms defined on the basis of user and business requirements, so as to declare it acceptable or non-acceptable for its use by the users. Acceptance testing is one of the last types of software testing performed over a software or application. It is conducted by a pool of targeted users to ensure the readiness and quality of the system from user's perspective, which allows the team to meet their needs and expectations.

Objectives of Acceptance Testing:

Following are the three major objectives of Acceptance Testing:

- Confirm that the system meets the agreed-upon criteria.
- Identify and resolve discrepancies, if there are any.
- Determine the readiness of the system for cut-over to live operations. The final acceptance of a system for deployment is conditioned upon the outcome of the acceptance testing. The acceptance test team produces an acceptance test report which outlines the acceptance conditions.

Advantages of Acceptance Testing:

Acceptance testing has the following benefits, complementing those which can be obtained from unit tests:

- Encouraging closer collaboration between developers on the one hand and customers, users or domain experts on the other, as they entail that business requirements should be expressed
- Providing a clear and unambiguous “contract” between customers and developers; a product which passes acceptance tests will be considered adequate (though customers and developers might refine existing tests or suggest new ones as necessary).
- Decreasing the chance and severity both of new defects and regressions (defects impairing functionality previously reviewed and declared acceptable).

6.2 DESIGN OF TEST CASES AND SCENARIOS

6.2.1 TEST CASE DESIGN:

The design of tests for software and other engineering products can be as challenging as the initial design of the product. Test case methods provide the developer with a systematic approach to testing. Moreover, these methods provide a mechanism that can help to ensure the completeness of tests and provide the highest like hood for uncovering errors in software.

Any Engineered product can be tested in either of the two ways:

1. Knowing the specified function that a product has been designed to perform, tests can be conducted. These tests demonstrate whether each function is full operational and at the same time searches for errors in each function.
2. Knowing the internal workings of a product, tests can be conducted to ensure that internal operations are performed according to specifications and all internal components hence been adequately exercised.

Test case design methods are divided into two types:

1. White-box testing
2. Black-box testing

1. White-Box Testing

White –box testing, sometimes called glass-box testing is a test, case designed method that uses the control structure of the procedural design to derive test cases. Using white-box testing methods, the s/w engineer can derive test cases that guarantee that all independent paths within a module have been exercised at least once. Exercise all logical decisions on their true and false sides. Execute all loops at their boundaries and within their operational bounds. Exercise internal data structures to ensure their validity.

Basis path testing is a white-box testing technique. The basis path method enables the test case designer to derive a logical complexity measure of a procedural design and use this measure as a guide for defining a basis set are guaranteed to exercise every statement in the program at least one time during testing.

The following table lists the advantages and disadvantages of white-box testing.

Table 6.1: Advantages and Disadvantages of White box testing

Advantages	Disadvantages
As the tester has knowledge of the source code, it becomes very easy to find out which type of data can help in testing the application effectively.	Due to the fact that a skilled tester is needed to perform white-box testing, the costs are increased.
It helps in optimizing the code.	Sometimes it is impossible to look into every nook and corner to find out hidden errors that may create problems, as many paths will go untested.
Extra lines of code can be removed which can bring in hidden defects.	It is difficult to maintain white-box testing, as it requires specialized tools like code analyzers and debugging tools.
Due to the tester's knowledge about the code, maximum coverage is attained during test scenario writing.	

2. Black-Box Testing

Black-box testing ,also called behavioural testing, focuses on the functional requirements of the s/w. Black-box testing enables the software engineer to derive sets of input conditions that will fully exercise all functional requirements of a program. It is a complementary approach that is likely to uncover a different class of errors that white-box methods could not.

Black-box testing attempted to find errors in the following categories.

- Incorrect or missing functions.
- Interface errors.
- Errors in data structures or external data base access.
- Behaviour or performance errors.
- Initialization and termination errors.

Black-box testing purposely disregards control structure; attention is focused on information domain. By applying black-box techniques, we derive a set of cases that satisfies the criteria test cases that reduce, by a count that is greater than one, the number of additional

test cases that must be designed to achieve reasonable testing. Test cases that tell us something about the presence or absence of classes of errors, rather than an error associated only with the specified.

The following table lists the advantages and disadvantages of black-box testing.

Table 6.2: Advantages and Disadvantages of Black box testing

Advantages	**Disadvantages**
Well suited and efficient for large code segments.	Limited coverage, since only a selected number of test scenarios is actually performed.
Code access is not required.	Inefficient testing, due to the fact that the tester only has limited knowledge about an application.
Clearly separates user's perspective from the developer's perspective through visibly defined roles.	Blind coverage, since the tester cannot target specific code segments or errorprone areas.
Large numbers of moderately skilled testers can test the application with no knowledge of implementation, programming language, or operating systems.	The test cases are difficult to design.

Table 6.3: Testing Scenario

S no	Description	Detected Stage	Result	Remarks
1	Uploaded file not a image	Unit Testing	Warning indicating Incorrect input format	Result Dissatisfied
2	Uploaded file is a image	Unit Testing	Data set loaded	Result satisfied
3	Predicting before image pre-processing step	Unit Testing	Warning indicated	Result Dissatisfied
4	Predicting after image pre-processing step	Unit Testing	Image pre-processing done	Result satisfied
5	Predict for image with correct input format and after image pre-processing	Unit testing	Predict Class	Result Satisfied

7.CONCLUSION

7.1 PROJECT CONCLUSION

An obstacle avoidance approach for a mobile robot in indoor environments based on a convolution neural network (CNN) has been developed. A dataset was compiled that contains depth images and robot orientation using a mobile robot platform. The RGBD images obtained by the Kinect camera and the angles acquired by the IMU were recorded. Based on angles, the data was sampled and discretized into five path labels. To the best of our knowledge, this is the first RGBD images dataset that is labeled with IMU data that are used for CNN structures. The results show that our system could manage the obstacle avoidance of mobile robots with an accuracy of approximately 85%. Although it is not completely successful at predictions, it shows a good classification accuracy with greater success. Moreover, the proposed system in this work has quite a low chance to generates a totally opposite decision. Therefore, we believe that the CNN trained by our dataset can result in high classification accuracy in avoiding obstacles in real-time

7.2 FUTURE ENHANCEMENT

In enhancement we will add some algorithms to increase accuracy.

8. REFERENCES

8.1 TEXT BOOKS

[1] R.Siegwart, I.R.N., Introduction to Autonomous Mobile Robot. Massachusetts Institute of Technology Press, Cambridge, USA, 2004.

.[2] Zavlangas, P.G. and S.G. Tzafestas, Motion control for mobile robot obstacle avoidance and navigation: a fuzzy logic-based approach. Systems Analysis Modelling Simulation, 2003. 43(12): p. 1625-1637.

[3] Gregory, D. and J. Michael, Computational principles of mobile robotics. Cambridge University, 2000.

[4] Saffiotti, A., The uses of fuzzy logic in autonomous robot navigation. Soft Computing-A Fusion of Foundations, Methodologies and Applications, 1997. 1(4): p. 180-197.

[5] Engel, J., T. Schöps, and D. Cremers. LSD-SLAM: Large-scale direct monocular SLAM. in European Conference on Computer Vision. 2014. Springer.

[6] Mur-Artal, R., J.M.M. Montiel, and J.D. Tardos, ORB-SLAM: a versatile and accurate monocular SLAM system. IEEE Transactions on Robotics, 2015. 31(5): p. 1147-1163.

[7] Guzel, M.S. and R. Bicker, Vision based obstacle avoidance techniques, in Recent advances in mobile robotics. 2011, InTech.

[8] Lowe, D.G. Object recognition from local scale-invariant features. in Computer vision, 1999. The proceedings of the seventh IEEE international conference on. 1999. Ieee.

[9] Bay, H., T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. in European conference on computer vision. 2006. Springer.

[10] Calonder, M., et al. Brief: Binary robust independent elementary features. in European conference on computer vision. 2010. Springer.

[11] Rublee, E., et al. ORB: An efficient alternative to SIFT or SURF. in Computer Vision (ICCV), 2011 IEEE international conference on. 2011. IEEE.

[12] Tai, L., et al. PCA-aided fully convolutional networks for semantic segmentation of multi-channel fMRI. in Advanced Robotics (ICAR), 2017 18th International Conference on. 2017. IEEE.

[13] Beale, M.H., M.T. Hagan, and H.B. Demuth, Neural Network Toolbox™ User's Guide. The Mathworks Inc, 1992.

[14] Murphy, K.P., Machine learning: a probabilistic perspective. 2012.

[15] Tai, L., S. Li, and M. Liu. A deep-network solution towards modelless obstacle avoidance. in Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on. 2016. IEEE

[16] Liu, C., et al. CNN-Based Vision Model for Obstacle Avoidance of Mobile Robot. in MATEC Web of Conferences. 2017.

[17]ApproximateTime,ROS.org.[Availableonline]:

http://wiki.ros.org/message_filters/ApproximateTime

[18] Obstacle Avoidance Dataset, EIVS Oman on GitHub. [Available online]: https://github.com/eivs-oman/obstacle_avoidance_dataset

8.2 WEBSITES

- www.google.com
- www.tutorialspoint.com
- www.geeksforgeeks.org
- www.javatpoint.com
- ieeexplore.ieee.org
- en.wikipedia.org/wiki/Machine_learning
- www.slideshare.net

8.3 PAPER REFERENCES

- Shiyang Xuan, Guanjun Liu, Zhenchuan Li, Lutao Zheng, Shuo Wang, Changjun Jiang, “Random forest for credit card fraud detection”, 15th International Conference on Networking, Sensing and Control (ICNSC) IEEE, 2018.
- Y. Sahin, E. Duman, "Detecting credit card fraud by decision trees and support vector machines", Proceedings of the International MultiConference of Engineers and Computer Scientists (IMECS), Vol I, 2011.
- KhyatiChaudhary, JyotiYadav, BhawnaMallick, “A review of Fraud Detection Techniques: Credit Card”, International Journal of Computer Applications, Volume 45– No.1, 2012.
- Ayushi Agrawal, Shiv Kumar, Amit Kumar Mishra, “Credit Card Fraud Detection: A case study”, 2nd International Conference on Computing for Sustainable Global Development (INDIACom) IEEE, 2015.
- John O. Awoyemi, Adebayo O. Adetunmbi, Samuel A. Oluwadare, “Credit card fraud detection using machine learning techniques: A comparative analysis”, International Conference on Computing Networking and Informatics (ICCNI) IEEE, 2017