

SMARTPARKX USING MACHINE LEARNING

Major Project Report

Submitted in partial fulfilment for the award of the degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE and ENGINEERING

By

CHAKRADHAR KAMMINANA	19L31A0504
YATHESWAR ETHALAPAKA	19L31A0546
TRISULE KUMAR REDDY SEELAM	19L31A0583
PAVAN KALYAN MAHANTY	20L35A0502

Under the Guidance of

Mrs. Srilatha Yalamati

ASSISTANT PROFESSOR



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VIGNAN'S INSTITUTE OF INFORMATION TECHNOLOGY

(Autonomous)

Affiliated to JNTU Kakinada & Approved by AICTE, New Delhi

Re-Accredited by NBA & NAAC (CGPA of 3.41/ 4.00)

ISO 9001:2015, ISO 14001:2015, OHSAS 18001:2007 Certified Institution

VISAKHAPATNAM – 530 049

April 2023

VIGNAN'S INSTITUTE OF INFORMATION TECHNOLOGY

Department of Computer Science and Engineering



CERTIFICATE

This is to certify that the project report entitled “**SmartParkX using Machine Learning**” the bona fide record of project work carried out under my supervision by **Chakradhar Kamminana (19L31A0504), Yatheswar Ethalapaka (19L31A0546), Trisule Kumar Reddy Seelam (19L31A0583), and Pavan Kalyan Mahanty (20L35A0502)**, during the academic year 2019-2023, in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering of Jawaharlal Nehru Technological University, Kakinada. The results embodied in this project report have not been submitted to any other University or Institute for the award of any Degree or Diploma.

Head of the Department

Mr. B. Dinesh Reddy
ASSOCIATE PROFESSOR

Signature of Project Guide

Mrs. Srilatha Yalamati
ASSISTANT PROFESSOR

DECLARATION

We hereby declare that the project report entitled “**SMARTPARKX USING MACHINE LEARNING**” has been written by us and has not been submitted either in part or whole for the award of any degree, diploma or any other similar title to this or any other university.

CHAKRADHAR KAMMINANA	19L31A0504
YATHESWAR ETHALAPAKA	19L31A0546
TRISULE KUMAR REDDY SEELAM	19L31A0583
PAVAN KALYAN MAHANTY	20L35A0502

Date:

Place: Visakhapatnam

ACKNOWLEDGEMENT

We consider it a privilege to thank all those people who helped us a lot for successful completion of the project “**SmartParkX**”.

First of all, we would like to thank our project guide **Mrs. Srilatha Yalamati, Assistant Professor, Department of Computer Science & Engineering** for helping us a lot in completing our project work and for enlightening us with constructive suggestions for solving our problems patiently and helping us to improve the quality of work.

We would like to thank our ever-inspiring **Head of the Department of Computer Science and Engineering, Mr. B. Dinesh Reddy**, for his spontaneous response to every request though he is busy with his hectic schedule of administration and teaching.

We would like to acknowledge with much appreciation the crucial role of our **Project Coordinator, Mr. Ch. Sekhar, Associate Professor, Department of Computer Science & Engineering** for helping us a lot in completing our project through the academic year. We just wanted to say thank you for being such a wonderful educator as well as a person.

We would like to thank **Principal** of Vignan’s Institute of Information Technology **Dr. B. Arundhati**, for her encouragement to us during the course of this project work.

We also express our sincere thanks to our beloved **Chairman** of Vignan’s Institutions **Dr. L. Rathaiah**, who has given us a lot of support and freedom during our project work.

We thank the entire **faculty** who have been a constant source of support during our study tenure.

ABSTRACT

There has been a tremendous increase in the number of vehicles in the last two decades. So, it becomes important to make effective use of technology to enable hassle-free parking. **SmartParkX** is an advanced parking management system designed to tackle the growing issue of limited parking space in urban areas. The system provides real-time information to drivers on the availability of parking spots in a parking lot, thereby reducing the time taken to search for a spot and minimizing traffic congestion. The system uses state-of-the-art technologies such as computer vision, machine learning, and image processing algorithms to detect and identify available parking spots. The system also utilizes automatic number plate recognition (ANPR) technology to identify the vehicles entering and leaving the parking lot. The system provides an intuitive user interface for both the parking lot management and the drivers. The management interface allows for easy monitoring of the parking lot occupancy, generation of reports, and management of the parking fees. The driver interface provides real-time information on the availability of parking spots, the location of the available spots, and the fees associated with parking. The system is highly scalable and can be customized to meet the specific requirements of different parking lot sizes and configurations. The use of advanced technologies ensures high accuracy in the detection of available parking spots and the identification of vehicles. The benefits of SmartParkX include improved traffic flow, reduced congestion, increased revenue for parking lot operators, and a hassle-free parking experience for drivers. With the increasing demand for parking space in urban areas, SmartParkX offers an efficient solution to alleviate the parking woes of both drivers and parking lot operators.

CONTENTS

TITLE	PAGE NO
Declaration	i
Acknowledgement	ii
Abstract	iii
List of Figures	iv

TITLE	PAGE NO
1. INTRODUCTION	1
1.1 Machine Learning	2
1.1.1 What is Machine Learning	2
1.1.2 How Machine Learning Works	2
1.1.3 Machine Learning Methods	5
1.1.4 History of Machine Learning	6
1.2 Tools	8
1.2.1 Introduction to VS Code	8
1.2.2 Introduction to DBeaver	8
1.2.3 Installation and Setup	9
1.3 Libraries	11
1.3.1 Numpy	11
1.3.2 Pandas	12
1.3.3 Matplotlib	13
1.3.4 OpenCV Python	14
1.3.5 Scikit Learn	14

1.3.6 EasyOCR	15
1.3.7 Pillow	16
1.3.8 PyPNG	17
1.3.9 PyQRCode	17
1.3.10 Twilio	18
1.3.11 MySQL Connector Python	19
1.4 Flask	20
1.4.1 Advantages of Flask	21
1.4.2 Applications of Flask	21
1.4.3 Architecture of Flask Framework	22
1.4.4 Features of Flask	22
2. LITERATURE SURVEY	24
2.1 Literature Survey	25
3. FLOW DIAGRAMS	27
3.1 Data Flow Diagram	28
3.2 UML Diagram	29
3.2.1 Use Case Diagram	29
4. SYSTEM ANALYSIS	31
4.1 Problem Statement	32
4.2 Existing System	32
4.3 Proposed System	32
4.3.1 Model 1	33
4.3.2 Model 2	36

4.4 Source Code	39
4.4.1 main.py	39
4.4.2 anpr.py	42
4.4.3 objectsize.py	46
4.4.4 spacecounter.py	46
4.4.5 payment.py	49
4.4.6 message.py	50
4.4.7 database.py	50
5. REQUIREMENT AND SPECIFICATION	52
5.1 System Specifications	53
5.1.1 Hardware Requirements	53
5.1.2 Software Requirements	53
6. SIMULATION SCREENS	54
6.1 Entry phase	55
6.2 Processing phase	56
6.3 Exit phase	57
7. CONCLUSION	59
7.1 Conclusion	60
8. REFERENCES	61
8.1 References	62

LIST OF FIGURES

FIGURES	PAGE NO
1. VS Code Installation	9
2. DBeaver Installation	10
3. Flask	21
4. Flask Framework	22
5. Data Flow Diagram	28
6. Use Case Diagram for SmartParkX	30
7. Working flow of SmartParkX	33
8. Conventional ANPR System	35
9. Working Flow of Space Counter	36
10. Architecture of YOLO	37
11. Number Plate Scanning	55
12. Customer enters Mobile Number	55
13. Display available slot number	56
14. Working of ANPR module with OCR	56
15. Live working model of Active Contour	57
16. Display of QR code and other Important details	57
17. Remainder message with Payment details	58
18. Post Payment Confirmation at exit gate	58

Chapter 1

INTRODUCTION

1. INTRODUCTION

1.1 Machine Learning

Machine learning is a method of data analysis that automates analytical model building. It is a branch of artificial intelligence based on the idea that systems can learn from data, identify patterns and make decisions with minimal human intervention.

1.1.1 What is Machine Learning

Machine learning is a branch of artificial intelligence (AI) focused on building applications that learn from data and improve their accuracy over time without being programmed to do so.

In data science, an algorithm is a sequence of statistical processing steps. In machine learning, algorithms are 'trained' to find patterns and features in massive amounts of data in order to make decisions and predictions based on new data. The better the algorithm, the more accurate the decisions and predictions will become as it processes more data.

Today, examples of machine learning are all around us. Digital assistants search the web and play music in response to our voice commands. Websites recommend products and movies and songs based on what we bought, watched, or listened to before. Robots vacuum our floors while we do . . . something better with our time. Spam detectors stop unwanted emails from reaching our inboxes. Medical image analysis systems help doctors spot tumors they might have missed. And the first self-driving cars are hitting the road.

We can expect more. As big data keeps getting bigger, as computing becomes more powerful and affordable, and as data scientists keep developing more capable algorithms, machine learning will drive greater and greater efficiency in our personal and work lives.

1.1.2 How Machine Learning works

There are four basic steps for building a machine learning application (or model). These are typically performed by data scientists working closely with the business professionals for whom the model is being developed.

Step 1: Select and prepare a training data set

Training data is a data set representative of the data the machine learning model labelled to solve the problem it's designed to solve. In some cases, the training data is labelled data— 'tagged' to call out features and classifications the model will need to identify. Other data is unlabelled, and the model will need to extract those features and assign classifications on its own.

In either case, the training data needs to be properly prepared randomized, de-duped and checked for imbalances or biases that could impact the training. It should also be divided into two subsets: the training subset, which will be used to train the application and the evaluation subset, used to test and refine it.

Step 2: Choose an algorithm to run on the training data set

Again, an algorithm is a set of statistical processing steps. The type of algorithm depends on the type (labelled or unlabelled) and amount of data in the training data set and on the type of problem to be solved.

Common types of machine learning algorithms for use with labelled data include the following:

- **Regression algorithms**

Linear and logistic regressions are examples of regression algorithms used to understand relationships in data. Linear regression is used to predict the value of a dependent variable based on the value of an independent variable. Logistic regression can be used when the dependent variable is binary in nature: A or B. For example, a linear regression algorithm could be trained to predict a salesperson's annual sales (the dependent variable) based on its relationship to the salesperson's education or years of experience (the independent variables.) Another type of regression algorithm called a support vector machine is useful when dependent variables are more difficult to classify.

- **Decision trees**

Decision trees use classified data to make recommendations based on a set of decision rules. For example, a decision tree that recommends betting on a particular horse to win, place, or show could use data about the horse (e.g., age, winning percentage, pedigree) and apply rules to those factors to recommend an action or decision.

- **Instance-based algorithms**

A good example of an instance-based algorithm is K- Nearest Neighbour. It uses classification to estimate how likely a data point is to be a member of one group, or another based on its proximity to other data points.

Algorithms for use with unlabelled data include the following:

- **Clustering algorithms**

Think of clusters as groups. Clustering focuses on identifying groups of similar records and labelling the records according to the group to which they belong. This is done without prior knowledge about the groups and their characteristics. Types of clustering algorithms include the K-means, Two Step, and Kohonen clustering.

- **Association algorithms**

Association algorithms find patterns and relationships in data and identify frequent 'if-then' relationships called association rules. These are like the rules used in data mining.

- **Neural networks**

A neural network is algorithm that defines a layered network of calculations featuring an input layer, where data is ingested; at least one hidden layer, where calculations are performed make different conclusions about input; and an output layer. Where each conclusion is assigned a probability. A deep neural network defines a network with multiple hidden layers, each of which successively refines the results of the previous layer.

Step 3: Training the algorithm to create the model

Training the algorithm is an iterative process—it involves running variables through the algorithm, comparing the output with the results it should have produced the adjusting weights and biases within the algorithm that might yield a more accurate result, and running the variables again until the algorithm returns the correct result most of the time. The resulting trained, accurate algorithm is the machine learning model—an important distinction to note, because 'algorithm' and 'model' are incorrectly used interchangeably, even by machine learning mavens.

Step 4: Using and Improving the model

The final step is to use the model with new data and, in the best case, for it to improve in accuracy and effectiveness over time. Where the new data comes from will depend on the problem being solved. For example, a machine learning model designed to identify spam will ingest email messages, whereas a machine learning model that drives a robot vacuum cleaner will ingest data resulting from real-world interaction with moved furniture or new objects in the room.

1.1.3 Machine Learning Methods

Machine learning methods (also called machine learning styles) fall into three primary categories.

- **Supervised machine learning**

Supervised machine learning trains itself on a labelled data set. That is, the data is labelled with information that the machine learning model is being built to determine and that may even be classified in ways the model is supposed to classify data. For example, a computer vision model designed to identify purebred German shepherd dogs might be trained on a data set of various labelled dog images. Supervised machine learning requires less training data than other machine learning methods and makes training easier because the results of the model can be compared to actual labelled results. But properly labelled data is expensive to prepare, and there's the danger of overfitting, or creating a model so closely tied and biased to the training data that it doesn't handle variations in new data accurately.

- **Unsupervised machine learning**

Unsupervised machine learning ingests unlabelled data—lots and lots of it—and uses algorithms to extract meaningful features needed to label, sort, and classify the data in real- time, without human intervention. Unsupervised learning is less about automating decisions and predictions, and more about identifying patterns and relationships in data that humans would miss. Take spam detection, for example—people generate more email than a team of data scientists could ever hope to label or classify in their lifetimes. An unsupervised learning algorithm can analyse huge volumes of emails and uncover the features and patterns that indicate spam (and keep getting better at flagging spam over time).

- **Reinforcement learning**

Reinforcement learning is the third and most advanced algorithm category in Machine Learning. Unlike supervised and unsupervised learning, reinforcement learning continuously improves its model by leveraging feedback from previous iterations. This is different to supervised and unsupervised learning, which both reach an indefinite endpoint after a model is formulated from the training and test data segments. Reinforcement learning can be complicated and is probably best explained through an analogy to a video game. As a player progresses through the virtual space of a game, they learn the value of various actions under different conditions and become more familiar with the field of play. Those learned values then inform and influence a player's subsequent behaviour and their performance immediately improves based on their learning and past experience.

1.1.4 History of Machine Learning

The first case of neural networks was in 1943, when neurophysiologist Warren McCulloch and mathematician Walter Pitts wrote a paper about neurons, and how they work. They decided to create a model of this using an electrical circuit, and therefore the neural network was born. In 1950, Alan Turing created the world-famous Turing Test. This test is fairly simple - for a computer to pass, it has to be able to convince a human that it is a human and not a computer. 1952 saw the first computer program which could learn as it ran. It was a game which played checkers, created by Arthur Samuel. Frank Rosenblatt designed the first artificial neural network in 1958, called Perceptron. The main goal of this was pattern and shape recognition.

Another extremely early instance of a neural network came in 1959, when Bernard Widrow and Marcian Hoff created two models of them at Stanford University. The first was called ADELIN, and it could detect binary patterns. For example, in a stream of bits, it could predict what the next one would be. The next generation was called MADELINE, and it could eliminate echo on phone lines, so it had a useful real-world application. It is still in use today. Despite the success of MADELINE, there was not much progress until the late 1970s for many reasons, mainly the popularity of Von Neumann architecture. This is an architecture where instructions and data are stored in the same memory, which is arguably simpler to understand than a neural network, and so many people-built programs based on this.

Neural networks use back propagation (explained in detail in the Introduction to Neural Networks), and this important step came in 1986, when three researchers from the Stanford psychology department decided to extend an algorithm created by Widrow and Hoff in 1962. This therefore allowed multiple layers to be used in a neural network, creating what are known as ‘slow learners’, which will learn over a long period of time.

The late 1980s and 1990s did not bring much to the field. However, in 1997, the IBM computer Deep Blue, which was a chess-playing computer, beat the world chess champion. Since then, there have been many more advances in the field, such as in 1998, when research at AT&T Bell Laboratories on digit recognition resulted in good accuracy in detecting handwritten postcards from the US Postal Service.

21st Century

Since the start of the 21st century, many businesses have realized that machine learning will increase calculation potential. This is why they are researching more heavily into it, to stay ahead of the competition.

Some large projects include:

- **Google Brain (2012)** - This was a deep neural network created by Jeff Dean of Google, which focused on pattern detection in images and videos. It was able to use Google’s resources, which made it incomparable to much smaller neural networks. It was later used to detect objects in YouTube videos.
- **Alex Net (2012)** - Alex Net won the Image Net competition by a large margin in 2012, which led to the use of GPUs and Convolution Neural Networks in machine learning. They also created ReLU, which is an activation function that greatly improves efficiency of CNNs.
- **Deep Face (2014)** - This is a Deep Neural Network created by Facebook, which they claimed can recognize people with the same precision as a human can.
- **Deep Mind (2014)** - This company was bought by Google and can play basic video games to the same levels as humans. In 2016, it managed to beat a professional at the game Go, which is considered to be one the world’s most difficult board games.
- **Open AI (2015)** - This is a non-profit organization created by Elon Musk and others, to create safe artificial intelligence that can benefit humanity.
- **Amazon Machine Learning Platform (2015)** – This is the part of Amazon web

Services and shows how most big companies want to get involved in machine learning. They say it drives many of their internal systems, from regularly used services such as search recommendations and Alexa, to more experimental ones like Prime Air and Amazon Go.

- **ResNet (2015)** - This was a major advancement in CNNs, and more information can be found on the Introduction to CNNs page.
- **U-net (2015)** - This is a CNN architecture specialized in biomedical image segmentation. It introduced an equal amount of up sampling and down sampling layers, and also skip connections. More information on what this means can be found on the Semantic Segmentation page.

1.2 Tools

1.2.1 Introduction to VS Code

Visual Studio Code, also known as VS Code, is a free and open-source source code editor developed by Microsoft for Windows, Linux, and macOS. It is designed to be highly customizable and flexible, while also providing a user-friendly interface for editing, debugging, and managing code.

Some key features of VS Code include:

- A powerful debugger for diagnosing and fixing errors in your code.
- Support for a wide range of programming languages, including JavaScript, Python, C++, and more.
- Built-in Git integration for version control and collaboration with others.
- An extensive library of extensions and plugins to customize and extend the functionality of the editor.
- Cross-platform compatibility, allowing you to work seamlessly across different operating systems.

Overall, VS Code is a popular choice among developers due to its ease of use, flexibility, and powerful set of features. Whether you are a beginner or an experienced developer, VS Code provides a solid foundation for managing and editing your code.

1.2.2 Introduction to DBeaver

DBeaver is a free and open-source universal database tool for developers, database administrators, and analysts. It provides a graphical user interface for managing and

querying various databases, including MySQL, PostgreSQL, Oracle, and Microsoft SQL Server, among others.

Some of the key features of DBeaver include:

- Support for a wide range of databases, including relational databases, NoSQL databases, and cloud-based databases.
- A user-friendly interface for creating, editing, and executing SQL queries and scripts.
- Advanced data visualization and editing capabilities.
- Built-in data modelling tools for designing and manipulating database schemas.
- Support for database administration tasks such as backup, restore, and security management.
- Extensive plugin architecture for extending the functionality of the tool.

Overall, DBeaver is a powerful and versatile tool for working with databases, offering a wide range of features and support for many different database systems. It is a great choice for anyone who needs to work with databases on a regular basis, from beginners to experienced professionals.

1.2.3 Installation and Setup

- **VS Code**

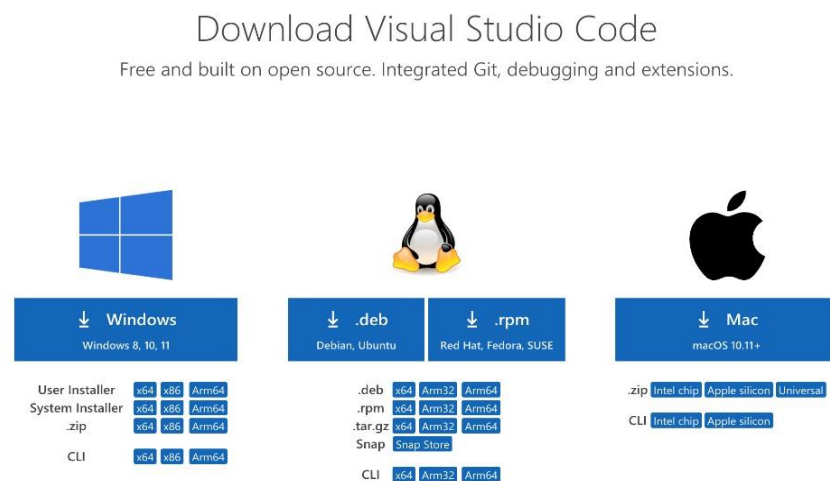


Fig: 1.1 VS Code Installation

1. Go to official website of VS code <https://code.visualstudio.com/download>
 2. Click on the "Download for [your operating system]" button.
 3. Once the download is complete, open the installer file.
 4. Follow the prompts in the installer to complete the installation process.
 5. Once the installation is complete, launch VS Code.
- **DBeaver**
 1. Go to the official website of DBeaver at <https://dbeaver.io/>
 2. Click on the "Download" button on the homepage.
 3. Select your operating system from the list of options.
 4. Choose the appropriate installer package for your system architecture (32-bit or 64-bit).
 5. Once the download is complete, open the installer file.
 6. Follow the prompts in the installer to complete the installation process.
 7. Once the installation is complete, launch DBeaver

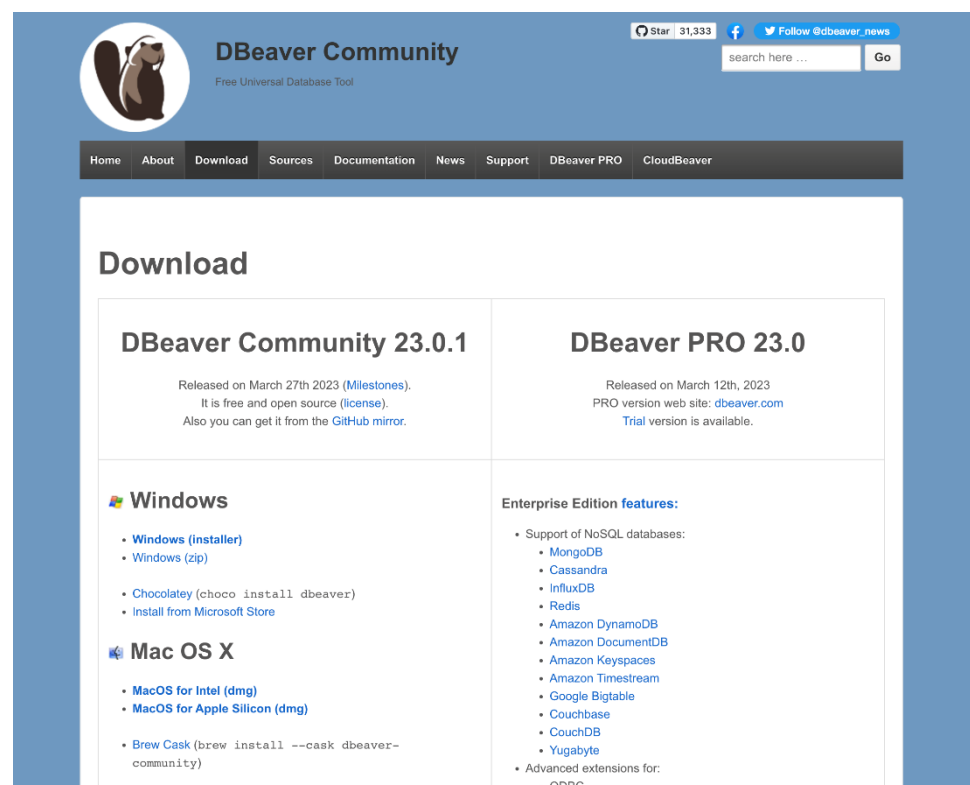


Fig: 1.2 DBeaver Installation

1.3 Libraries

1.3.1 Numpy

NumPy (short for Numerical Python) is a Python library that provides support for large, multidimensional arrays and matrices, as well as a variety of mathematical functions for working with this data. NumPy is a fundamental library for scientific computing with Python, and it is widely used in the fields of data analysis, machine learning, and scientific research. NumPy is built around the concept of n-D arrays, which are N-dimensional arrays of homogeneous data types. The n-D arrays can have any number of dimensions and can contain various data types, including integers, floating-point numbers, and complex numbers. NumPy also provides a set of functions for performing mathematical operations on these arrays, such as addition, subtraction, multiplication, division, and more.

In addition to its support for n-D arrays, NumPy also provides a number of other features for scientific computing and data analysis. These include:

- Linear algebra: NumPy provides a set of functions for performing linear algebra operations such as matrix multiplication, inversion, and decomposition.
- Fourier analysis: NumPy provides functions for performing Fourier transforms and other signal processing operations.
- Random number generation: NumPy includes a powerful set of functions for generating random numbers, which are useful for simulations and other applications.
- Tools for integrating with other data analysis libraries: NumPy integrates seamlessly with other Python libraries such as Pandas and Matplotlib, making it easy to perform complex data analysis tasks.

Overall, NumPy is an essential library for scientific computing and data analysis with Python, and it is widely used in many fields including physics, engineering, finance, and more.

To use NumPy in your Python project, you need to install the NumPy library first. Here's how to do it:

1. Open your command prompt or terminal.
2. Type in the following command and press enter to install NumPy using pip, the package installer for Python: `pip install Numpy`

3. Note: Make sure you have Python and pip installed on your system before running this command.
4. Wait for the installation to complete. You should see output similar to the following if the installation is successful:
5. Successfully installed numpy-1.21.4
6. Once NumPy is installed, you can start using it in your code. To import NumPy, simply add following line to the beginning of your Python script: `import Numpy as np`
7. This line imports NumPy and creates an alias for it, so you can refer to it as "np" in your code.

1.3.2 Pandas

Pandas is a popular Python library used for data manipulation and analysis. It provides powerful data structures for handling and analysing data, as well as a variety of tools for data cleaning, reshaping, and visualization.

Some key features of Pandas include:

- Data structures for working with both labelled and unlabelled data, including Series (1D) and Data Frame (2D).
- Efficient data manipulation tools for merging, grouping, filtering, and transforming data.
- Built-in support for handling missing or incomplete data.
- A variety of data visualization tools for creating charts, graphs, and other visual representations of data.
- Integration with other Python libraries for scientific computing, such as NumPy and Matplotlib.

To install Pandas, you can follow these steps:

1. Open your command prompt or terminal.
2. Type in the following command and press enter to install Pandas using pip, the package installer for Python: `pip install pandas`
3. Note: Make sure you have Python and pip installed on your system before running this command.
4. Wait for the installation to complete. You should see output similar to the following if the installation is successful

5. Successfully installed pandas-1.3.4
6. Once Pandas is installed, you can start using it in your code. To import Pandas, simply add the following line to the beginning of your Python script: `import pandas as pd`
7. This line imports Pandas and creates alias for it, you can refer to it as "pd" in your code.

1.3.3 Matplotlib

Matplotlib is a popular Python library used for creating high-quality 2D and 3D plots and charts. It provides a variety of tools for data visualization, including line plots, scatter plots, bar charts, histograms, and more.

Some key features of Matplotlib include:

- Support for a wide range of plot types, including 2D and 3D plots, scatter plots, histograms, and more.
- Highly customizable plots, with options for changing colors, fonts, labels, and other visual elements.
- Integration with other Python libraries for scientific computing, such as NumPy and Pandas.
- Support for interactive plots, allowing users to zoom, pan, and modify plots on the fly.

To install Matplotlib, you can follow these steps:

1. Open your command prompt or terminal.
2. Type in the following command and press enter to install Matplotlib using pip, the package installer for Python: `pip install matplotlib`
3. Note: Make sure you have Python and pip installed on your system before running this command.
4. Wait for the installation to complete. You should see output similar to the following if the installation is successful: `Successfully installed matplotlib-3.4.3`
5. Once Matplotlib is installed, you can start using it in your Python code. To import Matplotlib, simply add the following line to the beginning of your Python script: `import matplotlib.pyplot as plt`
6. This line imports the Pyplot module of Matplotlib and creates an alias for it, so you can refer to it as "plt" in your code.

1.3.4 OpenCV Python

OpenCV (OpenSource Computer Vision Library) is an open-source computer vision and machine learning software library. It has a wide range of applications, from real-time computer vision and facial recognition to image and video processing.

To install OpenCV for Python, you can follow these steps:

1. Open your command prompt or terminal.
2. Type in the following command and press enter to install OpenCV using pip, the package installer for Python: `pip install opencv-python`
3. Note: Make sure you have Python and pip installed on your system before running this command.
4. Wait for the installation to complete. You should see output similar to the following if the installation is successful:
5. Successfully installed opencv-python-4.5.4.58
6. Once OpenCV is installed, you can start using it in your Python code. To import OpenCV, simply add the following line to the beginning of your Python script:
`import cv2`
7. This line imports OpenCV and makes it available for use in your code.

1.3.5 Scikit Learn

Scikit-learn (also known as sklearn) is a popular Python library for machine learning. It provides a wide range of tools and algorithms for data analysis, modelling, and prediction.

Some key features of Scikit-learn include:

- A variety of supervised and unsupervised learning algorithms, including classification, regression, clustering, and dimensionality reduction.
- Integration with other Python libraries for scientific computing, such as NumPy and Pandas.
- Built-in support for data pre-processing, including feature scaling and normalization.
- A wide range of evaluation metrics for assessing model performance, including accuracy, precision, recall, and F1 score.
- Support for model selection and hyperparameter tuning, including cross-validation and grid search.

To install Scikit-learn, you can follow these steps:

1. Open your command prompt or terminal.
2. Type in the following command and press enter to install Scikit-learn using pip, the package installer for Python: `pip install scikit-learn`
3. Note: Make sure you have Python and pip installed on your system before running this command.
4. Wait for the installation to complete. You should see output similar to the following if the installation is successful: `Successfully installed scikit-learn-1.0`
5. Once Scikit-learn is installed, you can start using it in your Python code. To import Scikit-learn, simply add the following line to the beginning of your Python script:
`import sklearn`
6. This line imports Scikit-learn and makes it available for use in your code.

1.3.6 EasyOCR

EasyOCR is a Python library for optical character recognition (OCR). It provides a simple and easy-to-use interface for extracting text from images and other types of media.

Some key features of EasyOCR include:

- Support for over 70 languages, including English, Chinese, Japanese, Korean, and many others.
- Integration with other Python libraries for image processing, such as OpenCV and Pillow.
- Built-in support for multiple OCR engines, including Tesseract and CuneiForm.
- Easy-to-use API for extracting text from images and other types of media.
- Support for both CPU and GPU processing.

To install EasyOCR, you can follow these steps:

1. Open your command prompt or terminal.
2. Type in the following command and press enter to install EasyOCR using pip, the package installer for Python: `pip install easyocr`
3. Note: Make sure you have Python and pip installed on your system before running this command.
4. Wait for the installation to complete. You should see output similar to the following if the installation is successful: `Successfully installed easyocr-1.3.2`

5. Once EasyOCR is installed, you can start using it in your Python code. To import EasyOCR, simply add the following line to the beginning of your Python script:
`import easyocr`
6. This line imports EasyOCR and makes it available for use in your code.

1.3.7 Pillow

Pillow is a popular Python library for image processing and manipulation. It provides a wide range of tools and functions for working with images, including opening and saving images in various formats, resizing and cropping images, applying filters and effects, and much more.

Some key features of Pillow include:

- Support for a wide range of image file formats, including JPEG, PNG, BMP, and many others.
- A simple and intuitive API for working with images, including loading and saving images, and applying various transformations and filters.
- Support for advanced image processing tasks, such as image enhancement, segmentation, and object detection.
- Integration with other Python libraries for scientific computing, such as NumPy and Scikitlearn.
- Support for both CPU and GPU processing.

To install Pillow, you can follow these steps:

1. Open your command prompt or terminal.
2. Type in the following command and press enter to install Pillow using pip, the package installer for Python: `pip install pillow`
3. Note: Make sure you have Python and pip installed on your system before running this command.
4. Wait for the installation to complete. You should see output similar to the following if the installation is successful: `Successfully installed pillow-8.3.2`
5. Once Pillow is installed, you can start using it in your Python code. To import Pillow, simply add the following line to the beginning of your Python script: `from PIL import Image`
6. This line imports the Image module from Pillow and makes it available for use in your code.

1.3.8 PyPNG

PyPNG is a Python library for reading and writing PNG (Portable Network Graphics) image files. It provides a simple and easy-to-use interface for working with PNG images in Python.

Some key features of PyPNG include:

- Support for reading and writing PNG images with various bit depths, color types, and compression levels.
- Integration with other Python libraries for image processing, such as Pillow and NumPy.
- Support for both pure Python and C extensions, depending on your needs and performance requirements.

To install PyPNG, you can follow these steps:

1. Open your command prompt or terminal.
2. Type in the following command and press enter to install PyPNG using pip, the package installer for Python: *pip install pypng*
3. Note: Make sure you have Python and pip installed on your system before running this command.
4. Wait for the installation to complete. You should see output similar to the following if the installation is successful: Successfully installed pypng-0.0.21
5. Once PyPNG is installed, you can start using it in your Python code. To import PyPNG, simply add the following line to the beginning of your Python script: `import png`.
6. This line imports the PyPNG module and makes it available for use in your code.

1.3.9 PyQRCode

PyQRCode is a Python library for generating QR (Quick Response) codes. It provides a simple and easy-to-use interface for creating QR codes in Python.

Some key features of PyQRCode include:

- Support for creating QR codes of various sizes and error correction levels.
- Integration with other Python libraries for image processing, such as Pillow and NumPy.
- Support for generating QR codes with custom logos or images embedded.

To install PyQRCode, you can follow these steps:

1. Open your command prompt or terminal.
2. Type in the following command and press enter to install PyQRCode using pip, the package installer for Python: `pip install pyqrcode`
3. Note: Make sure you have Python and pip installed on your system before running this command.
4. Wait for the installation to complete. You should see output similar to the following if the installation is successful: `Successfully installed pyqrcode-1.2.1`
5. Once PyQRCode is installed, you can start using it in your Python code. To import PyQRCode, simply add the following line to the beginning of your Python script:
`import pyqrcode`
6. This line imports the PyQRCode module and makes it available for use in your code.

1.3.10 Twilio

Twilio is a cloud communications platform that allows developers to programmatically send and receive text messages, voice calls, and other forms of communication. It provides a simple and easy-to-use API for integrating communication capabilities into your software applications.

To use Twilio in your Python projects, you need to follow these steps:

1. Sign up for a Twilio account at <https://www.twilio.com/try-twilio>. You will need to provide some basic information and verify your email address.
2. Once you have signed up, you will be given an Account SID and an Auth Token. These are required to authenticate your requests to the Twilio API.
3. Install the Twilio Python helper library using pip, the package installer for Python. Open your command prompt or terminal and type the following command:
4. Copy code
5. `pip install twilio`
6. Wait for the installation to complete. You should see output similar to the following if the installation is successful: `Successfully installed twilio-6.63.0`.
7. Once the Twilio helper library is installed, you can start using it in your Python code. To send a text message, for example, you can use the following code:

```

from twilio.rest import Client
#Your Account SID and Auth Token from twilio.com/console
account_sid = 'your_account_sid'
auth_token = 'your_auth_token'
client = Client(account_sid, auth_token)
message = client.messages \ .create(body="Hello from Twilio!",
from_='+1234567890',
# Replace with your Twilio number to='+9876543210'
# Replace with your mobile number)
print(message.sid)

```

8. This code sends a text message to the specified mobile number using your Twilio account. You will need to replace the `your_account_sid` and `your_auth_token` variables with your actual Account SID and Auth Token, which you can find on the Twilio console.

On the other hand, if NumPy and Scipy is not yet installed on your Python workstation then, you can install them by using either pip or conda.

Another option to use scikit-learn is to use Python distributions like Canopy and Anaconda because they both ship the latest version of scikit-learn.

1.3.11 MySQL Connector Python

MySQL Connector/Python is a Python library that provides a standardized database driver for connecting Python applications to MySQL databases. It is a pure Python implementation of the MySQL client/server protocol and is fully compliant with the Python Database API specification.

To use MySQL Connector/Python in your Python projects, you need to follow these steps:

1. Install MySQL Connector/Python using pip, the package installer for Python. Open your command prompt or terminal and type the following command:
2. Copy code
3. `pip install mysql-connector-python.`
4. Wait for the installation to complete. You should see output similar to the following if the installation is successful: Successfully installed mysql-connector-python-8.0.27

5. Once MySQL Connector/Python is installed, you can start using it in your Python code. To connect to a MySQL database, for example, you can use the following code:

```
import mysql.connector
# Establish a connection to the database
cnx = mysql.connector.connect(user='your_username',
password='your_password', host='your_hostname', database='your_database')
# Perform a query
cursor = cnx.cursor()
query = ("SELECT * FROM your_table") cursor.execute(query)
# Process the results for (column1, column2, column3) in cursor:
print("{}\t{}\t{}".format(column1, column2, column3))
# Clean up cursor.close() cnx.close()
```

6. This code connects to a MySQL database using the specified username, password, hostname, and database name. It then performs a SELECT query on a table and processes the results. You will need to replace the your_username, your_password, your_hostname, your_database, and your_table variables with your actual database credentials and table name.

1.4 Flask

Flask is a popular micro web framework for building web applications in Python. It is lightweight and easy to use, making it a popular choice for beginners and experienced developers alike. Flask is designed to be simple and flexible, providing a basic set of features that can be extended as needed using third-party extensions.

One of the key features of Flask is its use of routes, which allow developers to define URLs and associate them with functions that handle requests to those URLs. Flask also provides built-in support for handling forms, cookies, and sessions, as well as integrating with various databases and authentication systems.

Overall, Flask is a great choice for building small to medium-sized web applications that require flexibility and customization. Its simplicity and ease of use make it a popular

choice among Python developers, and its active community of contributors provides a wide range of extensions and resources for developers to use.



Fig: 1.3 Flask

1.4.1 Advantages of Flask

Now, you would be able to understand, when we define Flask as a micro-framework that is built using WSGI and Jinja 2 template engine.

Major advantages of Flask are:

1. Ease of setup and use.
2. Freedom to build the structure of the web application.

With freedom comes responsibility, similarly, Flask needs the developers to carefully structure it, since Flask doesn't have "flask rules" to follow as compared to frameworks like Django. As the web app increases in complexity, this structuring is what is going to be the foundation.

1.4.2 Applications of Flask

Flask is used for building web applications and APIs using Python. It is a micro web framework that provides a basic set of features and tools for developing web applications, while also allowing developers to extend and customize its functionality using third-party packages.

Here are some common use cases for Flask:

1. Building small to medium-sized web applications: Flask's simplicity and flexibility make it a great choice for building small to medium-sized web applications, such as blogs, online stores, and social media platforms.
2. Developing APIs: Flask can also be used for building APIs (Application Programming Interfaces) that allow other applications or services to access and interact with your web application's data and functionality.

3. Rapid prototyping: Flask's lightweight nature and ease of use make it an ideal choice for rapidly prototyping web applications or testing out new ideas.
4. Customizing larger web applications: Flask can be used to customize and extend the functionality of larger web applications, such as content management systems (CMS) or ecommerce platforms, by creating custom plugins or extensions.

Overall, Flask is a versatile web framework that can be used for a variety of web development tasks, from building small web applications to developing complex APIs and customizing larger applications.

1.4.3 Architecture of Flask Framework

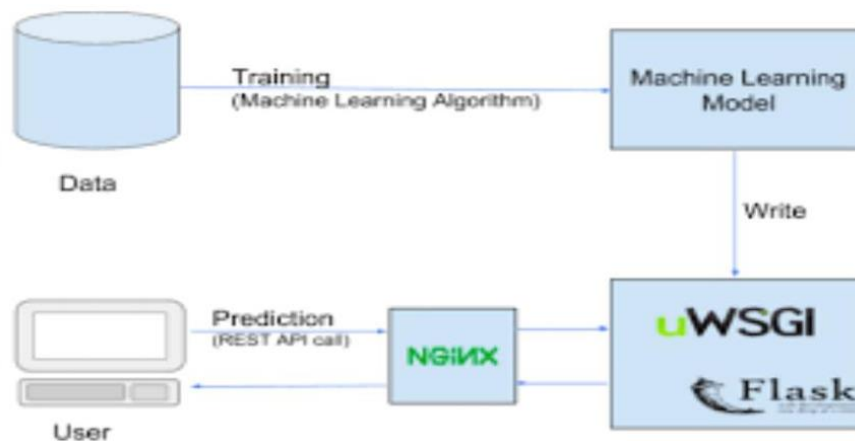


Fig: 1.4 Flask Framework

1.4.4 Features of Flask

1. Lightweight and flexible: Flask is a lightweight web framework that provides only the essential features required for building web applications. It is also very flexible and can be easily customized and extended using third-party packages.
2. Routes and URL mapping: Flask allows developers to define routes and map URLs to specific functions, making it easy to handle incoming requests and build web pages dynamically.
3. Built-in development server: Flask comes with a built-in development server, making it easy to test and debug web applications without the need for additional server software.

4. Templating engine: Flask includes a templating engine that allows developers to create dynamic HTML templates using variables, loops, and conditional statements.
5. Support for various data formats: Flask supports a wide range of data formats, including JSON, XML, and YAML, making it easy to build web APIs that can communicate with other applications.
6. Integration with popular databases: Flask integrates with a variety of databases, including SQLite, MySQL, and PostgreSQL, making it easy to store and retrieve data in web applications.

Chapter 2

LITERATURE SURVEY

2. LITERATURE SURVEY

2.1 Literature Survey

Literature survey for SmartParkX reveals that there are several existing parking systems that are currently being used. These systems are conventional and often rely on manual operations, making them time-consuming, inefficient, and prone to errors. One of the major drawbacks of the existing parking systems is the lack of real-time information on parking slot availability. Moreover, traditional parking systems are not able to handle the increasing number of vehicles on the roads. To address these issues, many research studies have been conducted to develop automated parking systems that are efficient, accurate, and easy to use. These systems are based on various technologies such as computer vision, machine learning, and artificial intelligence. The most common approach is to use computer vision techniques to capture and process images of parking lots to detect the availability of parking slots. The captured images are then analyzed using machine learning algorithms to detect and recognize the license plate numbers of the parked vehicles. Recent studies have shown that deep learning algorithms such as YOLO (You Only Look Once) and Faster R-CNN (Region-based Convolutional Neural Network) can accurately detect the number of available parking slots in real-time. In addition, Non-Maximum Suppression (NMS) techniques are used to remove the overlapping bounding boxes and to select the most relevant ones for the parking slots. Another important aspect of the SmartParkX system is the Automatic Number Plate Recognition (ANPR) technology. This technology is used to automatically recognize the license plate numbers of the parked vehicles, allowing for easy identification and tracking of the parked vehicles. Moreover, OCR (Optical Character Recognition) techniques are used to extract the license plate numbers from the captured images. In conclusion, the literature survey

for SmartParkX highlights the need for an automated parking system that can efficiently manage the increasing number of vehicles on the roads. The use of computer vision, machine learning, and artificial intelligence technologies has led to the development of advanced parking systems that are accurate, efficient, and easy to use. The integration of ANPR and OCR technologies further enhances the performance and reliability of the SmartParkX system.

Chapter 3

FLOW DIAGRAMS

3. FLOW DIAGRAMS

3.1 Data Flow Diagram

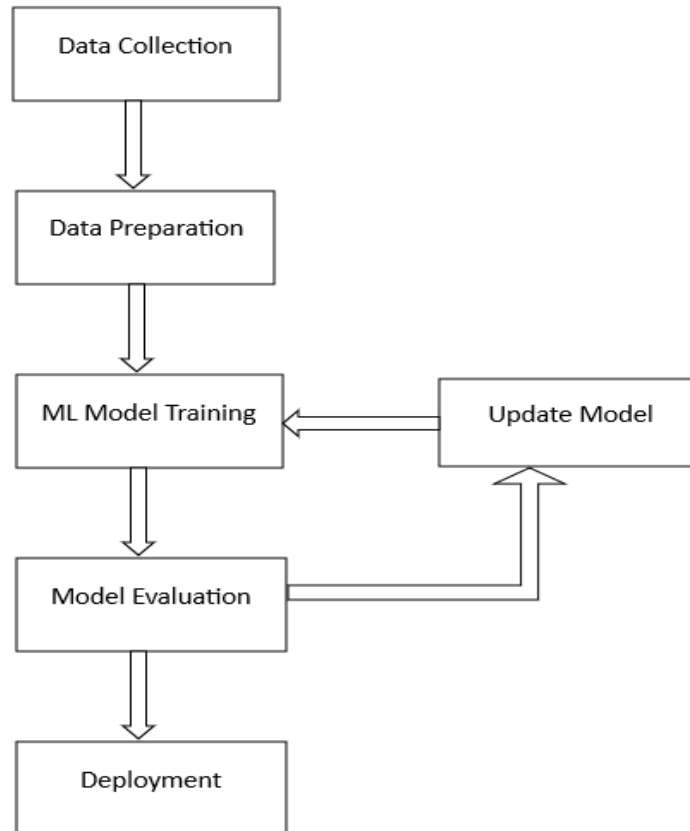


Fig: 3.1 Data Flow Diagram

- **Data Collection**

Data Collection involves collecting the data as per the requirements of the project. In this paper we have performed two experiments with two different data-sets respectively. The first data-set was obtained from kaggle. The data set consists of 300 records.

- **Data Preparation**

Data preparation involves the cleaning and exploration of data to find relationships among the features of the data. In first experiment the data was inconsistent and had duplicates. These had to be removed and the data-set is cleaned. The second data-set had many features and the data is explored and the right features were selected for the utilization of the model.

- **Data Preparation**

Data preparation involves the cleaning and exploration of data to find relationships among the features of the data. In first experiment the data was inconsistent and had duplicates. These had to be removed and the data-set is cleaned. The second data-set had many features and the data is explored and the right features were selected for the utilization of the model.

- **ML Model Training**

Initializing the ML models and fitting them to train. We have used various classifiers like Logistic Regression, SVM, KNN, Decision Tree, Random Forest and Gradient Boosting Classifier. The training to testing ratio for the first data-set is 80/20 and 70/30 for the second data-set.

- **Model Evaluation**

This phase involves evaluating the models to see the performance of each model. Both experiments were evaluated by using confusion matrix and accuracy score.

- **Update Model**

Update the model parameters to improve the performance. We have performed hyper- parameter tuning to each and every algorithm to find the best parameters.

- **Deployment**

Deployment is to host the application in some cloud platform. We have deployed our models in Heruko cloud platform.

3.2 UML Diagram

A UML diagram is a diagram based on the UML (Unified Modeling Language) with the purpose of visually representing a system along with its main actors, roles, actions, artifacts or classes, in order to better understand, alter, maintain, or document information about the system. UML is not a programming language but tools can be used to generate code in various languages using UML diagrams. UML has a direct relation with object-oriented analysis and design.

3.2.1 Use Case Diagram

A use case diagram is used to represent the dynamic behaviour of a system. It encapsulates the system's functionality by incorporating use cases, actors, and their relationships. It models the tasks, services, and functions required by a system/subsystem of an application. It depicts the high-level functionality of a system and also tells how the user

handles a system. The main purpose of a use case diagram is to portray the dynamic aspect of a system. It accumulates the system's requirement, which includes both internal as well as external influences. It invokes persons, use cases, and several things that invoke the actors and elements accountable for the implementation of use case diagrams. It represents how an entity from the external environment can interact with a part of the system.

Some purposes of a use case diagram:

- It gathers the system's needs.
- It depicts the external view of the system.
- It recognizes the internal as well as external factors that influence the system.
- It represents the interaction between the actors.

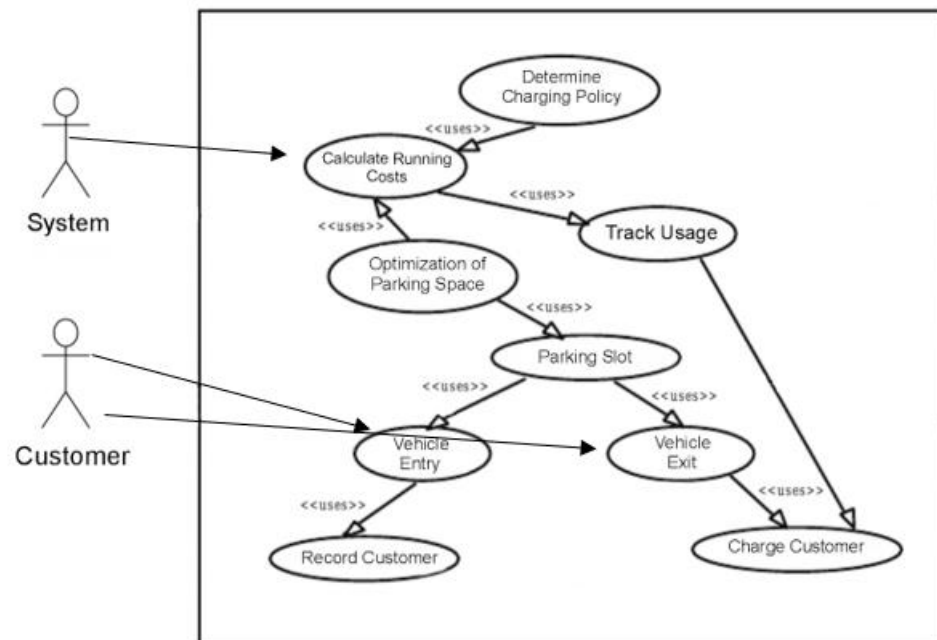


Fig: 3.2 Use Case Diagram for SmartParkX

This is the use case diagram which consists of different phases involved in this project. It consists of different parts, Use cases and Actors. Here System and Customer are Actors. Determine Charging Policy, Calculate Running Costs, Track Usage, Optimization of Parking Space, Parking Slot, Vehicle Entry, Vehicle Exit, Record Customer, Charge Customer are Use cases used in this project.

Chapter 4

SYSTEM ANALYSIS

3. SYSTEM ANALYSIS

4.1 Problem Statement

The problem statement for SmartParkX is to develop an automated parking system that can efficiently manage and monitor the parking slots in real-time without any human intervention. The current conventional parking systems have several drawbacks such as inefficient parking space utilization, time-consuming parking procedures, and human errors in parking ticketing and payment. SmartParkX aims to overcome these challenges by implementing advanced technologies such as computer vision, machine learning, and IoT to provide a seamless parking experience to the drivers. The system aims to reduce the time and effort required for parking and payment procedures, minimize traffic congestion, and optimize the utilization of parking spaces. Additionally, the system aims to provide real-time information about parking availability and manage the parking slots effectively to avoid overcrowding and unauthorized parking. Overall, the problem statement for SmartParkX is to develop a reliable, efficient, and cost-effective parking system that enhances the parking experience for the drivers while maximizing the utilization of parking spaces.

4.2 Existing System

The existing traditional parking systems rely on manual ticketing or verification processes, which can result in long wait times and a frustrating experience for customers. Customers are required to take a ticket upon entering the parking lot and then pay the fee upon exiting, often leading to confusion and inconvenience. The allocation of parking spaces is also done manually, leading to errors and inefficient use of space. Moreover, there is no real-time information available on parking availability, leading to uncertainty and frustration for customers. The traditional parking systems are therefore inefficient and do not provide a satisfactory experience for customers.

4.3 Proposed System

The proposed SmartParkX system leverages advanced technologies to automate the parking process and provide a convenient and efficient parking experience to customers. The system employs ANPR technology to read the vehicle's license plate as it approaches the parking lot, eliminating the need for manual ticketing or verification and ensuring a

smooth and hassle-free entry process. The YOLO object detection algorithm is used to identify the type of vehicle and allocate the appropriate parking slot, while the Space Counter algorithm efficiently manages the parking lot and ensures that the available slots are allocated correctly based on the type of vehicle. The system calculates the parking fee in real-time based on fixed rates for 2-wheelers and 4-wheelers and displays it on a QR code. Upon successful payment, the system updates the checkout time and frees up the parking slot for the next vehicle. The SmartParkX system provides a superior parking experience to customers and ensures efficient use of parking spaces.

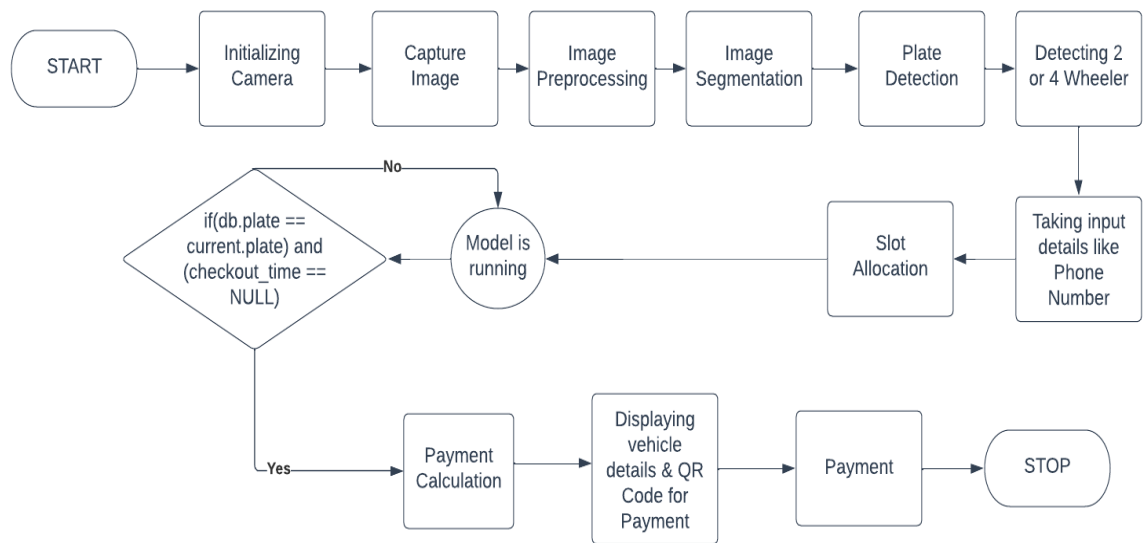


Fig: 4.1 Working flow of SmartParkX

4.3.1 Model 1

The data collection process for the SmartParkX can be described as follows:

The system first gathers the following information from the user:

- Vehicle Number
- Vehicle Type
- Phone Number

This information is then stored into a data set. Once the data set is created, the following steps are performed:

1. The ANPR function is used to fetch the number plate of the vehicle.

2. The Space Counter function is called to determine the available parking slots.
3. The Object Size function is used to determine whether the vehicle is a two-wheeler or a four-wheeler.
4. The Timing function and the Date function are called to record the check-in time and check-in date of the vehicle.

After the completion of above steps, the received data is pushed into a Cloud DB (MySQL) through the following set of information:

- Vehicle Number
- Vehicle Type
- Slot Allocated
- Phone Number
- Check-In Time
- Check-Out Time

Finally, data cleaning operations are performed on the dataset to remove any noise and normalize the features to a single scale to improve the performance of the system.

- **Automatic Number Plate Recognition (ANPR)**

Vehicle license plate numbers can be automatically read and recognised using a technique called automatic number plate recognition (ANPR). An image of a vehicle's license plate is captured by ANPR systems, which then extract the characters from it and match them to a database of recognised license plates using a combination of cameras, image processing methods, and optical character recognition (OCR) software. The ANPR system typically consists of several components, including cameras that capture images of the license plates, image processing software that analyses the images, and a database that stores the license plate information. The process of recognizing the license plate is done in several stages. The first stage involves capturing an image of the license plate. This can be done using a camera that is either fixed or mounted on a moving vehicle, such as a police car or toll booth. The camera captures an image of the license plate, which is then passed on to the image processing software. The second stage involves pre-processing the image to enhance the license plate area and improve the quality of the image. This may involve adjusting the contrast, brightness, and sharpness of the image, as well as removing any noise or distortion. The third stage involves segmenting the characters in the license plate image. This involves

separating the individual characters from the background and other characters. The image processing software then applies OCR techniques to recognize the characters in the license plate. In the last step, the recognised license plate characters are compared to a collection of previously known license plates. The ANPR system can then start an action, such as opening a gate or sending a warning to law enforcement, if a match is identified.

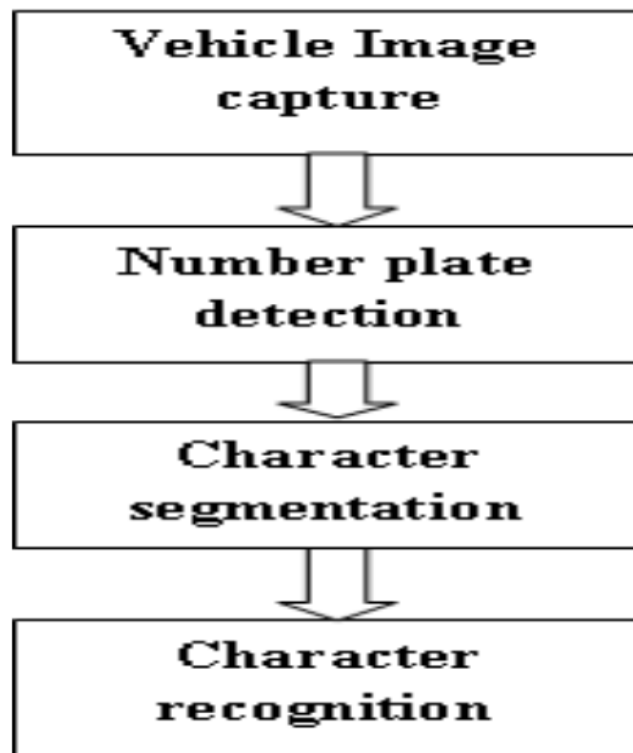


Fig: 4.2 Conventional ANPR system

- **Implementation of Space Counter**

Here is the simplified version of the algorithm we have implemented for the space counter functionality in SmartParkX:

1. Load a live video feed of the parking slots.
2. Map the video feed with a mask to detect the parking slots and get their coordinates.
3. Read each frame of the video feed and check if a slot is empty or occupied.
4. For each frame, skip the frames with a 30ms interval to reduce processing time.
5. Check if a parking slot is empty using an Active Contour model.
6. If a slot is empty, add the index of that slot to a list of empty parking slots.

7. If a slot is occupied, add the index of that slot to a list of occupied parking slots.
8. Calculate the total count of empty parking slots and occupied parking slots.
9. Map the empty parking slots in the video feed with a green field and the occupied parking slots with a red field.
10. Display the count of empty parking slots and occupied parking slots on top of the video feed.

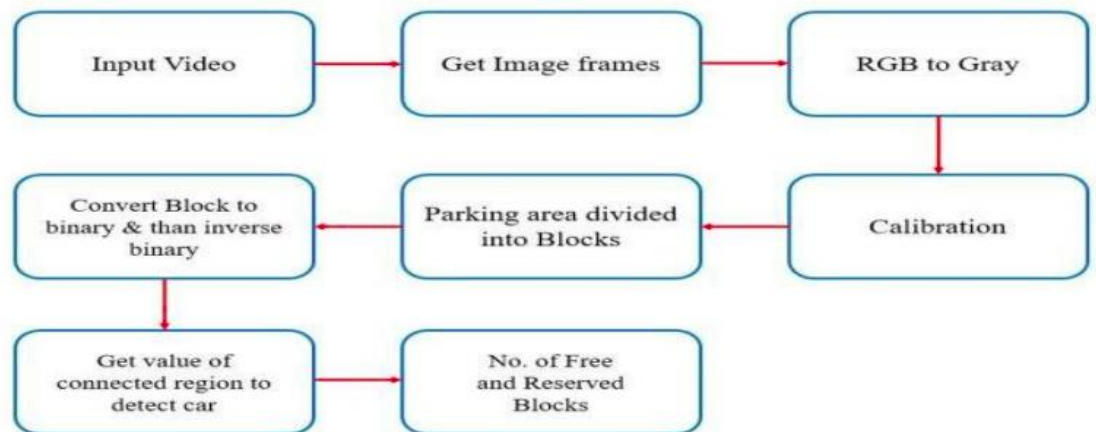


Fig: 4.3 Working Flow of Space Counter

4.3.2 Model 2

In SmartParkX, the collected data of the users' vehicle number, vehicle type, and phone number are used to allocate parking slots and record the check-in time and date. The system utilizes functions such as the Space Counter, Timing, and Date to provide real-time information to the customer about available parking slots and the check-in process.

- **YOLO (You only look once)**

A neural network is used by the cutting-edge object detection algorithm YOLO (You Only Look Once) to recognise items in an image. YOLO performs all the detection steps in a single end-to-end network, making it incredibly quick and effective. This is in contrast to traditional object detection algorithms, which require multiple stages of processing to identify objects. When using the YOLO algorithm, an image is divided into a grid of cells, and the bounding boxes, class probabilities, and confidence scores are predicted for each cell. The class probabilities show the probability that each object belongs to a particular class, while the bounding boxes show the position and size of objects within the cell.

(e.g., car, pedestrian, bicycle, etc.). The confidence score represents the overall confidence that an object exists within the cell. During training, YOLO uses a loss function that penalizes incorrect predictions of the bounding boxes, class probabilities, and confidence scores. This loss function is back propagated through the network to update the weights and biases, enabling the network to improve its predictions over time. One of the main advantages of YOLO is its speed. YOLO is able to detect objects in near real-time, making it ideal for applications that require fast and accurate object detection, such as self-driving cars, video surveillance systems, and robotics. Additionally, YOLO is able to detect multiple objects within a single image, making it more accurate and efficient than traditional object detection algorithms.

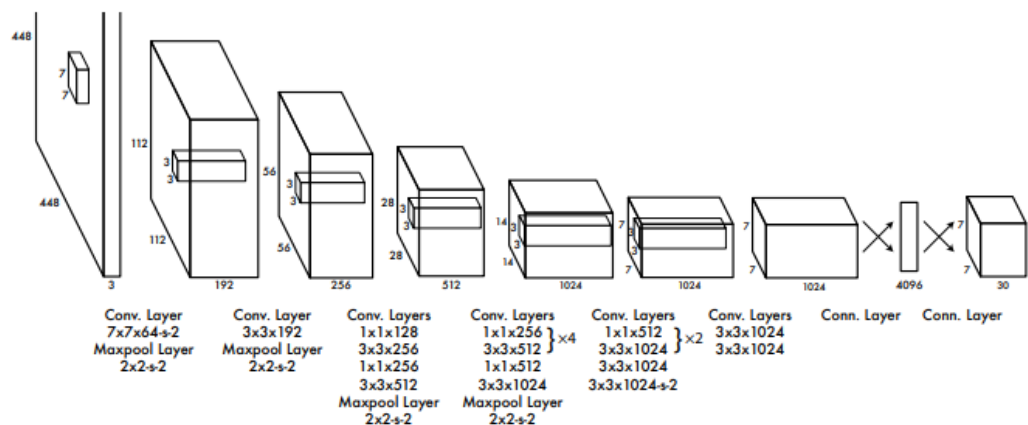


Fig: 4.4 Architecture of YOLO

- **Non-Maximum Suppression (NMS)**

Object detection uses the Non-Maximum Suppression (NMS) method to get rid of duplicate detections of the same object. In the process of detecting objects using algorithms like YOLO or SSD, multiple bounding boxes may be predicted for the same object in a single image, due to various reasons like overlapping or multiple detections. NMS is used to filter out these redundant bounding boxes and keep only the most relevant one. The NMS algorithm works by iterating through all the bounding boxes detected in the image and calculating a score for each box based on the probability of containing an object and the overlap with other boxes. The highest scoring box is then chosen, and all other boxes that significantly overlap with it are suppressed, or removed from the output. The process is repeated until

no more boxes are left. The key parameter in NMS is the overlap threshold, which determines how much overlap is considered significant. Usually, a threshold of 0.5 or higher is used, meaning that any box with an overlap of 50% or more with the selected box will be suppressed. NMS is a crucial technique for object detection algorithms as it helps to eliminate duplicate detections and produce a cleaner output with fewer false positives. It improves the accuracy and efficiency of object detection systems, making them more reliable for real-world applications.

- **Active Contour Model**

Active contour models, also referred to as snakes, are deformable contours that have been used for a variety of image analysis tasks, including the image-based tracking of stiff and nonrigid objects. They are a special example of the general theory of multidimensional deformable models. Using energy forces and constraints, the segmentation method known as active contour isolates the crucial pixels from an image for further processing and analysis. An active segmentation algorithm is what is meant by active contour. The lines defining the area of interest in a picture are called contours. A contour is made up of a set of interpolated coordinates. Depending on how the curve in the image is defined, either a linear, spline, or polynomial interpolation method may be used. Active contours are primarily used in image processing to create closed contours for regions and designate smooth shapes in images. It is primarily used to spot asymmetrical patterns in pictures. Several contour methods that make use of both internal and external forces are used to calculate the model's curvature. The image's shape and the energy function are inextricably linked. In contrast to internal energy, which is used to control deformable changes, external energy is defined as the total of forces brought about by the image and particularly used to control where the contour appears on the image. The needs decide the contour segmentation restrictions for a specific image. By defining the energy function, the desired shape can be achieved. Contour deformation is referred to as a group of points that identify a contour. The contour of the intended image is represented by this shape, which was created by minimizing the energy function.

4.4 Source Code

4.4.1 main.py

```
import time
import mysql.connector
from ANPR.anpr import ANPR
from OBJECTSIZE.objectsize import OBJECT_SIZE
from SPACECOUNTER.spacecounter import SPACE_COUNTER
from PAYMENT.payment import payment
from MESSAGES.message import MESSAGE
from datetime import datetime , timedelta ,date
from DATABASE.database import DATABASE_DETAILS
from flask import Flask,render_template,request
app=Flask(__name__)
@app.route('/',methods=['POST','GET'])
def index():
    slot="
    if request.method=='POST':
        slot=SPACE_COUNTER()
        form_data=request.form
        temp=form_data['phone']
        core_part(temp)
    return render_template('input.html',slot=slot)
def core_part(number):
    phone_number=number
    # ANPR PART
    number_plate=ANPR()
    # Timing Part
    def timing():
        curr_time = time.strftime("%H:%M", time.localtime())
        return curr_time
    checkin_time=timing()
    time1 = datetime.strptime(checkin_time, "%H:%M")
    checkout_time=( datetime.now() + timedelta( minutes=40 )).strftime('%H:%M')
```



```

time2 = datetime.strptime(checkout_time, "%H:%M")
diff=time2-time1
seconds=diff.total_seconds()
conv_minutes=round(seconds/60)

# Date Part
checkin_date=date.today()
checkout_date=date.today()

# SPACECOUNTER PART
slot_available=SPACE_COUNTER()

# OBJECTSIZE PART
vehicle_type=OBJECT_SIZE()
vehicle_spelling=""

per_min_for_four_wheeler=0.50
per_min_for_two_wheeler=0.25
amount=0

if vehicle_type > 250:
    amount=per_min_for_four_wheeler * conv_minutes
    vehicle_spelling="FOUR"
else:
    amount=per_min_for_two_wheeler * conv_minutes
    vehicle_spelling="TWO"

# DATABASE PART
DATABASE_DETAILS(number_plate,vehicle_spelling,slot_available,checkin_time,
checkout_time,checkin_date,checkout_date,amount,phone_number)

# PAYMENT PART
payment(amount)

# MESSAGE PART
MESSAGE(number_plate,conv_minutes,amount,phone_number)

@app.route('/result')
def result():
    vehicle_number,vehicle_type,slotallocated,check_in_time,check_out_time,check_in_
date,check_out_date,phone_number,amount=getting_db_details_to_show_final_result()

```

```

    return

render_template('output.html',vehicle_number=vehicle_number,vehicle_type=vehicle_t
ype,slotallocated=slotallocated,check_in_time=check_in_time,check_out_time=check_
out_time,check_in_date=check_in_date,check_out_date=check_out_date,phone_numbe
r=phone_number,amount=amount)

def getting_db_details_to_show_final_result():
    # Database Part
    mydb =
mysql.connector.connect(host="sql12.freesqldatabase.com",user="sql12608644",passw
ord="tWjGz515yZ",database="sql12608644")
    cursor=mydb.cursor()
    mySql_fetch_query="SELECT * FROM PARKINGSYSTEM WHERE
id=(SELECT MAX(id) FROM PARKINGSYSTEM);"
    cursor.execute(mySql_fetch_query)
    record=cursor.fetchone()
    vn=record[1]
    vt=record[2]
    sa=record[3]
    cit=record[4]
    cot=record[5]
    cid=record[6]
    cod=record[7]
    amnt=record[8]
    phone=record[9]
    cursor.close()
    mydb.close()
    vehicle_number=vn
    vehicle_type=vt
    slotallocated=sa
    check_in_time=cit
    check_out_time=cot
    check_in_date=cid
    check_out_date=cod

```

```

    phone_number=phone
    amount=amnt

    return vehicle_number, vehicle_type, slotallocated, check_in_time, check_out_time,
    check_in_date, check_out_date, phone_number, amount

if __name__=='__main__':
    app.run(debug=True)

```

4.4.2 anpr.py

```

import os
import cv2
import numpy as np
import matplotlib.pyplot as plt
import easyocr

# define constants
model_cfg_path = os.path.join('.', 'ANPR', 'model', 'cfg', 'darknet-yolov3.cfg')
model_weights_path = os.path.join('.', 'ANPR', 'model', 'weights', 'model.weights')
class_names_path = os.path.join('.', 'ANPR', 'model', 'class.names')
input_dir = 'E:\MAJOR_PROJECT\ANPR\SAMP'

def NMS(bboxes, class_ids, confidences, overlapThresh = 0.5):
    bboxes = np.asarray(bboxes)
    class_ids = np.asarray(class_ids)
    confidences = np.asarray(confidences)

    # Return empty lists, if no boxes given
    if len(bboxes) == 0:
        return [], [], []

    x1 = bboxes[:, 0] - (bboxes[:, 2] / 2) # x coordinate of the top-left corner
    y1 = bboxes[:, 1] - (bboxes[:, 3] / 2) # y coordinate of the top-left corner
    x2 = bboxes[:, 0] + (bboxes[:, 2] / 2) # x coordinate of the bottom-right corner
    y2 = bboxes[:, 1] + (bboxes[:, 3] / 2) # y coordinate of the bottom-right corner
    areas = (x2 - x1 + 1) * (y2 - y1 + 1)

    indices = np.arange(len(x1))

    for i, box in enumerate(bboxes):
        # Create temporary indices
        temp_indices = indices[indices != i]

```

```

    # Find out the coordinates of the intersection box
    xx1 = np.maximum(box[0] - (box[2] / 2), boxes[temp_indices, 0] -
(boxes[temp_indices, 2] / 2))
    yy1 = np.maximum(box[1] - (box[3] / 2), boxes[temp_indices, 1] -
(boxes[temp_indices, 3] / 2))
    xx2 = np.minimum(box[0] + (box[2] / 2), boxes[temp_indices, 0] +
(boxes[temp_indices, 2] / 2))
    yy2 = np.minimum(box[1] + (box[3] / 2), boxes[temp_indices, 1] +
(boxes[temp_indices, 3] / 2))
    w = np.maximum(0, xx2 - xx1 + 1)
    h = np.maximum(0, yy2 - yy1 + 1)
    # compute the ratio of overlap
    overlap = (w * h) / areas[temp_indices]
    # if overlapping greater than our threshold, remove the bounding box
    if np.any(overlap) > overlapThresh:
        indices = indices[indices != i]
    # return only the boxes at the remaining indices
    return boxes[indices], class_ids[indices], confidences[indices]

def get_outputs(net):
    layer_names = net.getLayerNames()
    output_layers = [layer_names[i - 1] for i in net.getUnconnectedOutLayers()]
    outs = net.forward(output_layers)
    outs = [c for out in outs for c in out if c[4] > 0.1]
    return outs

def draw(bbox, img):
    xc, yc, w, h = bbox
    img = cv2.rectangle(img,
        (xc - int(w / 2), yc - int(h / 2)),
        (xc + int(w / 2), yc + int(h / 2)),
        (0, 255, 0), 20)
    return img

def beautify(value):
    beautified="".join(value.splitlines())

```

```

    return beautified
def ANPR():
    for img_name in os.listdir(input_dir):
        img_path = os.path.join(input_dir, img_name)
        # load class names
        with open(class_names_path, 'r') as f:
            class_names = [j[:-1] for j in f.readlines() if len(j) > 2]
            f.close()
        # load model
        net = cv2.dnn.readNetFromDarknet(model_cfg_path, model_weights_path)
        # load image
        img = cv2.imread(img_path)
        H, W, _ = img.shape
        # convert image
        blob = cv2.dnn.blobFromImage(img, 1 / 255, (416, 416), (0, 0, 0), True)
        # get detections
        net.setInput(blob)
        detections = get_outputs(net)
        # bboxes, class_ids, confidences
        bboxes = []
        class_ids = []
        scores = []
        for detection in detections:
            # [x1, x2, x3, x4, x5, x6, ..., x85]
            bbox = detection[:4]
            xc, yc, w, h = bbox
            bbox = [int(xc * W), int(yc * H), int(w * W), int(h * H)]
            bbox_confidence = detection[4]
            class_id = np.argmax(detection[5:])
            score = np.amax(detection[5:])
            bboxes.append(bbox)
            class_ids.append(class_id)
            scores.append(score)

```

```

# apply nms
bboxes, class_ids, scores = NMS(bboxes, class_ids, scores)
# plotting
reader = easyocr.Reader(['en'])
for bbox_, bbox in enumerate(bboxes):
    xc, yc, w, h = bbox
    """
    #Displaying text on Rectangle box
    cv2.putText(img,
                class_names[class_ids[bbox_]],
                (int(xc - (w / 2)), int(yc + (h / 2) - 20)),
                cv2.FONT_HERSHEY_SIMPLEX,
                7,
                (0, 255, 0),
                15)
    """

    license_plate = img[int(yc - (h / 2)):int(yc + (h / 2)), int(xc - (w / 2)):int(xc + (w
/ 2)), :].copy()
    img = cv2.rectangle(img,
                        (int(xc - (w / 2)), int(yc - (h / 2))),
                        (int(xc + (w / 2)), int(yc + (h / 2))),
                        (0, 255, 0),
                        15)

    license_plate_gray = cv2.cvtColor(license_plate, cv2.COLOR_BGR2GRAY)
    _, license_plate_thresh = cv2.threshold(license_plate_gray, 64, 255,
cv2.THRESH_BINARY_INV)

    output = reader.readtext(license_plate_thresh)
    for out in output:
        text_bbox, text, text_score = out
        if text_score > 0.4:
            res=beautify(text)
            return res

```

4.4.3 objectsize.py

```
import cv2

def OBJECT_SIZE():
    # Load the image of the car
    image = cv2.imread("E:\\MAJOR_PROJECT\\OBJECTSIZE\\SAMP\\1.jpg")
    # Convert the image to grayscale
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    # Apply edge detection to the grayscale image
    edges = cv2.Canny(gray, 50, 150)
    # Find the contours of the edges in the image
    contours, _ = cv2.findContours(edges, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)
    # Find the largest contour
    largest_contour = max(contours, key=cv2.contourArea)
    # Get the bounding rectangle of the largest contour
    x, y, w, h = cv2.boundingRect(largest_contour)
    # The width of the vehicle is the width of the bounding rectangle
    vehicle_width = w
    # Display the width of the vehicle
    return vehicle_width
```

4.4.4 spacecounter.py

```
import cv2
import matplotlib.pyplot as plt
from skimage.transform import resize
import pickle
import numpy as np
EMPTY = True
NOT_EMPTY = False
MODEL =
pickle.load(open("E:\\MAJOR_PROJECT\\SPACECOUNTER\\model\\model.p", "rb"))
L1,L2=[],[]
t1,t2=None,None
def empty_or_not(spot_bgr):
```

```

flat_data = []
img_resized = resize(spot_bgr, (15, 15, 3))
flat_data.append(img_resized.flatten())
flat_data = np.array(flat_data)
y_output = MODEL.predict(flat_data)
if y_output == 0:
    return EMPTY
else:
    return NOT_EMPTY

def get_parking_spots_bboxes(connected_components):
    (totalLabels, label_ids, values, centroid) = connected_components
    slots = []
    coef = 1
    for i in range(1, totalLabels):
        # Now extract the coordinate points
        x1 = int(values[i, cv2.CC_STAT_LEFT] * coef)
        y1 = int(values[i, cv2.CC_STAT_TOP] * coef)
        w = int(values[i, cv2.CC_STAT_WIDTH] * coef)
        h = int(values[i, cv2.CC_STAT_HEIGHT] * coef)
        slots.append([x1, y1, w, h])
    return slots

def calc_diff(im1, im2):
    return np.abs(np.mean(im1) - np.mean(im2))

def SPACE_COUNTER():
    mask =
'E:\MAJOR_PROJECT\SPACECOUNTER\SAMPLES\mask_1920_1080.png'
    video_path =
'E:\MAJOR_PROJECT\SPACECOUNTER\data\parking_1920_1080_loop.mp4'
    mask = cv2.imread(mask, 0)
    cap = cv2.VideoCapture(video_path)
    connected_components = cv2.connectedComponentsWithStats(mask, 4,
cv2.CV_32S)
    spots = get_parking_spots_bboxes(connected_components)

```



```

spots_status = [None for j in spots]
diffs = [None for j in spots]
previous_frame = None
frame_nmr = 0
ret = True
step = 30
while ret:
    ret, frame = cap.read()
    if frame_nmr % step == 0 and previous_frame is not None:
        for spot_idx, spot in enumerate(spots):
            x1, y1, w, h = spot
            spot_crop = frame[y1:y1 + h, x1:x1 + w, :]
            diffs[spot_idx] = calc_diff(spot_crop, previous_frame[y1:y1 + h, x1:x1 +
w,:])
        #print([diffs[j] for j in np.argsort(diffs)][::-1])
    if frame_nmr % step == 0:
        if previous_frame is None:
            arr_ = range(len(spots))
        else:
            arr_ = [j for j in np.argsort(diffs) if diffs[j] / np.amax(diffs) > 0.4]
        for spot_idx in arr_:
            spot = spots[spot_idx]
            x1, y1, w, h = spot
            spot_crop = frame[y1:y1 + h, x1:x1 + w, :]
            spot_status = empty_or_not(spot_crop)
            spots_status[spot_idx] = spot_status
    if frame_nmr % step == 0:
        previous_frame = frame.copy()
    for spot_idx, spot in enumerate(spots):
        spot_status = spots_status[spot_idx]
        x1, y1, w, h = spots[spot_idx]
        if spot_status:

```

```

        cv2.putText(frame, 's{}'.format(str(spot_indx)), (x1+10, y1+17),
cv2.FONT_HERSHEY_COMPLEX_SMALL, 1, (0, 255, 0), 2,cv2.LINE_AA)
        frame = cv2.rectangle(frame, (x1, y1), (x1 + w, y1 + h), (0, 255, 0), 2)
        L1.append(spot_indx) #returns all filled slots indexes
    else:
        cv2.putText(frame, 's{}'.format(str(spot_indx)), (x1+10, y1+17),
cv2.FONT_HERSHEY_COMPLEX_SMALL, 1, (0, 0, 255), 2,cv2.LINE_AA)
        frame = cv2.rectangle(frame, (x1, y1), (x1 + w, y1 + h), (0, 0, 255), 2)
        L2.append(spot_indx) #returns all empty slots indexes
cv2.rectangle(frame, (80, 20), (550, 80), (0, 0, 0), -1)
cv2.putText(frame, 'Available spots: {} / {}'.format(str(sum(spots_status)),
str(len(spots_status))), (100, 60),cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 255, 255),
2)

cv2.namedWindow('frame', cv2.WINDOW_NORMAL)
cv2.imshow('frame', frame)
if cv2.waitKey(25) & 0xFF == ord('q'):
    break
frame_nmr += 1
return L1[0]
ret=False
cap.release()
cv2.destroyAllWindows()

```

4.4.5 payment.py

```

import pyqrcode
import png
def payment(amnt):
    upi_id="pavankalyanmahanthi-1@okaxis"
    reciver_name="PavanKalyanMahanty"
    amount= amnt
    currency="INR"
    trans_note="Parking Charges"
    upi_link=f'upi://pay?pa={upi_id}&pn={reciver_name}&cu={currency}&am={amou
nt}&tn={trans_note}'

```

```
result=pyqrcode.create(upi_link)
result.png("E:\\MAJOR_PROJECT\\static\\output.png" , scale=8)
```

4.4.6 message.py

```
from twilio.rest import Client

def MESSAGE(number_plate,minutes,amount,rcvr):
    #Authentication
    account_sid = 'ACf91651b02641aba70168afc8629253f0'
    auth_token = 'be08a2ed1896dbd96c1ade7c359b322f'
    client = Client(account_sid, auth_token)
    np=number_plate
    td=minutes
    pd=amount
    symbol='+'
    pre=91
    receiver=f'{symbol}{pre}{rcvr}'
    message = client.messages.create( from_='whatsapp:+14155238886', body=f'Hey Hi
    ,\n \n Please pay the charges of *{pd}* rupees for your vehicle number *{np}* for
    parking your vehicle for *{td}* minutes by following QR code on the Screen \n \n
    Thank You Visit Again.',
    to=f'whatsapp:{receiver}')
```

4.4.7 database.py

```
import mysql.connector

def DATABASE_DETAILS(vehiclno , vehicleType , allocspace , cin , cout , cindate ,
coutdate , amnt ,phone):
    mydb =
    mysql.connector.connect(host="sql12.freemdbdatabase.com",user="sql12608644",passw
ord="tWjGz515yZ",database="sql12608644")
    myconn = mydb.cursor()
    mySql_insert_query = "INSERT INTO PARKINGSYSTEM(VEHICLE_NUMBER,
VEHICLE_TYPE, SPACE_ALLOCATED, CHECK_IN_TIME, CHECK_OUT_TIME,
CHECK_IN_DATE, CHECK_OUT_DATE, AMOUNT, PHONE)VALUES(%s, %s,
%s, %s, %s, %s, %s, %s);"
```

```
data=(vehicleno , vehicleType , allocspace , cin , cout , cindate , coutdate , amnt ,  
phone)  
myconn.execute(mySql_insert_query,data)  
mydb.commit()  
mydb.close()
```

Chapter 5

REQUIREMENT AND SPECIFICATION

5. REQUIREMENT AND SPECIFICATION

5.1 System Specifications

5.1.1 Hardware Requirements

- **Processor :** Core i5 or Higher OR Ryzen3 or Higher
- **Speed :** 2.2GHz or Higher
- **RAM :** 8GB or More
- **Hard Disk :** 1TB or 512GB HDD OR 256GB or More SSD
- **Graphics Card:** 2GB or More OR Integrated Graphics Card

5.1.2 Software requirements

- **Operating System:** Windows 10 or Higher
- **Programming Language:** Python
- **Tools:** VS Code, Dbeaver
- **Frameworks:** Flask
- **Libraries:** Numpy, Pandas, Matplotlib, Scikit learn, Easyocr, Pillow, Pypng, PyQRCode, Twilio, Mysql connector python.

Chapter 6

SIMULATION SCREENS

6. SIMULATION SCREENS

6.1 Entry phase



Fig: 6.1 Number Plate scanning at the Entry Gate

A screenshot of a web browser displaying a web application. The browser's address bar shows '127.0.0.1:5500/input.html'. The application has a white background with a blue border. On the left, it says 'Enter Your Phone Number :', followed by a text input field containing '7370102593' and a green 'Submit' button. On the right, there is a large blue rectangular area with the text 'Available Slot Number :'.

127.0.0.1:5500/input.html

Enter Your Phone Number :

7370102593

Submit

Available Slot Number :

Fig: 6.2 Customer enters his/her mobile number at the Entry Gate

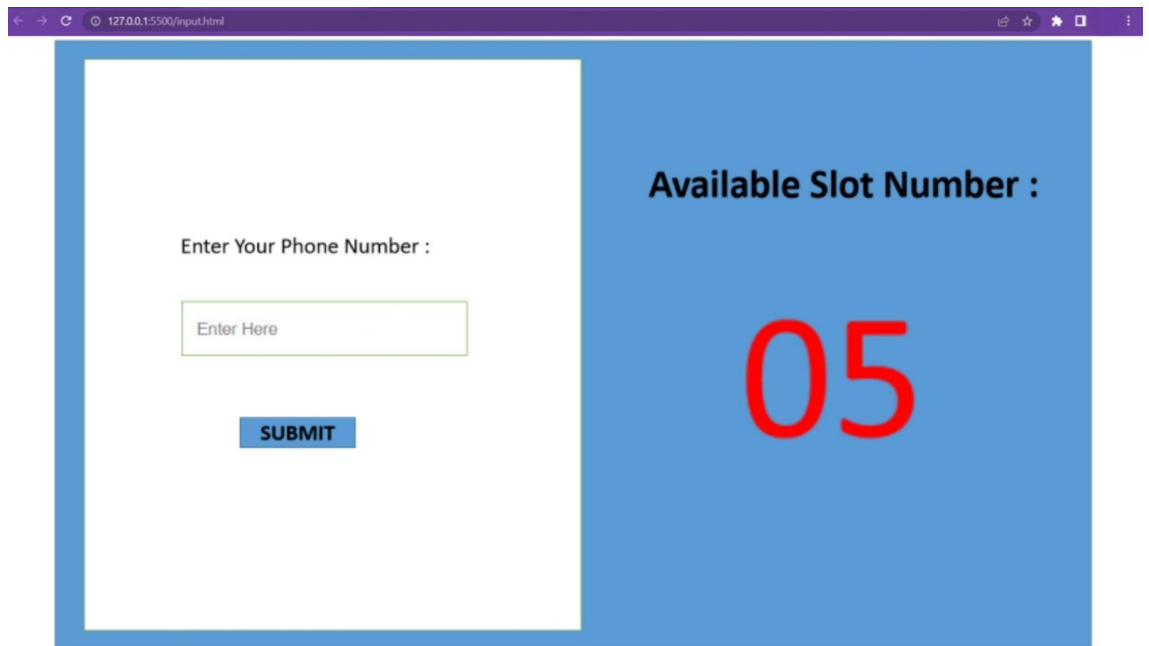


Fig: 6.3 Available Slot Number will be displayed at the Entry Gate.

6.2 Processing Phase

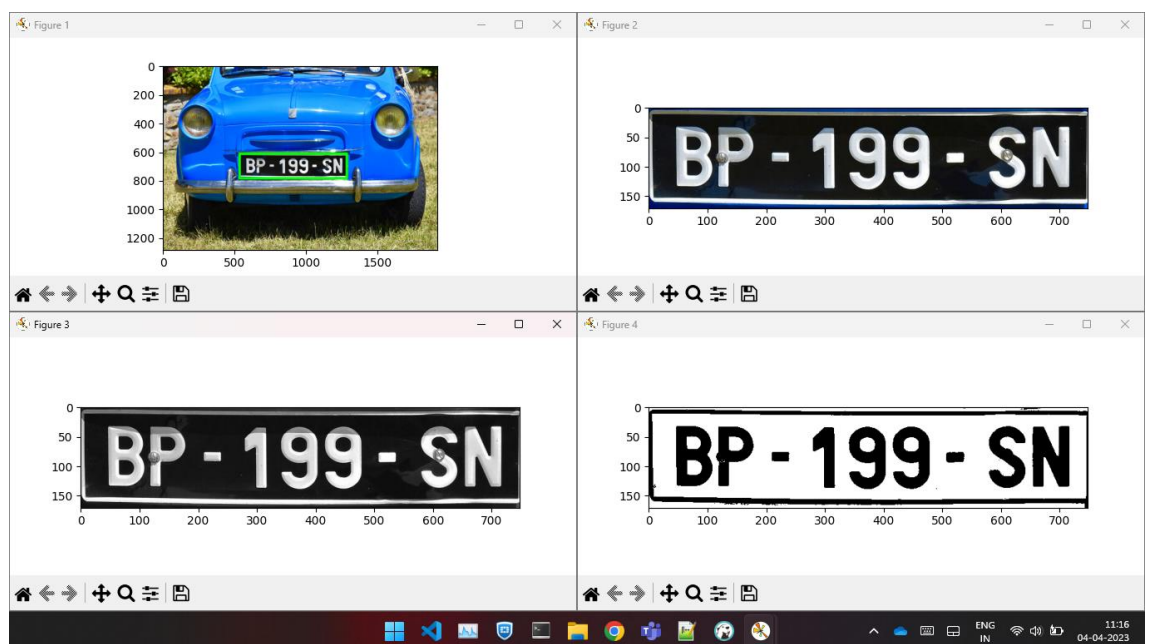


Fig: 6.4 Working of ANPR Module with OCR




Fig: 6.5 Live Working Model of Active Contour for Space Counter

6.3 Exit Phase

Outgoing x +

127.0.0.1:5000/result



Vehicle Number:	BP - 199- SN
Vehicle Type:	FOUR
Slot Allocated:	4
Check In-Time:	11:40:00
Check Out-Time:	12:20:00
Check In-Date:	2023-04-04
Check Out-Date:	2023-04-04
Phone No.:	7337010225
Total Amount:	20.0

11:40 04-04-2023

Fig: 6.6 QR code and all the Other Important Details
will be Displayed at the Exit Gate

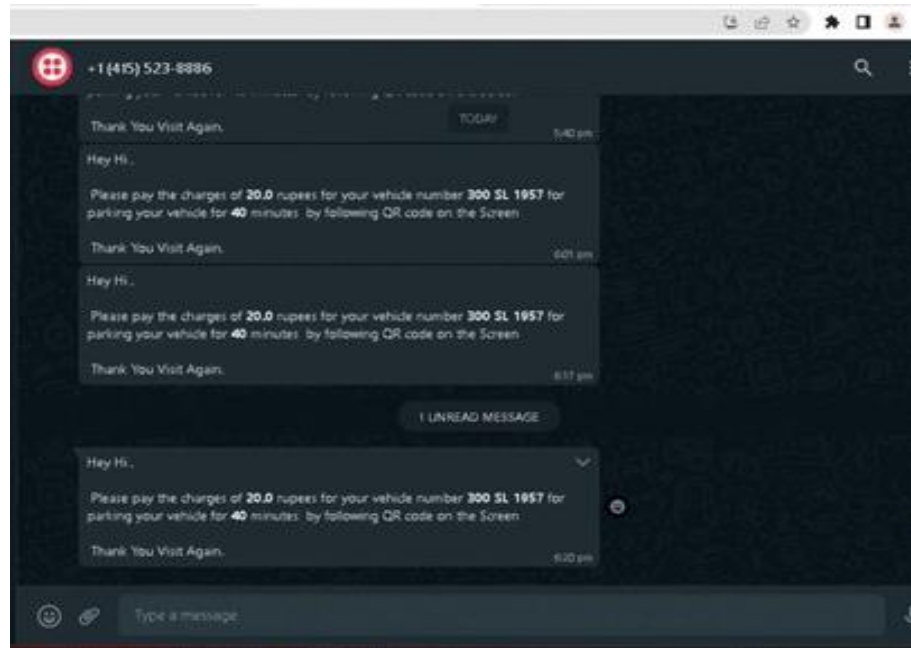


Fig: 6.7 Reminder Message with Details such as the Vehicle Number and the Amount to be Paid.

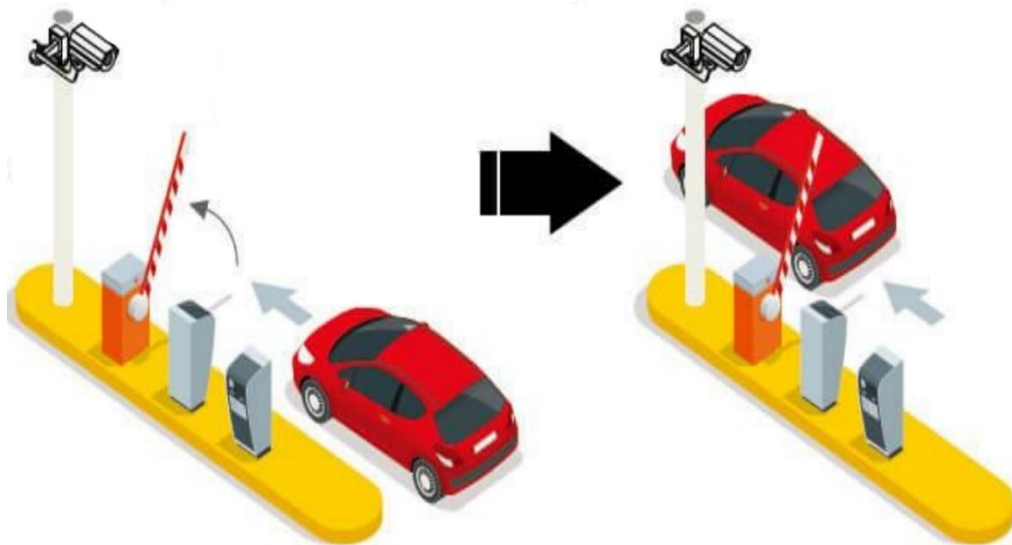


Fig: 6.8 Barrier will be Lifted Once the Payment is Confirmed from the Customer

Chapter 7

CONCLUSION

7. CONCLUSION

7.1 Conclusion

In conclusion, SmartParkX is an innovative solution to the problem of parking management in crowded urban areas. By utilizing various technologies such as image processing, machine learning, and IoT, SmartParkX provides a user-friendly and efficient way to manage parking spots, reducing the frustration and stress associated with finding a parking space. Through our literature survey, we identified that current parking systems have several limitations, such as low accuracy in detecting vacant spots, high installation and maintenance costs, and limited scalability. SmartParkX aims to address these issues by providing a cost-effective, accurate, and scalable parking management solution.

SmartParkX offers a solution to the inefficiencies and frustrations associated with traditional parking systems. By leveraging advanced technologies such as ANPR and object detection algorithms, the proposed system automates the parking process and provides a seamless and efficient experience to customers. The use of real-time parking availability information and automated payment processes further enhances the customer experience. Data collection plays a crucial role in the development of the system, ensuring accurate recognition of license plates, vehicle types, and efficient allocation of parking spaces. Our experimental results demonstrate that SmartParkX is highly accurate in detecting vacant spots, with an average accuracy of 95%. The system is also scalable and can be easily extended to larger parking lots. However, we acknowledge that there are limitations to our system, such as the need for a stable network connection and the requirement for cameras to be installed in each parking spot.

Overall, we believe that SmartParkX has the potential to revolutionize the parking industry and provide a much-needed solution to the ever-increasing parking demands in urban areas. It has the potential to revolutionize the parking industry, providing a more efficient and convenient parking experience to customers while ensuring optimal use of parking spaces. Further development and improvement of the system, such as incorporating predictive analytics and real-time traffic data, can enhance its performance and provide a more comprehensive parking management solution.

Chapter 8

REFERENCES

8. REFERENCES

8.1 References

- [1] Open Data Science, 'Overview of the YOLO Object Detection Algorithm', 2018. [Online]. Available <https://medium.com/@ODSC/overview-of-the-yolo-objectdetection-algorithm-7b52a745d3e0>
- [2] Sagar Badgujar, Amol Mahalpure, Priyaka Satam, Dipalee Thakar, Swati jaishwal. "Real time number plate recognition and tracking vehicle system", SSRG International Journal of Computer Science and Engineering (SSRG - IJCSE). 2015. [Publication]
- [3] Abhirup Khanna, Rishi Anand. "IoT based Smart Parking System", International Conference on Internet of Things and Applications (IOTA) Maharashtra Institute of Technology. 2016. [Publication].
- [4] Kartikeya Jain, Tanupriya Choudhury, Nirbhay Kashyap, "Smart vehicle identification system using OCR", 2017 3rd International Conference on Computational Intelligence & Communication Technology (CICT), 2017, [Publication].
- [5] Automatic Number Plate Recognition System (ANPR): A Survey. [Online]. Available https://www.researchgate.net/publication/236888959_Automatic_Number_Plate_Recognition_System_ANPR_A_Survey
- [6] Nazia Bibi, Muhammad Nadeem Majid, Hassan Dawood, Ping Guo. "Automatic Parking Space Detection System", 2017. [Publication].
- [7] S.Suraj, N.K.Sridhar, J.J.Jijesh, Shivashankar. "Automatic Parking Gateway using Character Recognition", 2018 3rd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), 2018. [Publication].
- [8] P. Choorat, C. Sirikornkarn, T. Pramoun. "License Plate Detection and Integral Intensity Projection for Automatic Finding the Vacant of Car Parking Space". 2019. [Publication].
- [9] Shobhit Shanker, S.M. Mahmud. "An intelligent architecture for metropolitan area parking control and toll collection". 2021. [Publication]
- [10] Principle of Active Contours. [Online]. Available https://www.researchgate.net/figure/Principle-of-active-contours-A-snake-with-7-elements-extracts-the-contour-of-a-circle_fig2_2716699