PROJECT REPORT

18CSE390T – COMPUTER VISION

(2018 Regulation)

II Year/ III Semester

Academic Year: 2022 -2023

By

Gunti Meghanath (RA2011026010219)

Nekkanti Shivram (RA2011026010237)

Harshitha Kambham (RA2011026010235)

Under the guidance of

Dr. A. ALICE NITHYA

Associate Professor

Department of Computational Intelligence

SRM INSTITUTE OF SCIENCE & TECHNOLOGY
(Deemed to be University u/s 3 of UGC Act, 1956)

FACULTY OF ENGINEERING AND TECHNOLOGY

SCHOOL OF COMPUTING

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

Kattankulathur, Kancheepuram

NOVEMBER 2022

BONAFIDE

This is to certify that **18CSE390T – COMPUTER VISION project report** titled “**…………..**” is the bonafide work of **<< students name & reg no.>>** who undertook the task of completing the project within the allotted time.

Signature of the Guide
Dr. A. Alice Nithya
Associate Professor
Department of CINTEL
SRM Institute of Science and Technology

ABSTRACT

Through this project,a Parking Space Counter is developed. A track is kept of total cars are present and how many spaces are vacant to park. Drivers often encounter problems associated with locating empty parking slots in parking areas. This paper presents a smart parking lot management system which operates using image processing. An image processing algorithm is used to detect empty parking areas from aerial images of the parking space. The algorithm processes the image, extracts occupancy information concerning spots, and their positions thereof. The system also reports if individual parking spots are occupied or otherwise. Occupancy information is made available to newly arriving drivers by projecting it unto large displays positioned at vantage points near the vicinity. The smart parking lot management system reduces the stress and time wastage associated with car parking and makes management of such areas less costly.

MODULE DESCRIPTION

Data Processing:

- User inputs file name for a video, a still image from the video, and a path for the output file of parking space coordinates.
- User clicks 4 corners for each spot they want tracked. Presses ‘q’ when all desired spots are marked.
- Video begins with the user provided boxes overlayed the video. Occupied spots initialized with red boxes, available spots with green.
    - Car leaves a space, the red box turns green.
    - Car drives into a free space, the green box turns red.

Image Processing Techniques:

LINE DETECTION:

To detect the parking spots, by taking advantage of the lines demarking the boundaries.

The Hough Transform is a popular feature extraction technique for detecting lines. OpenCV encapsulates the math of the Hough Transform into HoughLines(). Further abstraction in captured in HoughLinesP(), which is the probabilistic model of creating lines with the points that HoughLines() returns. For more info, check out the OpenCV Hough Lines tutorial.

img = cv2.imread(filename='examples/hough_lines/p_lots.jpg')

converted it to gray scale to reduce the info in the photo:

gray = cv2.cvtColor(src=img, code=cv2.COLOR_BGR2GRAY)

Gaussian blur to remove even more unnecessary noise:

blur_gray = cv2.GaussianBlur(src=gray, ksize=(5, 5), sigmaX=0)

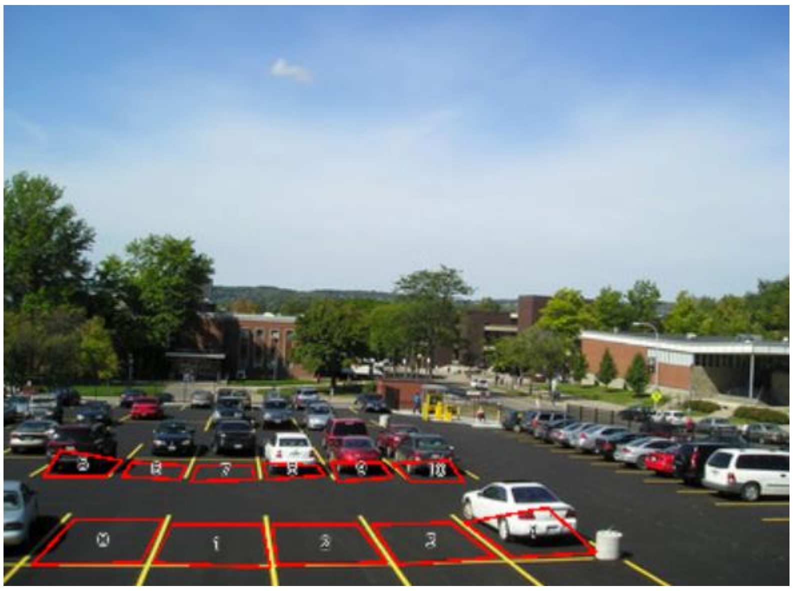
Detected the edges with Canny:

edges = cv2.Canny(image=blur_gray, threshold1=50, threshold1=150, apertureSize=3)

Drawing Rectangles:

In this program, we will draw a rectangle using the OpenCV function rectangle(). This function takes some parameters like starting coordinates, ending coordinates, color and thickness and the image itself.

```
Image = cv2.imread('testimage.jpg')
height, width, channels = image.shape
start_point = (0,0)
end_point = (width, height)
color = (0,0,255)
thickness = 5
image = cv2.rectangle(image, start_point, end_point, color, thickness)
cv2.imshow('Rectangle',image)
```



Finishing touches:

After drawing the rectangles,the area of each rectangle was examinated to see if there was a car in there or not.

By taking each (filtered and blurred) rectangle, determining the area, and doing an average on the pixels, I was able to tell when there wasn’t a car in the spot if the average was high (more dark pixels). The color of the bounding box accordingly was changed.

The code for drawing the rectangles and motion detection is pretty generic. It’s seperated out into classes and should be reusable outside of the context of a parking lot.

System Architecture with Explanation

The smart parking system proposed in this paper consists of three main components. These are the parking detection nodes with incorporated WiFi access points (APs) deployed within each major section of the parking facility, a wireless local area network (WLAN) integrated local base station, and a notification system for information delivery. The architecture of the proposed system is shown in Figure 1.

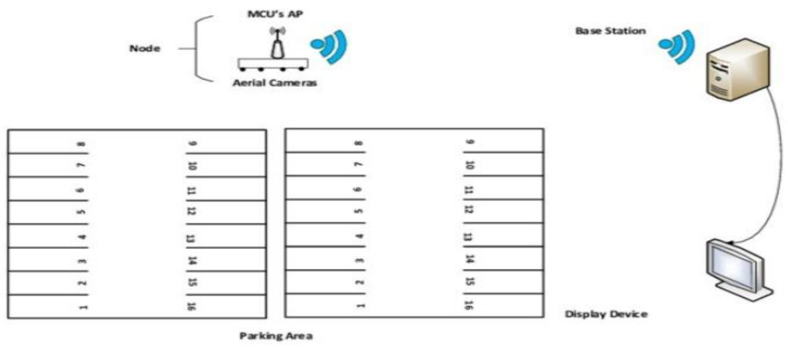


Fig 1Architecture of proposed parking management system.

The parking area is logically demarcated into major sections for easy deployment of parking detection nodes. Each major section of the area is equipped with a detection node. A node consists of wide field view cameras for taking snapshots of the assigned parking minor section at regular intervals and IEEE 802.11 b/g/n compatible microcontroller unit (MCU) for communication between the node and the base station. As shown in Figure 2, a node is affixed at a higher altitude close to the perimeter of its assigned parking section such that the node overlooks the area and the cameras are able to capture their individual region.

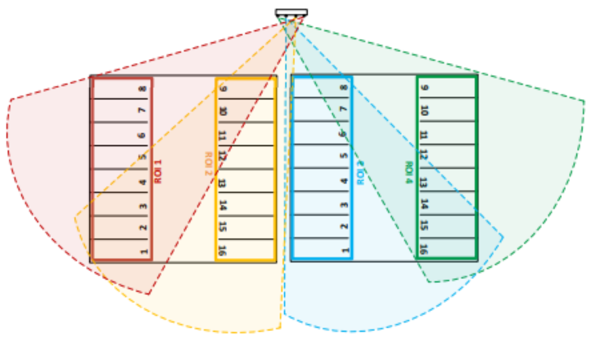


Fig. 2. Camera positions with their respective FOVs and assigned ROIs.

The operation of the system involved four major sub-operations as shown in Figure 3.

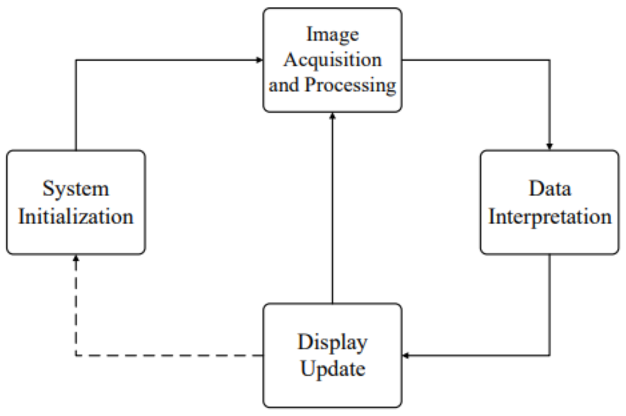


Fig. 3. Operational modules of proposed system.

These modules are system initialization, image acquisition and processing, data interpretation, and display update. The last three processes were then repeated as long as the system was active. The initialization of the parking guidance system takes place once, when the system is being set up for the first time or after a replacement of any of the systems module (shown by dashed arrow). During initialization, a refresh signal is sent from the node's controller to the image sensors in order to activate the process. The aerial cameras capture images of their assigned parking lot section whiles the area is empty. The images are transferred from the cameras to the MCU via the fast parallel image sensor interface

CODE

```
import cv2
import pickle

width, height = 107, 48

try:
    with open('CarParkPos', 'rb') as f:
        posList = pickle.load(f)
except:
    posList = []


def mouseClick(events, x, y, flags, params):
    if events == cv2.EVENT_LBUTTONDOWN:
        posList.append((x, y))
    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)

    with open('CarParkPos', '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, 0,
255), 2)

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

Main.py:

```
import cv2
import numpy as np
import pickle

rectW,rectH=107,48
cap=cv2.VideoCapture('carPark.mp4')

with open('carParkPos','rb') as f:
    posList=pickle.load(f)
frame_counter = 0
def check(imgPro):
    spaceCount=0
    for pos in posList:
        x,y=pos
        crop=imgPro[y:y+rectH,x:x+rectW]
        count=cv2.countNonZero(crop)
        if count<900:
            spaceCount+=1
            color=(0,255,0)
            thick=5
        else:
            color=(0,0,255)
            thick=2

        cv2.rectangle(img,pos,(x+rectW,y+rectH),color,thick)
    cv2.rectangle(img,(45,30),(250,75),(180,0,180),-1)
    cv2.putText(img,f'Free:
{spaceCount}/{len(posList)}',(50,60),cv2.FONT_HERSHEY_SIMPLEX,0.9,(255,255,255
),2)

while True:
    _,img=cap.read()
    if frame_counter == cap.get(cv2.CAP_PROP_FRAME_COUNT):
        frame_counter = 0 #Or whatever as long as it is the same as next line
        cap.set(cv2.cv.CV_CAP_PROP_POS_FRAMES, 0)
    gray=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
    blur=cv2.GaussianBlur(gray,(3,3),1)
    Thre=cv2.adaptiveThreshold(blur,255,cv2.ADAPTIVE_THRESH_GAUSSIAN_C,cv2.THR
ESH_BINARY_INV,25,16)
    blur=cv2.medianBlur(Thre,5)
    kernel=np.ones((3,3),np.uint8)
    dilate=cv2.dilate(blur,kernel,iterations=1)
    check(dilate)

    cv2.imshow("Image",img)
    cv2.waitKey(10)
```

Results and Discussion

The proposed parking space vacancy management system was tested to determine its ability to accurately extract vacancy information from captured images of the parking space. A test image was obtained using an 8MP aerially positioned camera. The image used is shown in Figure 1. The image was grayed, stretched, smoothed, and binarized by using the image processing techniques described earlier. Figure 2 shows blured,smoothed and dilated test image.

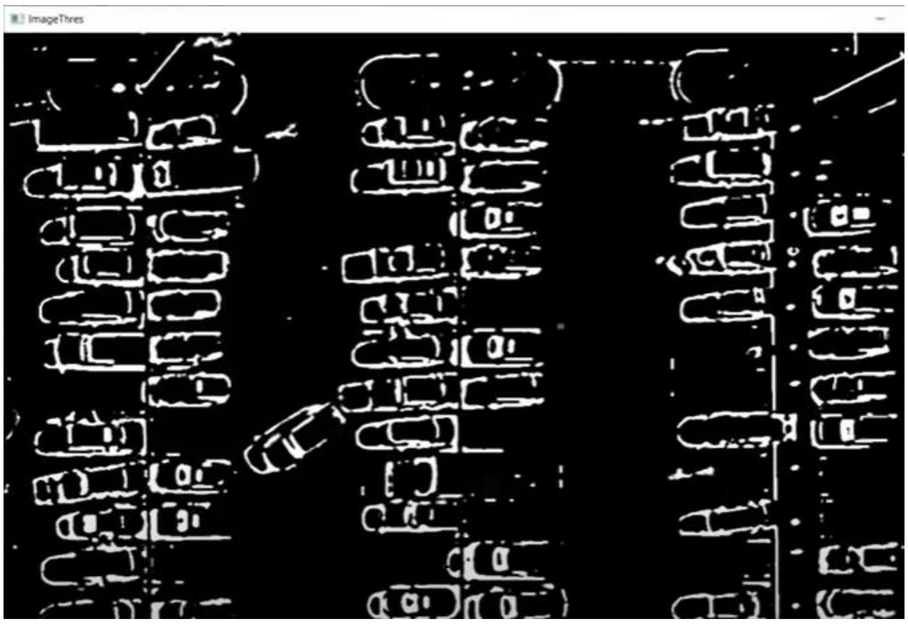


Fig 2 blured,smoothed and dilated test image

Using simple image processing techniques a sofisicated software car parking counter is developed successfully. For drivers, finding available parking spaces in parking lots can be a struggle. This paper describes a smart parking lot management system that uses image processing. An image processing method is used from aerial images of the parking area to discover empty spaces. The application analyses the image and gathers data on the locations and spot occupancy. The technology also tells you whether or not a certain parking place is occupied. By projecting occupancy information onto large displays positioned at key adjacent places, new drivers are given access to it. The cost of managing these spots is reduced, and parking cars takes less effort and time.

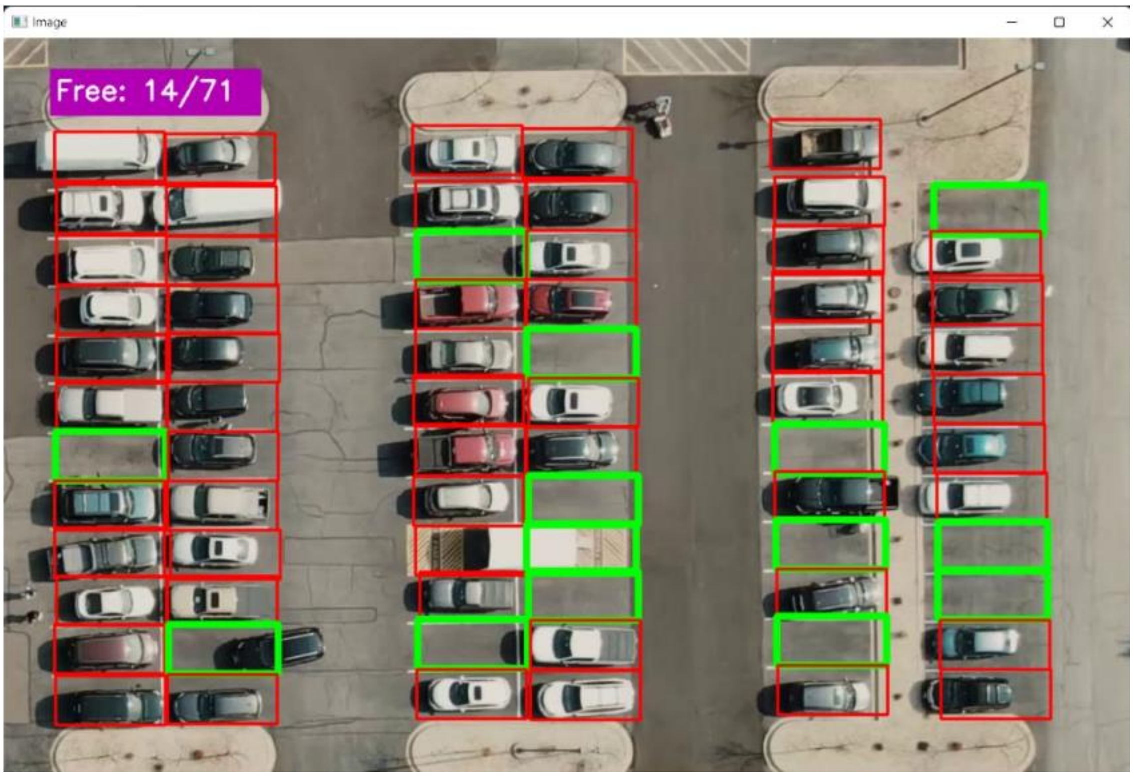


Fig3 image consisting of token count of the parking space.

Conclusion

The proposed system accurately detected the presence of cars in parking slots. The filled image approach performed better than the use of dilated edge images to determine vacancy of parking spots. The two approaches could be fused into one system. The camera position could be adjusted to improve the performance. From the above set of test results, it has been shown that the proposed imageprocessing-based system is a viable option for parking space vacancy management. Other technologies such as automatic number plate recognition and traffic light control could be conjoined with this system to form holistic intelligent transport systems.

Finding open parking spaces in parking lots is a common challenge for drivers. This study describes an image-processing-based smart parking lot management system. From aerial photos of the parking spot, an image processing technique is utilised to find unoccupied spaces. The programme analyses the image and extracts information on spot occupancy and their locations. The technology also provides information on whether or not specific parking spaces are occupied. New drivers are provided with occupancy information by projecting it onto sizable monitors placed at strategic locations nearby. The smart parking lot management system makes managing these spaces less expensive and lowers the stress and time waste involved with parking cars.

Through this program we are able to lessen the traffic concession faced at parking slots.

References (if any)

[1] A. S. Agbemenu, J. Yankey, and E. O. Addo. An automatic number plate recognition system using opencv and tesseract ocr engine. International Journal of Computer Applications, 180:1–5, 2018.

[2] T. Huang, Z. Zeng, and C. Li. Design and implementation of a prototype smart parking (spark) system using wireless sensor networks. In International Conference on Neural Information Processing,