Project Report

1. INTRODUCTION :

1.1 Project Overview :

i).This an Artificial Intellegence based project used in car parking,Which helps to find any free place for parking your car.

1.2 Purpose:

i).This project helps you to reduce your time for searching for parking space.

2. IDEATION & PROPOSED SOLUTION

2.1 Problem Statement Definition:

AI enabled carparking using openCV2. Use ai to find is parking space available or not.

2.2 Empathy Map Canvas:

empathy map link:

2.3 Ideation & Brainstorming :

brain strome link:

2.4 Proposed Solution :

proposed solution link:

3. REQUIREMENT ANALYSIS

3.1 Functional requirement:

i).Anaconda navigator.

ii).Python packages:

a.OpenCV

b.CVZone

c.Numphy

d.Flask

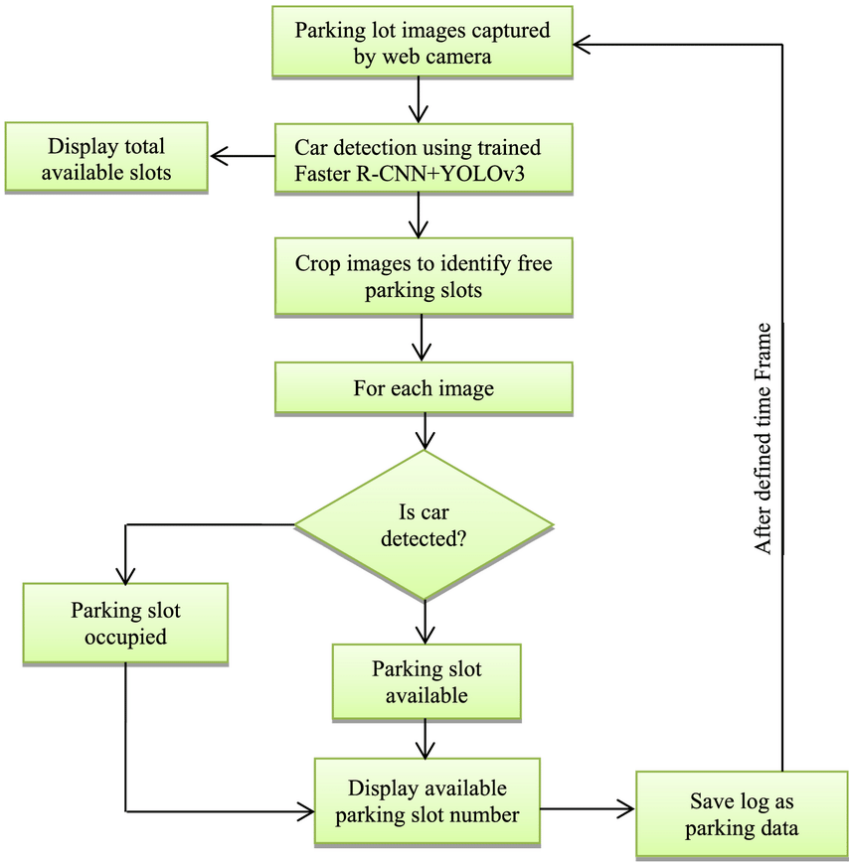
3.2 Non-Functional requirements :

i).You must have prior knowledge on below topics:

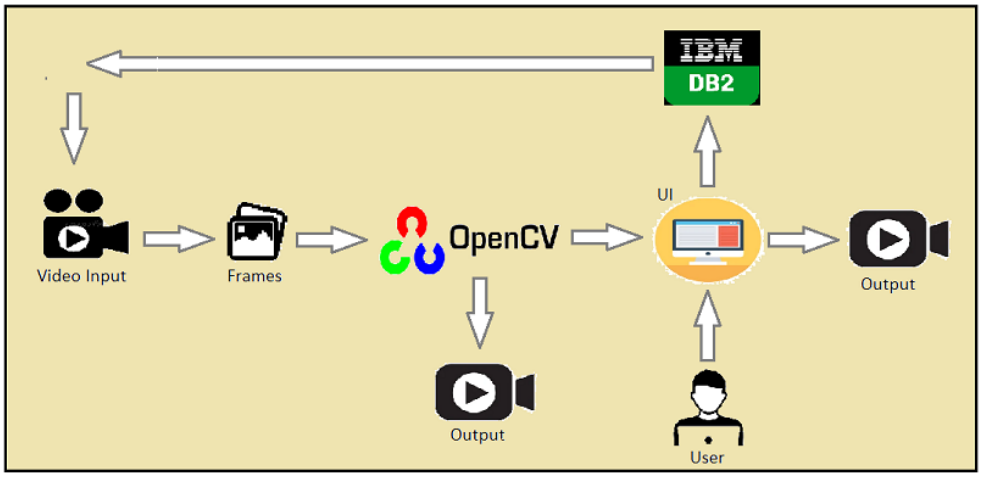
a.OpenCV.
b.CVZone
c.Flask

4. PROJECT DESIGN

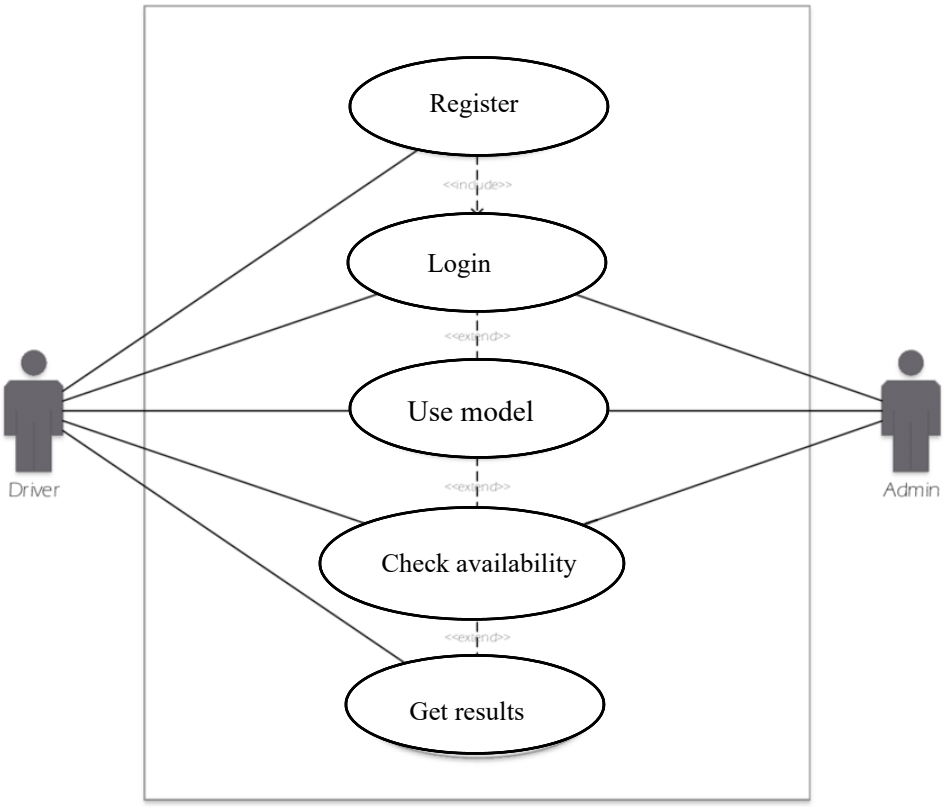
4.1 Data Flow Diagrams :



4.2 Solution & Technical Architecture :



4.3 User Stories:



5. CODING & SOLUTIONING (Explain the features added in the project along with code):

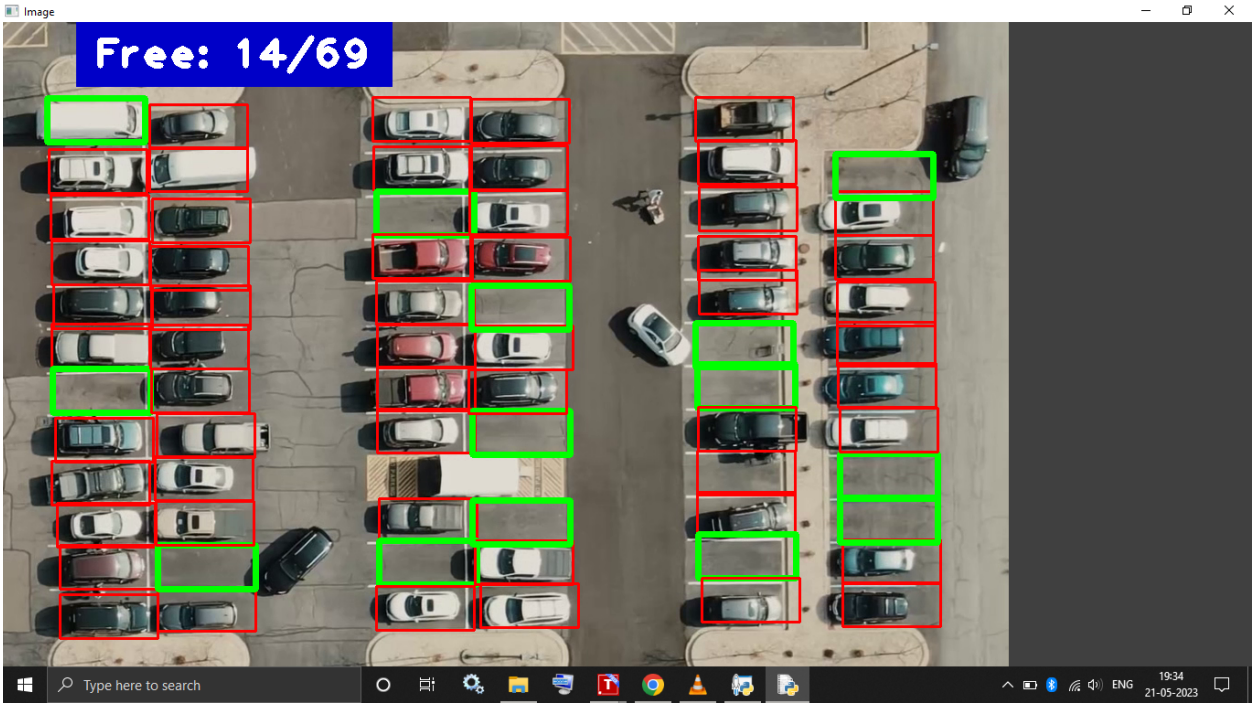
5.1 Feature 1:

Occupancy detection : Whether a particular parking place is occupied or not (in Real-Time)

5.2 Feature 2

Automatic car license plates recognition : Automatic Car's license plate recognition using efficient OCR integration.

6. RESULTS



7. ADVANTAGES & DISADVANTAGES

i).Advantages:

a.no time wasting.

b.ease of maintenance.

c.make our work simple.

d.reduce working and searching.

ii)Dis-advantages:

a.make people lazy.

b.software reliability and lability for damage.

c.high cost.

d.people may adaptive which make their thinking and working hard.

8. CONCLUSION

Thus the project we've created had worked and gave result succesfully.

9. FUTURE SCOPE

In future the number of car will increase nemorously,which make finding parking space for our car difficult than other.So the future scope for AI based car parking system may plays an major role in coming years.The requirement for ai based parking plot system will essential in every where.

10. APPENDIX

I).Source code:

a).model building:

selectingROI.py - C:\Users\V.S.D.M\Desktop\NM PROJECT\Model building\selectingROI.py (3.9.0)

File Edit Format Run Options Window Help

```python
import cv2
import pickle

# define the width and height of ROI
width,height=107,48

#creating an empty file and loading to a veriable & Creating an empty list
try:
    with open('E:\\NM project\\Model building\\parkingSlotPosition','rb')as f:
        posList=pickle.load(f)
except:
    posList=[]

def mouseClick(events, x,y, flags, params):
    # adding ROI values to posList
    if events == cv2.EVENT_LBUTTONDOWN:
        posList.append((x,y))
    #removing unwanted ROI from posList
    if events == cv2.EVENT_RBUTTONDOWN:
        for i, pos in enumerate(posList):
          x1,y1=pos
          if x1 < x < x1 + width and y1 < y < y1 + height:
            posList.pop(i)
    #saving the posList values to parkingSlotPossition file
    with open('parkingSlotPosition','wb') as f:
        pickle.dump(posList, f)

while True:
    img=cv2.imread('carParkImg.png')
    for pos in posList:
        cv2.rectangle(img,pos,(pos[0]+width,pos[1]+height),(255,255,255),2)

    cv2.imshow("Image",img)
    cv2.setMouseCallback("Image",mouseClick)
    cv2.waitKey(1)
```

Ln: 1 Col: 0

Type here to search ENG 20:02 21-05-2023

b).VID process:

i).

VID process.py - C:\Users\V.S.D.M\Desktop\NM PROJECT\Model building\VID process.py (3.9.0)

File Edit Format Run Options Window Help

```python
import cv2
import pickle
import cvzone
import numpy as np

#video feed
cap=cv2.VideoCapture('carParkingInput.mp4')

#loading the ROI from parkingslotposition file
with open('parkingSlotPosition','rb')as f:
    posList=pickle.load(f)

#define width and height
width,height=107,48

def checkParkingSpace(imgPro):
    spaceCounter=0
    for pos in posList:
        x,y=pos
        #crop the image on ROI:
        ingCrop=imgPro[y:y +height,x+x +width]
        # counting the pixel values from cropped image:
        if count <900:
            color=(0,255,0)
            thickness=5
            spaceCounter+=1
        else:
            color=(0,0,255)
            thickness=2
        #draw the rectangel based on the ondition defined above:
        cv2.rectangle(img,pos,(pos[0]+width,pos[1]+height),color,thickness)
    #display the available parking slot count / total parking slot count:
    cvzone.putTextRect(img,f'Free:{spaceCounter}/{len(posList)}',(100,50),scale=3,thickness=5,offset=0,color=(0,200,0))

    # loop the video:
    while True:
        #looping the video
        if cap.get(cv2.CAP_PROP_POS_FRAMES)==cap.get(cv2.CAP_PROP_FRAME_COUNT):
            cap.set(cv2.CAP_PROP_POS_FRAMES,0)

        #reading frame by frame from video:
```

Ln: 1 Col: 0

Type here to search ENG 20:04 21-05-2023

ii).

VID process.py - C:\Users\V.S.D.M\Desktop\NM PROJECT\Model building\VID process.py (3.9.0)

File Edit Format Run Options Window Help

```python
def checkParkingSpace(imgPro):
    spaceCounter=0
    for pos in posList:
        x,y=pos
        #crop the image on ROI:
        ingCrop=imgPro[y:y +height,x+x +width]
        # counting the pixel values from cropped image:
        if count <900:
            color=(0,255,0)
            thickness=5
            spaceCounter+=1
        else:
            color=(0,0,255)
            thickness=2
        #draw the rectangel based on the ondition defined above:
        cv2.rectangle(img,pos,(pos[0]+width,pos[1]+height),color,thickness)
    #display the available parking slot count / total parking slot count:
    cvzone.putTextRect(img,f'Free:{spaceCounter}/{len(posList)}',(100,50),scale=3,thickness=5,offset=0,color=(0,200,0))

    # loop the video:
    while True:
        #looping the video
        if cap.get(cv2.CAP_PROP_POS_FRAMES)==cap.get(cv2.CAP_PROP_FRAME_COUNT):
            cap.set(cv2.CAP_PROP_POS_FRAMES,0)

        #reading frame by frame from video:
        success,img=cap.read()
        #convertinf to fray scale image:
        imgGray=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
        imgBlur=cv2.GaussianBlur(imgGray,(3,3),1)
        #applying threshold to image:
        imgThreshold=cv2.adaptiveThreshold(imgBlur,255,cv2.ADAPTIVE_THRESH_GAUSSIAN_C,cv2.THRESH_BINARY_INV,25,16)
        imgMedian=cv2.medianBlur(imgTHreshold,5)
        kernal=np.ones((3,3),np.unit8)
        imgDilate=cv2.dilate(imgMedian,kernal,iteration=1)
        #passing dilate image to the function:
        checkParkingSpae(imgDilate)
        cv2.imshow("Image",img)
        cv2.waitKey(10)
```

Ln: 1 Col: 0

Type here to search ENG 20:04 21-05-2023

II).app.py

i)

app.py - C:\Users\V.S.D.M\Desktop\NM PROJECT\flask\app.py (3.9.0)

File Edit Format Run Options Window Help

```python
from flask import Flask, render_template
import cv2
import pickle
import cvzone
import ibm_db
import numpy as np

app = Flask(__name__)

conn=ibm_db.connect("DATABASE=bludb;HOSTNAME=54a2f15b-5c0f-46df-8954-7e38e612c2bd.clogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=32733;SECURITY=SSL;SSLServerCerti
print("connected")



@app.route('/')
def project():
    return render_template('index.html')


@app.route('/hero')
def home():
    return render_template('index.html')


@app.route('/model')
def login():
    return render_template('model.html')


@app.route('/predict_live')
def liv_pred():
    # Video feed
    cap = cv2.VideoCapture('carParkingInput.mp4')
    with open('parkingSlotPosition', 'rb') as f:
        posList = pickle.load(f)
    width, height = 107, 48
    def checkParkingSpace(imgPro):
        spaceCounter = 0
        for pos in posList:
            x, y = pos
            imgCrop = imgPro[y:y + height, x:x + width]
```

Ln: 1 Col: 0

Type here to search ENG 20:10 21-05-2023

ii).

app.py - C:\Users\V.S.D.M\Desktop\NM PROJECT\flask\app.py (3.9.0)

File Edit Format Run Options Window Help

```python
        spaceCounter = 0
        for pos in posList:
            x, y = pos
            imgCrop = imgPro[y:y + height, x:x + width]
            # cv2.imshow(str(x * y), imgCrop)
            count = cv2.countNonZero(imgCrop)
            if count < 900:
                color = (0, 255, 0)
                thickness = 5
                spaceCounter += 1
            else:
                color = (0, 0, 255)
                thickness = 2
            cv2.rectangle(img, pos, (pos[0] + width, pos[1] + height), color, thickness)
            """cvzone.putTextRect(img, str(count), (x, y + height - 3), scale=1,
                                  thickness=2, offset=0, colorR=color)"""
        cvzone.putTextRect(img, f'Free: {spaceCounter}/{len(posList)}',(100, 50), scale=3,
                           thickness=5, offset=20, colorR=(200, 0, 0))
    while True:
        if cap.get(cv2.CAP_PROP_POS_FRAMES) == cap.get(cv2.CAP_PROP_FRAME_COUNT):
            cap.set(cv2.CAP_PROP_POS_FRAMES, 0)
        success, img = cap.read()
        imgGray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
        imgBlur = cv2.GaussianBlur(imgGray, (3, 3), 1)
        imgThreshold = cv2.adaptiveThreshold(imgBlur, 255, cv2.ADAPTIVE_THRESH_GAUSSIAN_C,
                                             cv2.THRESH_BINARY_INV, 25, 16)
        imgMedian = cv2.medianBlur(imgThreshold, 5)
        kernel = np.ones((3, 3), np.uint8)
        imgDilate = cv2.dilate(imgMedian, kernel, iterations=1)
        checkParkingSpace(imgDilate)
        cv2.imshow("Image", img)
        # cv2.imshow("ImageBlur", imgBlur)
        # cv2.imshow("ImageThres", imgMedian)
        if cv2.waitKey(1) & 0xFF == ord('q'):
            break


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

Ln: 10 Col: 153

Type here to search ENG 20:11 21-05-2023

III).Source code links:

1).Source code and html templates google drive link: https://drive.google.com/drive/folders/1PUrKX2N9LZsAcp8cSYbEUvmaPOHVi_e3?usp=drive_link

2).GitHub & Project Video Demo Link:

i).Demo video google drive link: https://drive.google.com/file/d/1Gn6dMIn_7LMVufmB-CiEDJtaGsMEPzb3Z/view?usp=drive_link

ii).Git hub repositary link : https://github.com/DineshkumarV01/Al-enabled-car-parking-using-open-CV/upload/main