# GCH3

FYP Final Report

# Real-time Vacancy Detection System Using Fisheye Cameras

by

HE Qihao, LIN Zhaorun, YIN Zhuohao, ZHOU Siyuan

**GCH3**

Advised by

Prof. Gary CHAN Shueng Han

Submitted in partial fulfillment of the requirements for COMP 4981

in the

Department of Computer Science

The Hong Kong University of Science and Technology

2022-2023

Date of submission: April 20, 2023

# Abstract

Smart parking systems that automatically detect vacant parking spaces have become increasingly popular. While magnetic sensors and signal lights are commonly used to detect vacancies and inform drivers of available spaces, they can be expensive due to high installation and maintenance costs. Vision-based solutions that use deep learning models to analyze images captured by cameras are a cost-effective alternative. However, existing solutions may not be suitable for indoor parking lots with limited height, which significantly limits the number of parking spaces a camera can detect.

This project aims to develop an indoor real-time vacancy detection system that achieves high accuracy and reduces costs compared to sensor-based solutions using fisheye cameras. The system was developed and tested in the HKUST parking lot using a fisheye camera to capture images and a deep-learning model that combined YOLOv5 and a customized Convolutional Neural Network (CNN) to detect vacancies. The model was trained only on an online open dataset and achieved an accuracy of 92.8% in testing on the HKUST parking lot. Besides, we develop a webpage to visualize detection results.

In conclusion, our system is accurate and highly robust enough to adapt to different parking lots without further training. Besides, as far as we investigate, we are the first group using fisheye cameras in indoor vacancy detection, which allows our system to detect more vacancies using fewer cameras.

# Table of Contents

# 1. Introduction

## 1.1 Overview

As vehicle possessions continue to grow globally, finding a vacant spot to park is becoming a challenging task, especially in busy car parks of large scale. Being unable to locate a vacant parking space has caused great frustration to drivers and immensely damaged the user experience. Hence, a smart system to detect vacancies precisely and quickly is in urgent demand. In our FYP, we seek to design and implement a real-time vacancy detection and visualization system using fisheye cameras.

## 1.2 Objectives

Our system is inspired by a common source of frustration of drivers under the circumstance where few vacant spots are left in a huge parking lot. With no additional information or guidance, drivers are left to drive around repeatedly, and finding a vacant spot depends entirely on chance. This is mainly due to the lack of an elaborate system, which is designed to provide accurate and fine-grained vacancy information. While some car parks are equipped with LED displays showing the number of vacant parking spaces of the whole car park, and others use signal lights onsite for each parking space to signify vacant as green and occupied as red, few car parks detect vacancy for each spot and gather information of all the spots to visualize for users. In other words, the information that can be of great assistance to ‘blind’ drivers in a parking complex is rarely provided, let alone in a straightforward manner. To remedy this unsatisfying user experience, we seek to identify the exact locations of vacant spots so that drivers are provided with more information and thus easier to find parking spaces.

To detect the real-time status of each parking space, we decided to use cameras to surveil them and apply object detection algorithms to identify the occupied ones. Amongst all types of cameras, one with a large field of view (FOV) is preferred since a more extensive coverage of each camera yields a smaller total number of cameras, hence lower deployment and maintenance costs. Fisheye cameras have a FOV of over 180 degrees and thus meet our demand perfectly. However, the severe distortion of fisheye images makes it challenging for traditional object detection algorithms to perform well. Consequently, we have proposed a pipeline that identifies vacancies by combining results from a custom Convolutional Neural Network model and the YOLOv5 model.

To fully develop our system, we focus mainly on the following objectives:

1. Capture parking lot images with fisheye cameras and use them as inputs to the detection system.
2. Train a CNN model to perform real-time classifications of vacancy status and combine its detection results with those of YOLOv5.
3. Implement a web page for visualization purposes, which will be updated every few seconds to show the vacancy conditions of each parking space.

# 1.3 Literature Survey

## 1.3.1 Sensor-based Vacancy Detection

Currently, there have been several successful smart parking lot examples for us to study, although most of them used sensors for occupancy detection tasks. One of the notable projects is the SFpark in San Francisco. In 2011, the San Francisco Municipal Transportation Agency (SFMTA) launched an initiative termed SFpark, which aimed to augment the management of on-street parking in San Francisco by utilizing demand-responsive pricing adjustments [1]. To gather information on on-street parking availability, roughly 8,000 parking spaces were outfitted with specialized sensors embedded in the pavement that transmitted data on availability at periodic intervals. French city Nice tried to replicate San Francisco's success. Approximately 15 million euros were spent on sensor deployment, yet the performance was far from satisfying [2]. Enríquez et al. also summarizes that sensor-based smart parking systems generally suffer from high maintenance costs [3]. A possible explanation is that sensors are prone to malfunctioning and that numerous sensors are required to cover a large region, hence a high maintenance cost.

## 1.3.2 Vision-based Vacancy Detection

In recent years, more occupancy detection systems powered by computer vision emerged. Tatulea et al. [4] developed a framework based on image-processing and feature-extraction techniques with an accuracy rate of over 95%. Acharya et al. [5]

approached the problem of occupancy detection by applying Convolutional Neural Networks (CNNs). They adopted the ImageNet-VGG-f model as the backbone, i.e., feature extractor and a binary Support Vector Machine (SVM) classifier head. The input images are taken from a surveillance camera above an outdoor parking lot. Each parking space is annotated on the input images and segmented before being fed into the deep neural net for classification. They eventually reached an accuracy of 96.6% on an independent test dataset. Cazamias and Marek [6] from Stanford University also adopted a similar workflow, where they used a CNN together with batch normalization techniques to produce classification results on manually segmented images of parking spaces.

However, like most existing projects that detect car park occupancy based on computer vision, it used images shot by standard cameras with no distortion but a narrower viewing angle. Our advantage lies in that fisheye cameras see 180 degrees, and with a proper shooting angle, they can even capture a dozen parking spaces by the two sides of a pavement. In this way, we have distorted images but also a much wider view so that fewer cameras are needed, hence lower deployment and maintenance costs. Additionally, as a majority of existing works studied outdoor parking lots, our work aims to fill in the gap of indoor parking buildings, where views are strictly confined due to much more complex terrains. In outdoor parking lots, cameras are easily mounted high and thus have a broad overhead view, which most existing vision-based methods use. On the contrary, in indoor scenarios, the horizontal view is more commonly used, yet no existing works implemented such methods.

# 2. Methodology

## 2.1 Design

### 2.1.1 System Architecture

The pipeline of the whole system is demonstrated in Fig. 1.

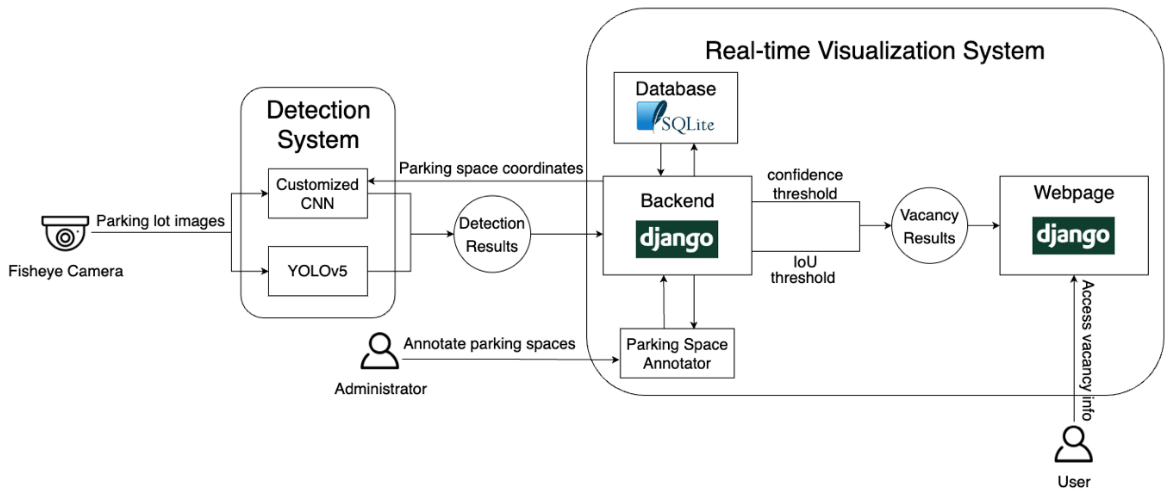



Figure 1: System architecture.

Our real-time vacancy detection system includes five major modules:

1. A fisheye camera to capture parking lot video stream.
2. A detection system able to extract images from the video stream and detect vacancies from captured images.
3. A backend system to send detection results to the frontend and store parking spaces coordinates.

4. A parking space annotator to enable administrators to draw a bounding box for each parking space.
5. A webpage to visualize the detection result.

### 2.1.2 Image Capturing and Transmission Design

To capture input data for detection purposes, we purchased a 5-megapixel TP-LINK fisheye camera (TL-IPC55AE), which is connected using 2.4 GHz Wi-Fi and supports the Real Time Streaming Protocol (RTSP). A picture is presented in Fig. 2. Data transmission is done conveniently when our detection system reads images of size 1726*1726 pixels directly from the RTSP stream of our camera using the OpenCV library.


Figure 2: A picture of our purchased fisheye camera.

### 2.1.3 Backend System Design

Our backend system consists of two sub-systems, namely the vacancy detection system, which contains a YOLOv5 detection module and a customized CNN detection module, and the real-time visualization system. The former retrieves raw images from the fisheye camera and produces detection results. They are then sent to the latter, and the real-time visualization system outputs the vacancy results for each parking space and sends them to the frontend for visualization. The real-time visualization system also stores information, such as the coordinates of the parking spaces, so that when combined with the detection results from the YOLOv5 model, the occupancy status can be determined.

### 2.1.4 Parking Space Annotator Design

For a given image, we need to know the coordinates of the target parking spaces for our model to predict whether each parking space is vacant. Therefore, a module named 'Parking Space Annotator' is demanded for efficient labeling, naming, and saving. Specifically, the administrator can draw bounding boxes for target parking spaces and name each parking space, such as 'A1'. Also, the module can save the coordinates of target spaces to our database.

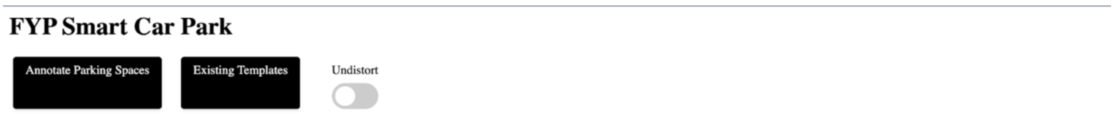



Figure 3: Access to the parking space annotator.

The administrator can use this module by clicking 'Annotate Parking Spaces' on the home page, as illustrated in Fig. 3. Afterwards, an interface will pop out for the administrator to annotate target parking spaces and name them. An example is presented in Fig. 4.

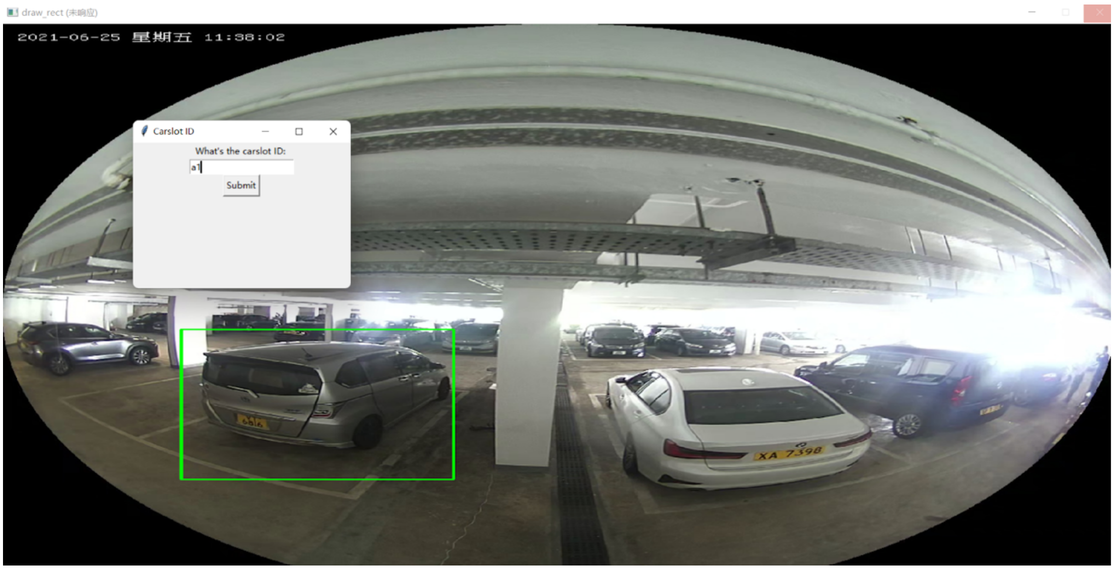



Figure 4: Parking Space Annotator interface.

## 2.1.5 Detection Module Design

### YOLO

As YOLO is one of the state-of-the-art object detection models, which is widely implemented in the industry due to its fast response and high accuracy, we decided to include it in our detection system [7]. The model architecture is shown in Fig. 5. In this project, YOLOv5 was adopted, and no further training was conducted since it has already been trained on the COCO dataset and has demonstrated robust performance for detecting vehicles in our preliminary experiments.

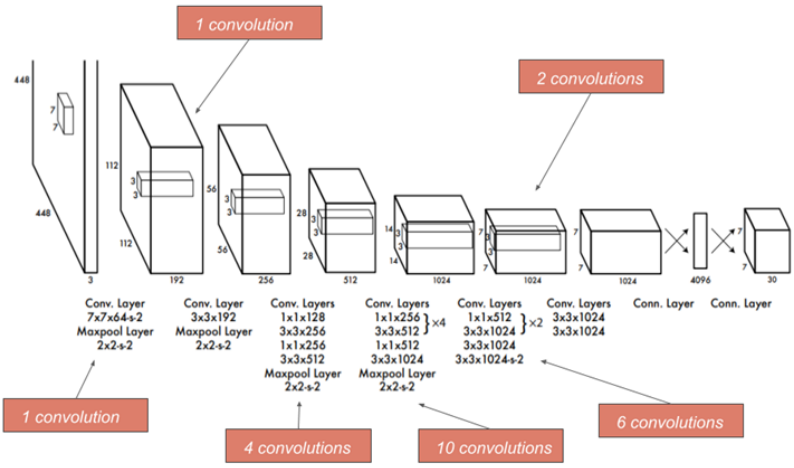
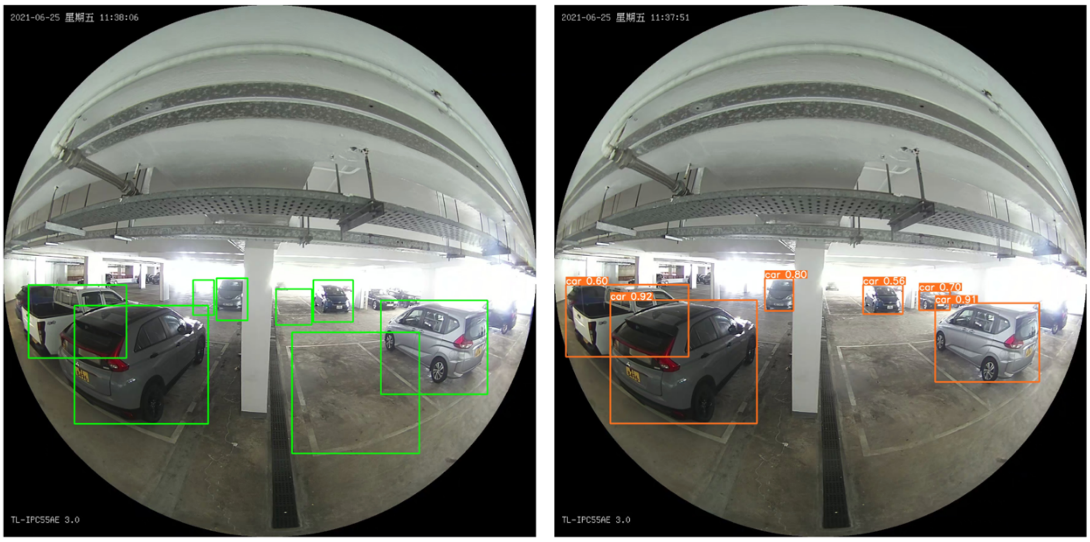



Figure 5: The structure of the YOLOv5 network.

However, YOLOv5 only detects vehicles and their locations within the frames, while the desired outputs are parking space occupancies, i.e., 0 or 1 for each parking space. In other words, we need to know which parking spaces are occupied by the detected cars. Fig. 6 (a) shows the parking spaces that the administrator annotated, while Fig. 6 (b) shows the output of the YOLOv5 model. To determine whether a car is parking in a specific parking space, we need a metric to evaluate the degree of overlap of each detected vehicle and each annotated parking space. To this end, we further incorporated an IoU (Intersection over Union) mechanism that evaluates the extent of overlap between the bounding box of the vehicle and that of the parking space. Eventually, 1 is assigned to the parking spaces with an IoU greater than 0.3 and 0 otherwise.




(a) (b)

Figure 6: (a) Annotated parking spaces. (b) YOLOv5 output.

**Customized CNN Design**

Currently, several works have used Convolutional Neural Network (CNN) to detect vacant parking spaces [6, 8]. However, their model only shows good performance in outdoor scenarios, which installs the camera in a high position and has an overhead view. Our system is designed for an indoor environment, so the camera position cannot be high enough to get an overhead view. Another issue is the generalization ability. Because we hope our model can be applied to other parking lots without training CNN again, this requires our model to be robust and has good generalization ability while the existing method shows the opposite [6].

To remedy the above problems, we used data-augmentation methods and Resnet-50 as our backbone to extract image features. Our customized CNN structure is shown in Fig. 7.

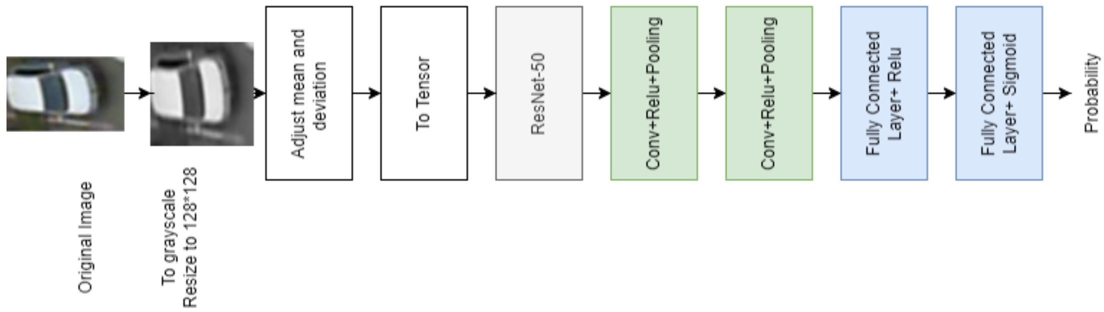



Figure 7: The structure of our customized CNN.

Because the lighting condition in the image of the online dataset is different from that of HKUST, which may decrease the accuracy, and it has been shown that pre-training on grayscale ImageNet improves medical image classification [9]. Therefore, we will first transform training images to grayscale and resize them to 128*128 pixels. Then adjust every image's standard deviation and mean to be the same to simulate the same lighting conditions. Besides, we utilized a pre-trained backbone, 'ResNet-50', to extract image features.

The input is a cropped image containing one parking space. Our customized CNN outputs a probability describing the likelihood that the parking space is occupied. In practice, we set a threshold equal to 0.8, meaning if the probability is larger than 0.8, we will regard the parking space as occupied, otherwise vacant.

The testing result shows that our customized CNN performs better than another CNN structure trained on the same dataset [5].

### 2.1.6 Web Page Design

With a straightforward web page design, users can get the occupancy information in the parking lot where 'red' indicates 'occupied' and 'green' otherwise. Fig. 8 below illustrates our frontend design.

In the HKUST parking lot, parking spaces are not pre-labeled with bay IDs, so we manually labeled them. In real-world applications, the administrator needs to draw bounding boxes for each parking space and define them according to the label system of the parking lot.

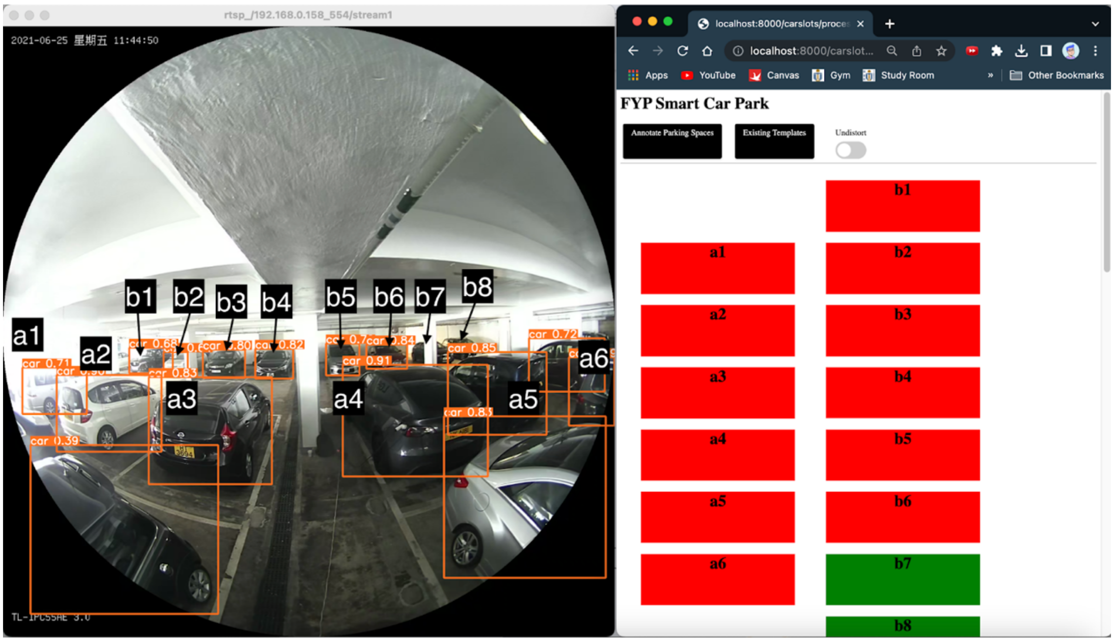



Figure 8: An illustration of the web page design.

There are also several functions available for the administrator to:

1. Set up the coordinates of parking spaces using 'Annotate Parking Spaces.'

2. View existing parking space setups using ‘Existing Templates.’
3. Activate calibration mode for overhead camera setup using the ‘Undistort’ switch.

## 2.2 Implementation

In this Section, we will first introduce the camera setup, including location, posture, and calibration method we have applied to our fisheye camera. Secondly, we will introduce CNN training details and some technical tricks we applied. Thirdly, we will show how to deploy the whole detection system, including data flow, IoU algorithm for YOLOv5 detection, and a combination of CNN and YOLOv5. Lastly, we will discuss details about frontend development.

### 2.2.1 Camera Set Up

**Location**

The system aims to help people find parking spaces in a large parking lot, especially when few vacant parking spaces are left. Our system is prototyped on the HKUST indoor parking lot. In particular, we deployed our camera on LG5 in the HKUST parking lot, as shown in Fig. 9.


Figure 9: Our designated experiment venue (LG5 of the HKUST parking lot).

Our preliminary experiments found that the YOLO-based detection model outperforms the CNN-based model in scenarios where the camera is mounted on the wall and shoots horizontally, whereas the CNN-based model performs better when the camera is mounted on the ceiling and points downwards. Based on these preliminary results, we have developed two setups, i.e., the horizontal setup and the overhead setup.

**Horizontal setup**

In this setup, the fisheye camera points towards 2 rows of parking spaces and is able to capture about 4 to 5 parking spaces per row. Our experiments discovered that the distortion in the horizontal setup did not significantly hinder the detection performance. Thus, no image undistortion is performed in this setup. An example is shown in Fig. 10.


Figure 10: An example view of the horizontal setup.

**Overhead setup**

In this setup, the fisheye camera is mounted on the ceiling pointing downwards. It is installed in the middle of driveways and captures 4 parking spaces on each side of the driveway. Since the distortion in this setup is extremely serious, image undistortion is applied for our detection models to perform robustly. An undistorted example is shown in Fig. 11.

Figure 11: An undistorted example of the overhead setup.

### 2.2.2 Image Undistortion

Due to the property of fisheye cameras, polar coordinates are adopted to represent a point in the original image, as shown in Fig. 12.

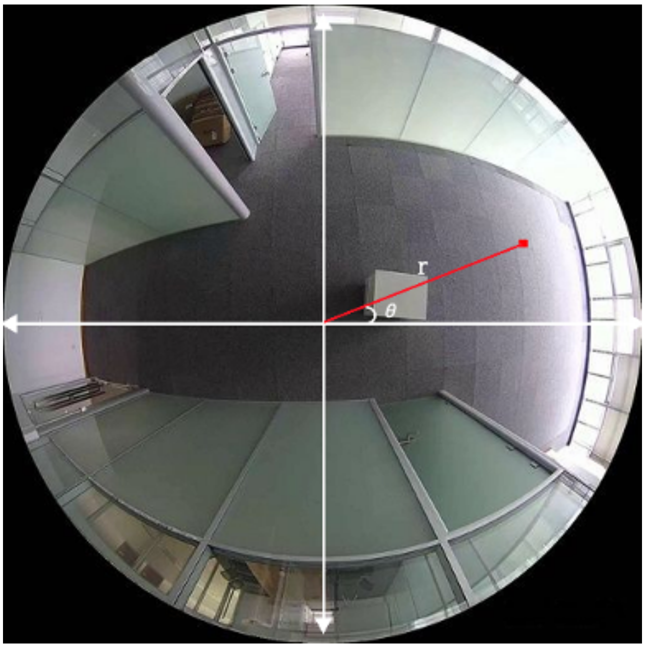



Figure 12: A raw fisheye image with polar coordinate system annotations.

Denote the new pixel coordinate as $(x, y)$. The mapping can be expressed in Eq. (1) and (2)

$$x = R + r\sin(\theta) \quad \text{Eq. (1)}$$

$$y = R + r\cos(\theta) \quad \text{Eq. (2)}$$

, where R is the radius of the effective region of the fisheye image.

Traversing all pixel points in the image yields two remap matrices, where each pair of x and y represents the new location of a pixel in the original image. The process is approximately a transformation from polar coordinates to Cartesian coordinates. The remapped image is shown in Fig. 13.

Figure 13: An undistorted image using the horizontal unfolding method.

### 2.2.3 Parking Space Annotator Implementation and Image Segmentation

The parking space annotator is written mainly using a python library called Tkinter and opencv-python. Tkinter is a Tk GUI toolkit, and opencv-python is a library for most kinds of image processing. To make sure different users draw rectangles fairly and consistently, we made the following rules:

1. The bounding box should wrap most areas of the target parking space.

2. The bounding box should wrap adjacent parking spaces as little as possible.

In practice, every time we install the camera, we take a picture and annotate target parking spaces on it. Then each space's number and coordinates are saved in our database. For each camera installation, we only need to annotate once because we assume the camera's position will not change, as do the coordinates of target parking spaces. An example of the bounding boxes of the parking spaces is shown in Fig. 14. Note that this example was captured using the overhead camera setup and was undistorted.

Figure 14: An example of standard bounding boxes.

## 2.2.4 Customized CNN Development

### Training Dataset

To train our CNN, we used the PKLot dataset[1]. Details of the PKLot dataset can be found in Table 1. It provides 12,000 images taken from three camera feeds of two parking lots. For each image in the dataset, we decide to transfer each image to a grayscale image and generate three different images by modifying its contrast ratio in order to enhance the generalization ability of the model.

| Parking lot | Weather condition | # of days | # of images | # of parking spaces occupied | | # of parking spaces empty | | # of parking spaces total |
|---|---|---|---|---|---|---|---|---|
| UFPR04 | Sunny | 20 | 2,098 | 32,166 | (54.98%) | 26,334 | (45.02%) | 58,400 |
| (28 parking spaces) | Overcast | 15 | 1,408 | 11,608 | (29.47%) | 27,779 | (70.53%) | 39,387 |
| | Rainy | 14 | 2,85 | 2,351 | (29.54%) | 5,607 | (70.46%) | 7,958 |
| | Subtotal | | 3,791 | 46,125 | (43.58%) | 59,720 | (56.42%) | 105,845 |
| UFPR05 | Sunny | 25 | 2,500 | 57,584 | (57.65%) | 42,306 | (42.35%) | 99,890 |
| (45 parking spaces) | Overcast | 19 | 1,426 | 33,764 | (59.27%) | 23,202 | (40.73%) | 56,966 |
| | Rainy | 8 | 2,26 | 6,078 | (68.07%) | 2,851 | (31.93%) | 8,929 |
| | Subtotal | | 4,152 | 97,426 | (58.77%) | 68,359 | (41.23%) | 165,785 |
| PUCPR | Sunny | 24 | 2,315 | 96,762 | (46.42%) | 111,672 | (53.58%) | 208,433 |
| (100 parking spaces) | Overcast | 11 | 1,328 | 42,363 | (31.90%) | 90,417 | (68.10%) | 132,780 |
| | Rainy | 8 | 831 | 55,104 | (66.35%) | 27,951 | (33.65%) | 83,056 |
| | Subtotal | | 4,474 | 194,229 | (45.78%) | 230,040 | (54.22%) | 424,269 |
| TOTAL | | | 12,417 | 337,780 | (48.54%) | 358,119 | (51.46%) | 695,899 |

Table 1: Summary of the PKLot characteristics.

[1] "PKLot - A Robust Dataset for Parking Lot Classification" (https://web.inf.ufpr.br/vri/databases/parking-lot-database/)

### Training Details

In practice, we used the Pytorch library to train our CNN. We set batch size = 16, learning rate = le-4, iteration number = 3750, and balance cross-entropy loss. Another technical trick is random rotating. Since original training images are all in vertical direction, as opposed to the random directions in reality, we applied random rotation to training images to make our model more robust apart from preprocessing mentioned in Section 2.1.5. Notice that random rotation is only applied in the training stage. In testing we do not use random rotation.

### Stabilize CNN Results

In order to make the predictions yielded by CNN more robust against noise, for example, passers-by occupying the parking spaces, we deployed a moving average mechanism that smooths out the effect of non-vehicle objects present in parking spaces. Since CNN is not able to determine the difference between car and non-vehicle objects accurately like YOLO, we do not want such objects' presence in parking spaces for a short duration in time being detected by CNN and marked as vacant in the frontend (e.g., a pedestrian or driver walks past some parking space). The algorithm for this mechanism is described in Fig. 15 below.

```python
# Write results
post_cnn = {}
for key in CNN_results:
    tmp = CNN_results[key].numpy()[0][0]
    if tmp > 0.7:
        post_cnn[key] = 1
        if trick_dictionary[key][trick_counter] == 0:
            trick_sum[key] += 1
        trick_dictionary[key][trick_counter] = 1  # This line may be unnecessary
    else:
        post_cnn[key] = 0
        if trick_dictionary[key][trick_counter] == 1:
            trick_sum[key] -= 1
        trick_dictionary[key][trick_counter] = 0  # This line may be unnecessary

trick_counter = (trick_counter + 1) % trick_counter_max  # Update trick_counter

for key in CNN_results:
    tmp = trick_sum[key]/trick_counter_max
    if tmp > 0.7:
        post_cnn[key] = 1
    else:
        post_cnn[key] = 0
```

Figure 15: The algorithm used by CNN to smooth out noise.

The idea is simply to store the prediction results of the past trick_counter_max, which is set to 100 frames of captured image stream in a tick_dictionary dictionary, with the keys being the frame numbers, and the value being the prediction results. The dictionary is updated in a First-In-First-Out manner through time. We average the sum of all values in this dictionary to obtain the averaged predictions and compare them with a threshold as would normally be done in the original CNN prediction pipeline.

The key point here is the threshold value reflects how robust the new CNN prediction pipeline is against noise, e.g., if the threshold is set to 0.5, then as long as in the past trick_counter_max frames, there are no more than 50% of frames containing noise that would affect CNN's prediction, the final results presented to the frontend would not be changed by this “50% factor”. On the contrary, it is also true that if a real prediction result does not persist over threshold*trick_counter_max frames (variable notations used here to clarify that threshold and trick_counter_max could be set to any values) and stored in the dictionary, it will not be reflected in the frontend. However, our assumption is that any parking status should at least last for a duration of more than half a minute, and except for significantly large threshold*trick_counter_max, the algorithm would not capture such status, which is not the case in our system where everything is reflected in time with a moderate threshold*trick_counter_max value. Another observation of this algorithm is that it consistently executes in constant time in each frame, and the constant time is also small, where we only need to update one value in the trick_dictionary and update the trick_sum once for all parking spaces. As a result, the runtime efficiency of the system is not affected.

### 2.2.5 Detection System Deployment

As introduced in Section 2.1, our detection system first retrieves sequences of raw images from the RTSP stream. Two separate detection modules (YOLOv5 and CNN) then run simultaneously to produce detection results. Specifically, the YOLOv5 detection module outputs the bounding boxes of the detected objects, and CNN produces probabilities of each parking space being occupied. These detection results are then sent to the Real-time visualization system for further processing and visualization.

### 2.2.6 Real-time Visualization System Deployment

### IoU Algorithm for YOLOv5 Detection

Since the YOLOv5 model only generates detected object types and their coordinates, we cannot determine whether a parking space is occupied directly. To solve this problem, we use the Intersection Over Union (IoU) algorithm to map them to occupancy status by calculating the intersection and union areas between the detected objects and the parking spaces. If the resulting IoU is larger than 0.3, we regard the parking space as occupied. An illustration of IoU is shown in Fig. 16.

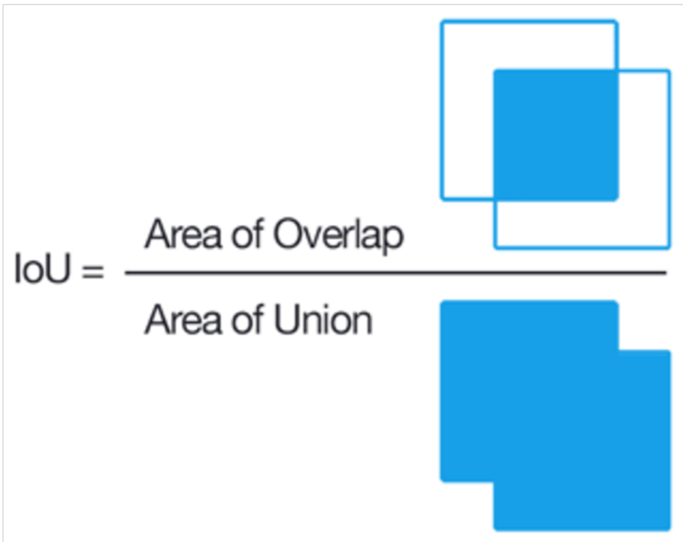



Figure 16: An illustration of the IoU algorithm.

### OR Operation

Since YOLOv5 first identifies vehicles before the bounding boxes of the detected vehicles are compared with those of the underlying parking spaces, it is extremely precise in that it cannot produce any false positive results. Based on this fact and our preliminary experiments, we combined the results from YOLOv5 and the customized

CNN model using an OR operation. The combined system demonstrated robust performance, and details of the testing results will be introduced in Section 2.3.2.3.

**A Dynamic Frontend**

We utilized the framework of Django to develop the whole system. It coordinates with our vacancy detection system by enabling the development wholly on the python environment. Furthermore, it also provides an integrated database with SQLite as the default so that we can easily store information about the parking lot, such as the coordinates of parking spaces.

JavaScript, CSS, and HTML were used to render the vacancy information onto the webpage. We used JavaScript to dynamically render the frontend so that it coordinates with the information stored in the SQLite database.

Ajax is used to send GET requests to the backend of the real-time detection system to fetch the data every 0.5 seconds so that the frontend is displayed in real-time.

## 2.3 Testing

The system modules have been tested separately, i.e., the Distortion Elimination (DE) functionality on the fisheye camera, three Occupancy Detection Models (ODM), the backend system for data transmission, and the user interfaces.

### 2.3.1 Test Distortion Elimination

To test the DE method, we have seen its effect on our machine learning model(s) (MLM). Based on our onsite testing results, both models perform well under the horizontal camera setup, where most of the parking spaces are further away from the camera, and therefore the distortion effect is not having a significant influence on the detection results. While for the overhead camera setting, the two models perform better if distortion elimination is applied to the images since the distortion effect is too significant as the parking spaces are closer to the camera.

## 2.3.2 Test Occupancy Detection Models

The Occupancy Detection Models deployed in the system that will be tested are a pre-trained YOLOv5 model, a custom CNN model, and a combination of the two based on logical OR operation (referred to as the YOLO+CNN model below). In this subsection, we will first introduce the test dataset we have collected and used, then the metrics we used to test the model performances, and finally how all the three models performed that are visualized through plots.

### 2.3.2.1 Test Dataset

The test dataset we are using comprises images we have taken in the HKUST LG5 indoor parking lot during periods in the noon/afternoon.

There are four positions from which we collect images, namely, two horizontal setups for cameras on the walls (called h1, h2) and two overhead setups for cameras on the ceiling (called o1, o2). The numbers of images we collected for h1, h2, o1, o2 are 6, 7, 6, and 6, respectively, and the numbers of parking spaces that are captured in h1, h2, o1, o2 are 48, 84, 53 and 50 respectively. In summary, we have 132 parking spaces for horizontal perspective, and 103 parking spaces for overhead perspective. Altogether there are in total 235 parking spaces contained in 25 images as data points in our test dataset. Table 2 below gives a holistic view of our test dataset.

| Camera Setup | Number of Images Captured | Number of parking spaces Captured |
|---|---|---|
| Horizontal setup 1 | 6 | 48 |
| Horizontal setup 2 | 7 | 84 |
| Overhead setup 1 | 6 | 53 |
| Overhead setup 2 | 6 | 50 |
| Total | 25 | 235 |

Table 2: Summary of our test dataset.

### 2.3.2.2 Test Metrics

We are concerned about the metrics commonly used to test classification models, which are namely accuracy, recall, precision and f1-score, all of which are associated with True Positive (TP), False Positive (FP), True Negative (TN) and False Negative (FN), and these relations are listed below in equations:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad \text{Eq. (3)}$$

$$Recall = \frac{TP}{TP+FN} \quad \text{Eq. (4)}$$

$$Precision = \frac{TP}{TP+FP} \quad \text{Eq. (5)}$$

$$F1\ score = \frac{TP}{TP+\frac{1}{2}(FP+FN)} \quad \text{Eq. (6)}$$

### 2.3.2.3 Test Results

The test results of all three models will be displayed in a similar manner in order to reveal their performances against all four metrics. Below we first present the models' performances separately in the form of confusion matrices, and then their results are compared side-by-side in terms of the metrics in bar charts.

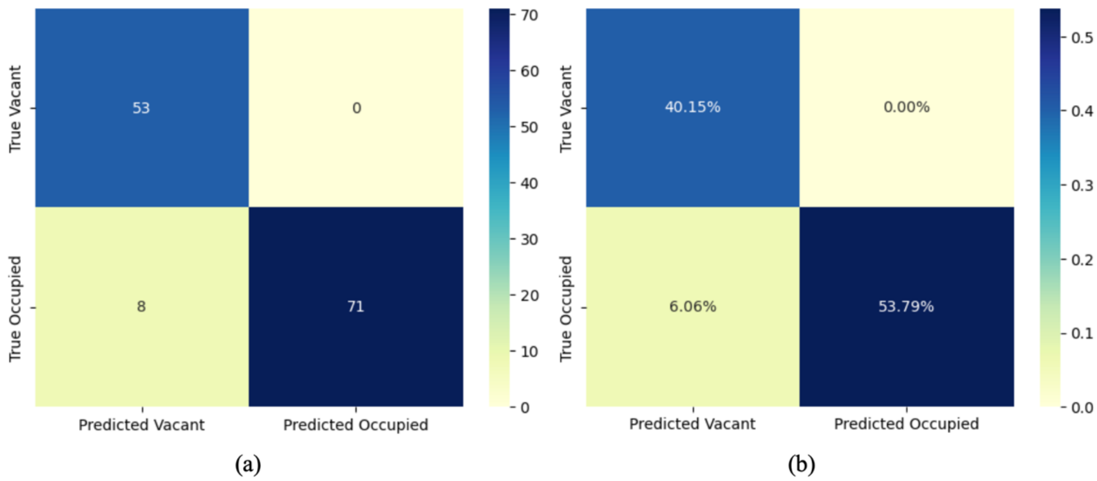
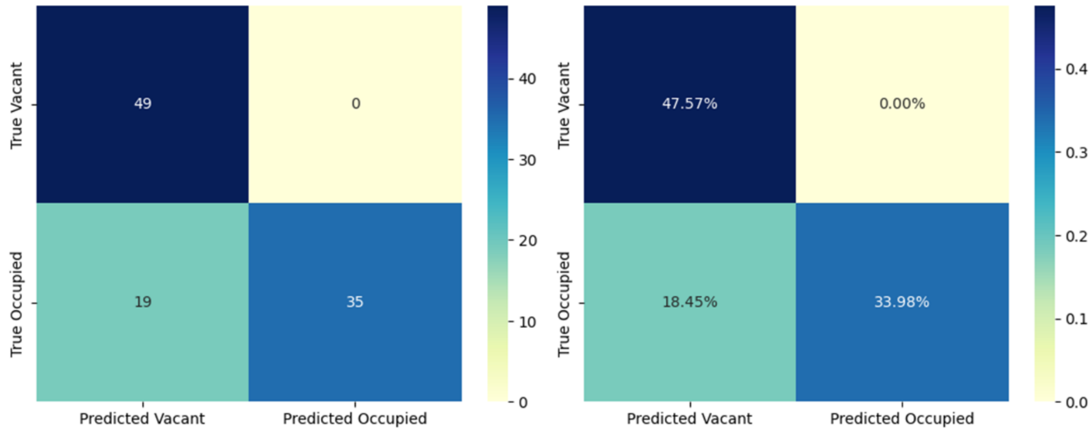


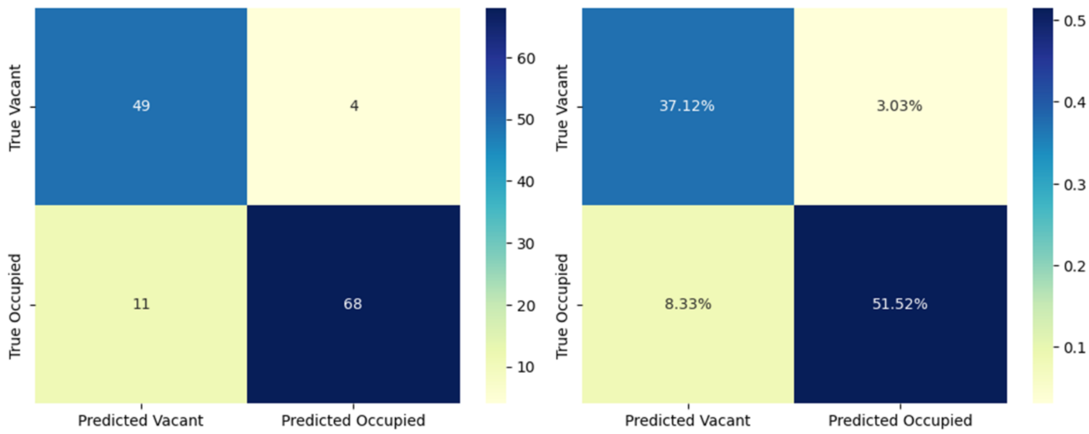

Figure 17: The confusion matrices of the YOLOv5 test results in the horizontal setup.

In Fig. 17, we have the confusion matrix plot for YOLOv5, which is assessed with the data points that belong to horizontal setting camera captured images. The upper left cell stands for TN; the upper right cell stands for FP; the bottom left cell stands for FN; the bottom right cell stands for TN. Fig. 17 (a) shows the absolute number of data points contained in each cell while Fig. 17 (b) shows the percentage. The same logic of interpreting confusion matrices applies to Fig. 18~23 below.




(a) (b)

Figure 18: The confusion matrices of the YOLOv5 test results in the overhead setup.




(a) (b)

Figure 19: The confusion matrices of the CNN test results in the horizontal setup.

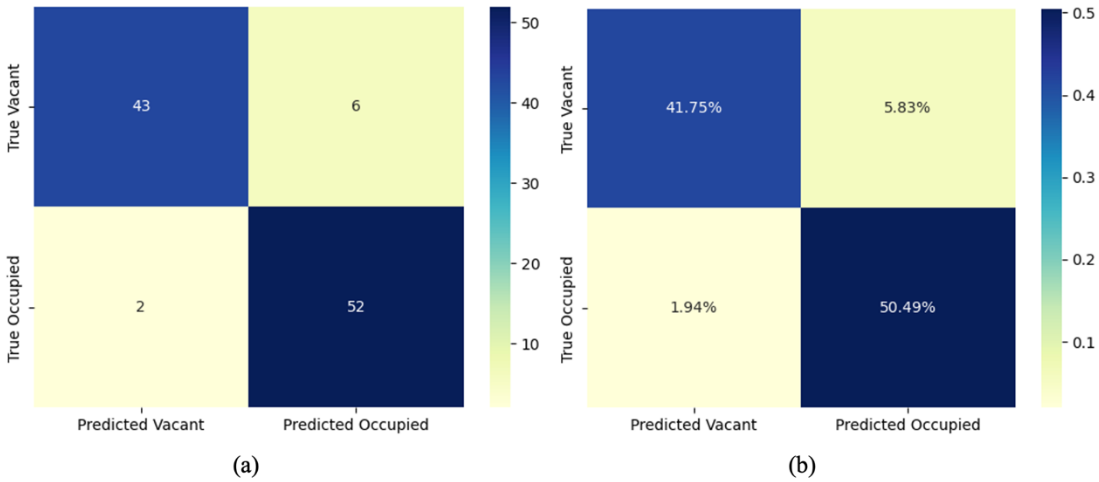



Figure 20: The confusion matrices of the CNN test results in the overhead setup.

Fig. 21~23 illustrate the testing results from three models on the entire test dataset.

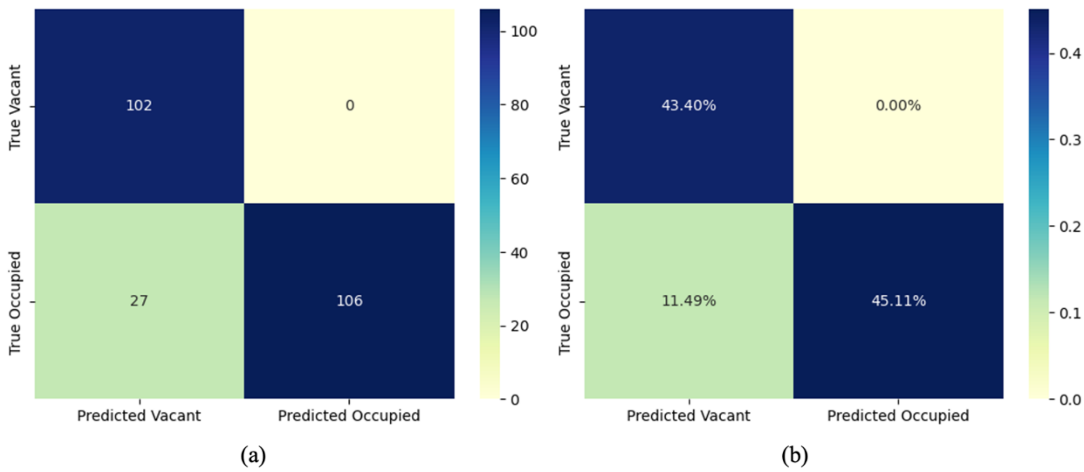



Figure 21: The confusion matrices of the YOLOv5 test results on the entire test dataset.

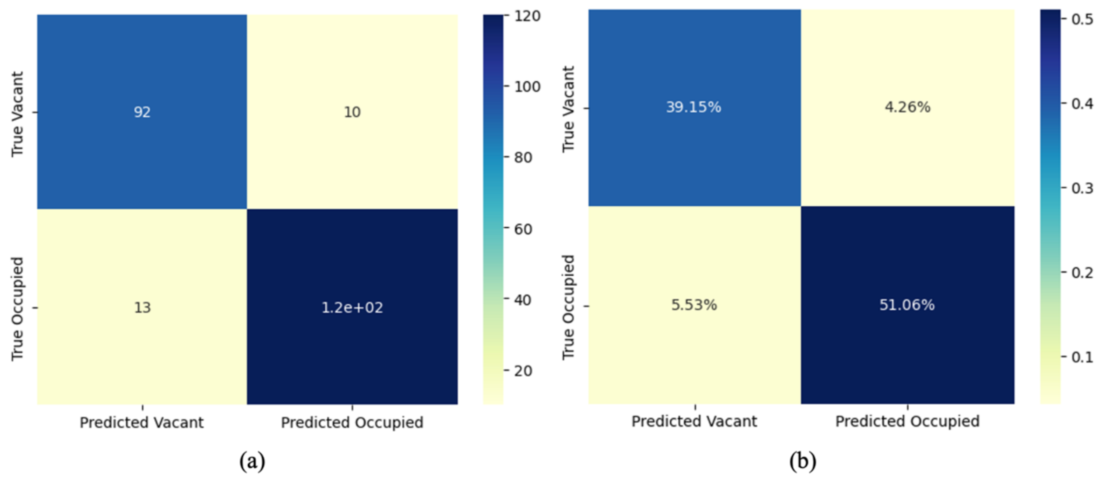



Figure 22: The confusion matrices of the CNN test results on the entire test dataset.

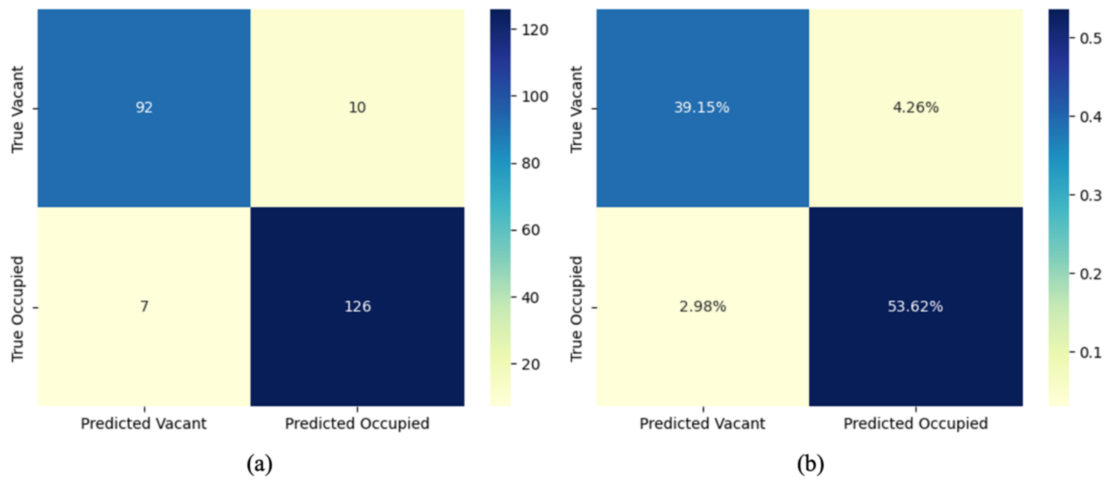



Figure 23: The confusion matrices of the YOLO+CNN test results on the entire test dataset.

In summary, CNN outperforms YOLOv5 in overhead camera settings, YOLOv5 outperforms CNN in horizontal camera settings, and YOLO+CNN outperforms both models when the entire test dataset is included. The three models' performances on all four test metrics are illustrated below in Figure 24, while one noticeable feature of the YOLOv5 performance is that it has a precision of 100% due to the fact that it never misclassified vacant parking space as occupied, since the model will never be affected by non-car object occupying the parking spaces, while the CNN might be affected under such kind of situations, e.g., a pedestrian walking by a vacant parking space.

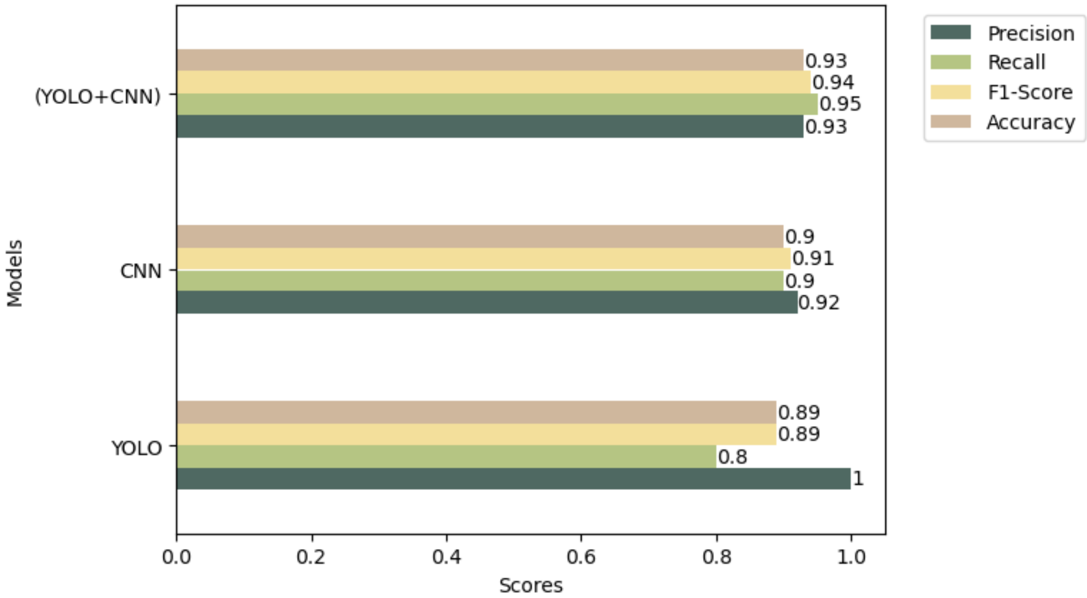



Figure 24: YOLO, CNN, YOLO+CNN's precision, recall, F1-score, and accuracy.

### 2.3.3 Test Real-Time Feedback System

To test the entire Real-Time Feedback System's (RTFS) performance, there are two components that require individual unit tests, which are the Data Transmission module and Computing module. We have done integration testing when combining the modules into the integral RTFS, which are fed into the integration process with the ODM, presumably with DE, which are from Section 2.3.2 and 2.3.1 respectively. Now the system has gone through more than 20 times of onsite trial runnings and the overall efficiency of it has been proven. In the following part the detailed testing of the said two components critical to the System response time will be discussed.

The Data Transmission process has been tested when onsite trial runnings were conducted. The overall latency for the captured image to be delivered to the detection models is small, with an average of approximately 10 seconds, an upper bound of 20 seconds and a lower bound of 5 seconds. So it is reliable enough to support updates in real-time with a frequency no smaller than one update per 30 seconds, which satisfies the objective of our system aiming at delivering parking slots vacancy distribution in real time.

The speed of computing, which mainly depends on the inference speed of the detection models, is reasonably fast, both models compute the results for up to 13 parking slots with almost no latency, i.e., the moment image frames are delivered to the models, results are obtained immediately. So this stage is also not a performance bottleneck for the system.

As a result of both fast data transmission and model computing speed, RTFS has been proven to provide reliable and fast results to users in real time.

# 2.4 Evaluation

## 2.4.1 Using Fisheye Cameras for Vacancy Detection

In Section 1.2, our first objective was to incorporate a fisheye camera used to collect input video streams for detection purposes. Overall, we have accomplished this objective despite several technical bottlenecks. In the preliminary stage of our project, we purchased a fisheye camera, as introduced in Section 2.1.2. The first bottleneck we encountered was that there weren't sufficient sockets in the HKUST indoor parking lot, which is required for the power supply of the camera. After we managed to find an appropriate spot for our camera installation, the next bottleneck was that our camera was unable to connect to the eduroam Wi-Fi possibly due to its special authentication process. To remedy this, we bought a router to provide Local Area Network (LAN) for our camera to connect. With the hardware configurations in place, we were able to access the video stream via the RTSP transmission scheme and use them for downstream tasks.

However, our work does have some limits. First, if we are to make our system available on the Internet, more work is needed regarding network configuration and hardware deployment for the system to provide reliable services. Moreover, we have only used one camera in our project as a prototype, while in the real-world situation, the intended outcome is to cover parking lots with multiple fisheye cameras. In this way, it remains to be solved how many cameras are needed for a given parking lot, as well as where and how to install them to achieve the minimum number of cameras. Intuitively, this is a nontrivial problem that presumably can only be determined manually. However, future work can be done to propose a deterministic solution to the deployment scheme of the cameras.

### 2.4.2 Building Detection Models

The second objective we have set is to build a detection model capable of giving fine-grained vacancy information. We have also fulfilled this objective by implementing 2 detection modules. The first detection module is developed based on the popular object detection model YOLO, which utilizes YOLO to detect vehicles and compares their bounding box coordinates with those of the parking spaces to identify vacancies. The accuracy of the YOLO-based detection module reached 93.9% in the horizontal camera setup. The second detection module is based on a customized CNN model, which works on cropped parking space images to produce a binary classification result. It achieved an accuracy of 92.2% in the overhead camera setup. Furthermore, the combined detection modules reached an accuracy of 93% on the whole test dataset. Apart from the test accuracies, in real-time system testing, our combined detection system also works smoothly and stably since we incorporated the moving average trick to stabilize the update of vacancies, as introduced in Section 2.2.4. Thus, we have accomplished this objective based on the accuracy results and real-time system performance.

### 2.4.3 Building Frontend Visualizations

The third objective we proposed was to build a web page that visualizes fine-grained vacancy information and updates in real time. We have accomplished it by developing a visualization system using the Django framework. On the web page, we implemented the basic functionalities, including visualizing the parking lot map and coloring parking spaces green or red according to the vacancy status. Additionally, we implemented a Parking Space Annotator (PSA) for the administrators to manually annotate the coordinates of parking spaces. However, more functionalities can be added, such as storing the vacancy histories and generating parking logs for the administrators to analyze the utilization of different areas of the parking lot and propose solutions to possible uneven parking issues.

# 3. Discussion

## 3.1 Camera setup

During the implementation of the vacancy detection system, we discovered that the YOLOv5 model performed the best in horizontal setup, and the customized CNN model generated the best results in overhead setup. However, no universal solution satisfies both, so combining the two models creates a more general solution and provides more flexibility to our system.

## 3.2 Real-time Visualization System

As the customized CNN model only detects whether the parking space is occupied, it may be affected by other factors, such as a car passing by. To cope with this issue, we implemented a technique to store the occupancy status of the past 30 seconds and maintain a moving average. In this way, we efficiently smooth out the factors that may pose a sudden effect on our system.

# 4. Conclusion

## 4.1 Summary of Achievements

In our project, we develop a real-time vacancy detection system for indoor parking lots. Our unique detection model design takes advantage of a customized CNN and YOLOv5, making our system support two camera installation ways, horizontally and overhead. Therefore, our system is easy to deploy in any parking lot to cover every parking space. Besides, trained with several data augmentation methods, our detection model shows high robustness – it is trained by an online dataset but shows high accuracy on testing data from HKUST parking lot. Thus, our system can be directly deployed in other parking lots without further training. We also developed a user-friendly webpage to visualize the detection outcome, enabling users to learn parking lot conditions clearly before arrival.

## 4.2 Future Work

Although our system seems complete and shows good performance, there are several aspects we consider extending in the future:

1. **Enable the parking space annotator to support polynomial bounding boxes**: Currently, our parking space annotator only supports rectangle

bounding boxes. Although it can satisfy most needs, in some cases, rectangle bounding boxes will inevitably wrap adjacent parking spaces' area, leading to occlusion and false positive detection results. This issue can be eliminated largely, if we can draw polynomial bounding boxes.

2. **A system able to finetune models according to different parking lots**: Although our system achieves 92.77% accuracy on the HKUST parking lot without further training, the performance will be better with only a few more local training data. However, updating our model using more local data will be complex. Because YOLOv5 requires users to label every vehicle with coordinates and our customized CNN only requires cropped parking spaces. The discrepancy between YOLO and CNN data format makes further training complex. In the future, we can design a pipeline based on the current system that collects local parking lot images and ground truth according to the administrator's feedback. This can be realized by allowing users to input ground truth when using our parking space annotator module, and our backend will save the image and ground truth in the required format. In this way, we can improve our detection model.
3. **Edge AI**: If we can deploy our machine learning model to the camera, it will largely reduce latency, bandwidth requirements, and costs. Currently, we use a router to enable communication between the camera and our backend, which is clumsy and complex.

# 5. References

[1] D. Simons, ‘SFpark: San Francisco knows how to park it’, Sustainable Transport, no. 23, 2012.

[2] T. Lin, H. Rivano, and F. Le Mouël, ‘A survey of smart parking solutions’, IEEE Transactions on Intelligent Transportation Systems, vol. 18, no. 12, pp. 3229–3253, 2017.

[3] F. Enríquez, L. M. Soria, J. A. Álvarez-García, F. Velasco, and O. Déniz, ‘Existing approaches to smart parking: An overview’, in Smart Cities: Second International Conference, Smart-CT 2017, Málaga, Spain, June 14-16, 2017, Proceedings 2, 2017, pp. 63–74.

[4] P. Tătulea, F. Călin, R. Brad, L. Brâncovean, and M. Greavu, ‘An image feature-based method for parking lot occupancy’, Future Internet, vol. 11, no. 8, p. 169, 2019.

[5] D. Acharya, W. Yan, and K. Khoshelham, ‘Real-time image-based parking occupancy detection using deep learning’, Research@ Locate, vol. 4, pp. 33–40, 2018.

[6] J. Cazamias and M. Marek, ‘Parking space classification using convolutional neural networks’, Technical Report, 2016.

[7] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, ‘You only look once: Unified, real-time object detection’, in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 779–788.

[8] A. Farley, H. Ham, and Others, ‘Real time IP camera parking occupancy detection using deep learning’, Procedia Computer Science, vol. 179, pp. 606–614, 2021.

[9] Y. Xie and D. Richmond, ‘Pre-training on grayscale imagenet improves medical image classification’, in Proceedings of the European conference on computer vision (ECCV) workshops, 2018, pp. 0–0.

# 6. Appendix A: Meeting Minutes

## 6.1 Minutes of the 1st Project Meeting

Date: July 4, 2022

Time: 9:00 pm

Place: Zoom

Present: HE Qihao, LIN Zhaorun, YIN Zhuohao, ZHOU Siyuan

Absent: None

Recorder: YIN Zhuohao

1. Approval of minutes

   This was the first formal group meeting, so there were no minutes to approve.

2. Report on progress

   2.1 All team members have read the instructions of the Final Year Project online and have done research for the topic.

   2.2 All team members have done some research on existing smart parking systems.

3. Discussion items

3.1 Qihao and Siyuan found a survey paper which covers existing hardware and software technologies applied in smart parking lots and their performances. This got us thinking about the limitations and our niche.

3.2 Zhuohao proposed that a parking space reservation system can help drivers reserve their preferred spots. After reservation, indoor navigation service may also be provided to guide the driver to his/her chosen spot.

3.3 Zhaorun reminded us of the problem that drivers have to pay manually before they leave, which is far from a smooth and simplified user experience. Thus, we raised another direction that we can build an auto-payment system to facilitate automatic counting of hours and payment.

3.4 Siyuan found a great example app which features parking space selection and indoor navigation. However, the methodology seems a myth to us.

3.5 Zhuohao found two successful smart parking systems, LA Express Parking and SF Parking, but they are both outdoor and rely on a mass number of sensors and meters.

4. Goals for the coming week

4.1 All members will brainstorm more to find limitations of existing systems or areas that haven't been explored.

5. Meeting adjournment and next meeting

The meeting was adjourned at 11:00 pm.

The next meeting will be at 2:00 pm on July 20th on Zoom.

## 6.2 Minutes of the 2nd Project Meeting

Date: July 20, 2022

Time: 2:00 pm

Place: Zoom

Present: Dr. Kent CHEN, HE Qihao, LIN Zhaorun, YIN Zhuohao, ZHOU Siyuan

Absent: None

Recorder: YIN Zhuohao

1. Approval of minutes

   The minutes of the last meeting were approved without amendment.

2. Report on progress

   2.1 All team members have done more reviews on existing solutions.

   2.2 Considering the fact that in busy areas parking spaces are extremely hard to find, we proposed to build a reservation system.

3. Discussion items

   3.1 Zhuohao explained the mechanism of the reservation system: A user can reserve a spot for a specific time period at a specific parking lot, so that he/she won't have to drive around to find a vacant spot. There will be two queues, one for reserved users and the other for walk-in users.

3.2 Kent challenged us that most car parks have limited spaces for traffic, which makes it difficult to facilitate a 2-queue model. Possible traffic jams are likely to occur.

3.3 Kent also questioned our business model: How should the parking lots charge drivers so that they have an incentive to adopt the reservation system?

4. Goals for the coming week

4.1 All members will think of the potential challenges Kent mentioned as well as possible solutions.

5. Meeting adjournment and next meeting

The meeting was adjourned at 2:00 pm.

The next meeting will be at 2:00 pm on August 11st on Zoom.

# 6.3 Minutes of the 3rd Project Meeting

Date: Aug 11, 2022

Time: 2:00 pm

Place: Zoom

Present: Dr. Gary CHAN, HE Qihao, LIN Zhaorun, YIN Zhuohao, ZHOU Siyuan

Absent: None

Recorder: YIN Zhuohao

1. Approval of minutes

The minutes of the last meeting were approved without amendment.

2. Report on progress

2.1 All members have done research on the potential challenges the reservation system may face.

2.1.1 For traffic jam at the entrance, the team agreed that it is hard to address the problem since more factors are to be considered when reservation is made available.

2.1.2 For business model, after doing research, the team found that the drivers are not so motivated to adopt such a system since it increases the potential costs of booking a slot, making it unprofitable.

3. Discussion items

3.1 The team decided to discard the idea of building a reservation system, and instead, turned to car occupancy detection system, which is rooted from a fraction of the original idea and reinforce it.

3.2 The team decided to integrate the car occupancy detection model to a fisheye camera to increase the efficiency of car occupancy detection.

3.3 The following factors of the project are discussed:

3.3.1 Target parking lot size: parking lots in large shopping mall since they have higher demands on such occupancy detection systems.

3.3.2 Experiment place: HKUST indoor parking lot since it resembles large parking lots in shopping malls.

3.3.3 Advantages over traditional cameras: Wider angle of view and hence more slots can be captured at once. This leads to fewer cameras deployed, and less installation and maintenance costs.

3.3.4 Data transmission methods: Wireless transmission to pc

3.3.5 Training dataset:

1. Online 2D images

2. Distorted images collected from our own fisheye cameras, which needs to be undistorted and labeled.

3.3.6 Training methods: YOLO algorithm with online 2D images, YOLO algorithm with distorted images collected, CNN trained by our own with online 2D images, CNN trained by our own with distorted images collected.

4. Goals for the coming month

4.1 The team set work distributions for a few topics to be further studied on.

4.1.1 YIN Zhuohao will focus on YOLO algorithm.

4.1.2 ZHOU Siyuan will look for methods to create our own CNN.

4.1.3 HE Qihao will search for post YOLO methods for detecting the car slot position.

4.1.4 LIN Zhaorun will analyze the potential effects of different types of datasets on the precision of car occupancy detection model

5. Meeting adjournment and next meeting

The meeting was adjourned at 4:00 pm.

The next meeting will be at 21:00 on August 31st via Zoom.

## 6.4 Minutes of the 4th Project Meeting

Date: Aug 31, 2022

Time: 9:00 pm

Place: Zoom

Present: HE Qihao, LIN Zhaorun, YIN Zhuohao, ZHOU Siyuan

Absent: None

Recorder: HE Qihao

1. Approval of minutes

The minutes of the last meeting were approved without amendment.

2. Report on progress

2.1 All team members are working on the proposal report and examining all the ideas discussed throughout the summer term.

2.2 The team is now moving on to implement the crucial components of the system, namely the models.

3. Discussion items

3.1 Qihao and Siyuan are going to construct a custom CNN model.

3.2 Zhuohao and Zhaorun will construct a system over which YOLOv5 could run with inputs of both types of images and video stream.

4. Goals for the coming month

4.1 The team should have both models implemented and test them using data collected by earlier projects also using fisheye cameras.

5. Meeting adjournment and next meeting

The meeting was adjourned at 11:00 pm.

The next meeting will be at 2:00 pm on Oct 20th on Zoom.

## 6.5 Minutes of the 5th Project Meeting

Date: Oct 20, 2022

Time: 2:00 pm

Place: Zoom

Present: Dr. Gary CHAN, HE Qihao, LIN Zhaorun, YIN Zhuohao, ZHOU Siyuan

Absent: None

Recorder: HE Qihao

1. Approval of minutes

The minutes of the last meeting were approved without amendment.

2. Report on progress

2.1 The goals of the previous month were accomplished successfully, with both models running correctly on real data captured by fisheye cameras. Their performances in terms of accuracy are also promising.

3. Discussion items

3.1 Siyuan proposed a custom data augmentation pipeline that could enhance the models' performance and robustness.

3.2 Qihao proposed that a segmentation process is needed for both models. For the CNN model, each parking space (with 0~1 car) can be cropped out from the original image and sent to the model to make decisions about the occupancy status of each parking space individually. Such a process can be achieved either manually or automatically.

3.3 Zhaorun and Zhuohuo proposed that for YOLOv5, that segmentation can be achieved using bounding boxes (BB) labeled manually, and determine occupancy using IoU (Intersection over Union) of the BB labeled manually and BB output by the model.

4. Goals for the coming month

4.1 Members will try to develop the segmentation algorithm.

4.2 Zhaorun will purchase a fisheye camera of our own that will be used to collect data of our own from the HKUST parking lot. And that will enable us to conduct testing in real-time.

5. Meeting adjournment and next meeting

The meeting was adjourned at 3:00 pm.

The next meeting will be at 2:00 pm on Nov 17th on Zoom.

## 6.6 Minutes of the 6th Project Meeting

Date: Nov 17, 2022

Time: 2:00 pm

Place: Zoom

Present: Dr. Gary CHAN, HE Qihao, LIN Zhaorun, YIN Zhuohao, ZHOU Siyuan

Absent: None

Recorder: HE Qihao

1. Approval of minutes

The minutes of the last meeting were approved without amendment.

2. Report on progress

2.1 The team tested in the UST parking space using the newly obtained fisheye camera and found that a router is needed for smooth data transmission locally.

2.2 Zhaorun developed a demo website that showcases the basic functionalities that could be used by drivers and managers of the car park.

2.3. Siyuan developed a demo bounding box drawing program that the managers of the car park could use.

3. Discussion items

3.1 The team decided to test the entire system on the UST car park after the modules, including the BB drawing program and the website, are completed.

3.2 The professor gave the advice of using deep learning for image segmentation, and the team will consider adding it if the system could work accurately given manually segmented images using BBs drawn.

4. Goals for the coming month

4.1 The team will test the system in the UST parking lot, given both software and hardware modules are in place.

5. Meeting adjournment and next meeting

The meeting was adjourned at 3:00 pm.

The next meeting will be at 2:00 pm on December 22nd on Zoom.

# 6.7 Minutes of the 7th Project Meeting

Date: Dec 22, 2022

Time: 2:00 pm

Place: Zoom

Present: Dr. Gary CHAN, HE Qihao, LIN Zhaorun, YIN Zhuohao, ZHOU Siyuan

Absent: None

Recorder: HE Qihao

1. Approval of minutes

The minutes of the last meeting were approved without amendment.

2. Report on progress

2.1 Zhuohao Proposed putting everything together using the Django web framework so that different important modules could interoperate without much difficulty.

2.2 Siyuan finished the bounding box drawing program.

3. Discussion items

3.1 The team agreed that further improvements and adjustments will wait till the system is up and tested in the UST parking lot.

4. Goals for the coming month

4.1 Zhuohao will develop the Real Time Streaming Protocol module.

4.2 Siyuan will develop a website to demo his BB drawing program using Django.

4.3 Zhaorun will develop the entire Django project that is primarily based on YOLO.

4.4 Qihao will develop the procedure that combines BB information with the CNN model.

5. Meeting adjournment and next meeting

The meeting was adjourned at 3:00 pm.

The next meeting will be at 9:00 pm on January 28th on Zoom.

## 6.8 Minutes of the 8th Project Meeting

Date: Jan 28, 2022

Time: 9:00 pm

Place: Zoom

Present: HE Qihao, LIN Zhaorun, YIN Zhuohao, ZHOU Siyuan

Absent: None

Recorder: HE Qihao

1. Approval of minutes

The minutes of the last meeting were approved without amendment.

2. Report on progress

2.1 Zhaorun built the system that contains YOLOv5 functionality.

2.2 Siyuan demonstrated his bounding box drawing program under Django framework.

2.3 Qihao built the process that combines BB drawing with the CNN model.

2.4 Zhuohao finished the RTSP part of the system. And the system could now accept real-time video as the input for the models.

3. Discussion items

3.1 The team will test the system in the UST car park before the new semester begins.

4. Goals for the coming month

4.1 The team will consider how to improve the system given the new results obtained from the test mentioned in the discussion.

5. Meeting adjournment and next meeting

The meeting was adjourned at 11:00 pm.

The next meeting will be at 4:00 pm on Feb 2nd in the HKUST Car park.

## 6.9 Minutes of the 9th Project Meeting

Date: Feb 2, 2022

Time: 4:00 pm

Place: HKUST Car park LG5 (demo)

Present: Dr. Gary CHAN, HE Qihao, LIN Zhaorun, YIN Zhuohao, ZHOU Siyuan

Absent: None

Recorder: HE Qihao

1. Approval of minutes

The minutes of the last meeting were approved without amendment.

2. Report on progress

2.1 The team successfully tested the system equipped with YOLOv5 in a real-time situation and obtained the prediction results for 8 parking spaces.

3. Discussion items

3.1 Professor Gary suggested the team consider where to put the camera, so that more parking spaces may be covered.

3.2 The team found the prediction results are not very stable, so an improvement in the IoU algorithm may be needed.

4. Goals for the coming month

4.1 Qihao will put the CNN in the system.

4.2 The team will investigate if there is a way to improve the system's performance after the CNN module is integrated into the system.

5. Meeting adjournment and next meeting

The meeting was adjourned at 5:00 pm.

The next meeting will be at 21:00 pm on Feb 11th via Zoom.

## 6.10 Minutes of the 10th Project Meeting

Date: Feb 11, 2022

Time: 9:00 pm

Place: Zoom

Present: HE Qihao, LIN Zhaorun, YIN Zhuohao, ZHOU Siyuan

Absent: None

Recorder: HE Qihao

1. Approval of minutes

The minutes of the last meeting were approved without amendment.

2. Report on progress

2.1 Qihao integrated the CNN model into the system.

2.2 The CNN model appears to perform with robustness and accuracy, given both distorted and undistorted inputs. But there could still be some misclassifications.

3. Discussion items

3.1 Team members decided what the Progress report should focus on after a talk with the Communicator Tutor.

3.2 Siyuan proposed that the priority for the project now is how to increase the models' performances and stabilize the results obtained in real-time given the demo result obtained on Feb 2nd. The major focus points will be as follows:

3.2.1 New structure for the CNN model to eradicate misclassification under a stable environment (when the occupancy status does not vary much).

3.2.2 A better IoU algorithm for the YOLOv5 model so that it obtains more stable results on occupancy status.

3.2.3 Combination of both YOLOv5 and CNN to optimize the system output. Either in terms of accuracy or efficiency.

3.2.4 Improve the system UI.

4. Goals for the coming month

4.1 As listed in items of Section 3.2.

5. Meeting adjournment and next meeting

The meeting was adjourned at 11:00 pm.

The next meeting will be at 21:00 pm on Feb 16th via Zoom (tentatively).

## 6.11 Minutes of the 11th Project Meeting

Date: February 16, 2023

Time: 6:00 pm

Place: Library LG1

Present: HE Qihao, LIN Zhaorun, YIN Zhuohao, ZHOU Siyuan

Absent: None

Recorder: HE Qihao

1. Approval of minutes

The minutes of the last meeting were approved without amendment.

## 2. Report on progress

2.1 The YOLO team (LIN Zhaorun, YIN Zhuohao) adjusted the IoU algorithm by adopting a smaller threshold value and made the system refresh time longer to avoid flicking of occupancy detection results.

2.2 A new CNN model was proposed by ZHOU Siyuan with a resnet50 backbone.

2.3 A study of similar nature was brought up by HE Qihao, and their CNN model was examined, potentially to be used in test & evaluation against our own model in the HKUST parking lot.

## 3. Discussion items

Team members discussed what to do on the model-fusing algorithm. The conclusion was reached that a max voting may still be the most suitable option. But more test&evaluation is needed.

## 4. Goals after this meeting

The team will continue collecting testing data, and more camera installation settings will be tried out.

5. Meeting adjournment and next meeting

The meeting was adjourned at 7:00 pm.

The next meeting will be at 9:00 am on February 22nd via Zoom.

## 6.12 Minutes of the 12th Project Meeting

Date: February 22, 2023

Time: 9:00 am

Place: Zoom

Present: Dr. Kent Chen, HE Qihao, LIN Zhaorun, YIN Zhuohao, ZHOU Siyuan

Absent: None

Recorder: HE Qihao

1. Approval of minutes

The minutes of the last meeting were approved without amendment.

2. Report on progress

There was not much progress from the last meeting (February 16).

3. Discussion items

3.1. Dr. Kent advised the team to have a larger number of captured parking spaces. And the optimal camera setting in this case needs to be discovered.

3.2 Dr. Kent acknowledged the level of completeness of the system, and suggested the team should work on improving the stability and accuracy of the system by working on the details.

4. Goals after this meeting

The team will try out more camera installation settings, and at the same time try to discover possible improvements in the algorithms.

5. Meeting adjournment and next meeting

The meeting was adjourned at 10:00 am.

The next meeting will be at 2:30 pm on March 15th in the HKUST parking lot.

# 6.13 Minutes of the 13th Project Meeting

Date: March 15, 2023

Time: 2:30 pm

Place: HKUST parking lot LG5

Present: HE Qihao, LIN Zhaorun, YIN Zhuohao, ZHOU Siyuan

Absent: None

Recorder: HE Qihao

1. Approval of minutes

The minutes of the last meeting were approved without amendment.

2. Report on progress

2.1 Qihao tested the new CNN with a resnet50 backbone, which showed better performance on testing images compared with the old CNN model.

2.2 Zhaorun tested various possible settings of the camera beforehand and contributed several testing data points.

3. Discussion items

This meeting primarily serves the purpose of data collection and system testing.

3.1 The team found that the two models show different performance under different camera settings, as for the YOLO model, it performs better than the CNN model when the camera is attached on the wall, while the CNN model performs better when the camera is attached on the ceiling (as before).

3.2 As a result of 3.1, the team decided to fuse the two models in a way that the result will be obtained based on the better-performing model under the current camera setting. So that a weighted max-voting or even logical-or operation on the models' results are possible choices instead of a vanilla max-voting algorithm.

4. Goals after this meeting

4.1 The team will collect more data from both camera settings (wall & ceiling).

4.2 The team will be preparing for a presentation in the coming Friday.

5. Meeting adjournment and next meeting

The meeting was adjourned at 5:00 pm.

The next meeting will be at 6:00 pm on March 17th via Zoom.

## 6.14 Minutes of the 14th Project Meeting

Date: March 17, 2023

Time: 6:00 pm

Place: Zoom

Present: Dr. Gary CHAN, HE Qihao, LIN Zhaorun, YIN Zhuohao, ZHOU Siyuan

Absent: None

Recorder: HE Qihao

1. Approval of minutes

The minutes of the last meeting were approved without amendment.

2. Report on progress

There was not much progress from the last meeting (March 15).

3. Discussion items

3.1 The team presented the system to Dr. Gary CHAN.

3.2 Dr. Gary CHAN points out the lack of clarity and illustration of the main idea in the presentation and advised the team to better be more concise in the slides, and focus on the unique parts of the system while doing the talk.

4. Goals after this meeting

4.1 The team will make the system more general in a way that the detection results obtained from both camera settings (wall & ceiling) will be accurate.

4.2 The team will prepare to do a better presentation next time to deliver the system's uniqueness to the audience.

5. Meeting adjournment and next meeting

The meeting was adjourned at 7:00 pm.

The next meeting will be at 8:00 pm on April 2 via Zoom.

## 6.15 Minutes of the 15th Project Meeting

Date: April 2, 2023

Time: 8:00 pm

Place: Zoom

Present: HE Qihao, LIN Zhaorun, YIN Zhuohao, ZHOU Siyuan

Absent: None

Recorder: HE Qihao

1. Approval of minutes

The minutes of the last meeting were approved without amendment.

2. Report on progress

YIN Zhuohao and ZHOU Siyuan started working on a new set of slides.

3. Discussion items

The team confirmed that testing and preparing for the presentation are the top priorities currently.

4. Goals after this meeting

4.1 The team would start collecting more testing data on a regular basis, starting on April 6th.

4.2 LIN Zhaorun will look for a way to make the labeling of bounding boxes of parking spaces easier. (Label at the time of drawing)

4.3 HE Qihao, LIN Zhaorun, YIN Zhuohao, ZHOU Siyuan will work on the powerpoint given the template provided by YIN Zhuohao and the structure provided by ZHOU Siyuan.

5. Meeting adjournment and next meeting

The meeting was adjourned at 8:30 pm.

The next meeting will be at 12:00 pm on April 6 in the HKUST parking lot.

## 6.16 Minutes of the 16th Project Meeting

Date: April 6, 2023

Time: 1:00 pm

Place: HKUST parking lot LG5

Present: HE Qihao, LIN Zhaorun, YIN Zhuohao, ZHOU Siyuan

Absent: None

Recorder: HE Qihao

1. Approval of minutes

The minutes of the last meeting were approved without amendment.

2. Report on progress

2.1 LIN Zhaorun made the system UI more approachable, with all the functionality easy to use as directly interactable buttons.

2.2 HE Qihao proposed a new algorithm for CNN that filters out temporal noise to improve the model's robustness when random pedestrians walk by/stand in the parking spaces.

3. Discussion items

The team gathered at this parking lot mainly for collection of testing data, and the number of parking spaces to be collected is hoped to reach 200, where each time, the team is collecting images at four different spots (two on-the-wall, two attached-to-ceiling) where the camera is installed.

4. Goals after this meeting

The team will continue collecting data on parking spaces for a week, and the data will be used to demonstrate the capability of the system.

5. Meeting adjournment and next meeting

The meeting was adjourned at 4:00 pm.

The next meetings will be from 3:00 pm ~ 4:00 pm in the HKUST parking lot till April 12 (as described in 4).

# 7. Appendix B: Project Planning

## 7.1 Division of Work

| | HE Qihao | LIN Zhaorun | YIN Zhuohao | ZHOU Siyuan |
|---|---|---|---|---|
| **Research** | | | | |
| *Exploration of existing technologies* | 25% | 25% | 25% | 25% |
| *Survey of image undistortion methods* | 30% | 15% | 40% | 15% |
| *Literature review* | 5% | 5% | 85% | 5% |
| **Design** | | | | |
| *System pipeline* | 15% | 35% | 35% | 15% |
| *CNN architecture* | 0% | 0% | 0% | 100% |
| *Web page UI* | 5% | 85% | 5% | 5% |
| **Implementation** | | | | |
| *CNN training* | 15% | 0% | 0% | 85% |
| *CNN integration* | 80% | 0% | 0% | 20% |
| *YOLO integration* | 0% | 70% | 30% | 0% |
| *Parking spcae annotator* | 0% | 10% | 0% | 90% |
| *Undistortion algorithm* | 0% | 10% | 90% | 0% |
| *IoU mechanism* | 0% | 50% | 50% | 0% |
| *Testing* | 70% | 10% | 10% | 10% |
| **Documentation** | | | | |
| *Proposal* | 25% | 25% | 25% | 25% |
| *Monthly reports* | 70% | 10% | 10% | 10% |
| *Progress report* | 25% | 25% | 25% | 25% |
| *Final report* | 25% | 25% | 25% | 25% |

## 7.2 Gantt Chart

The Gantt chart of our project is shown below.

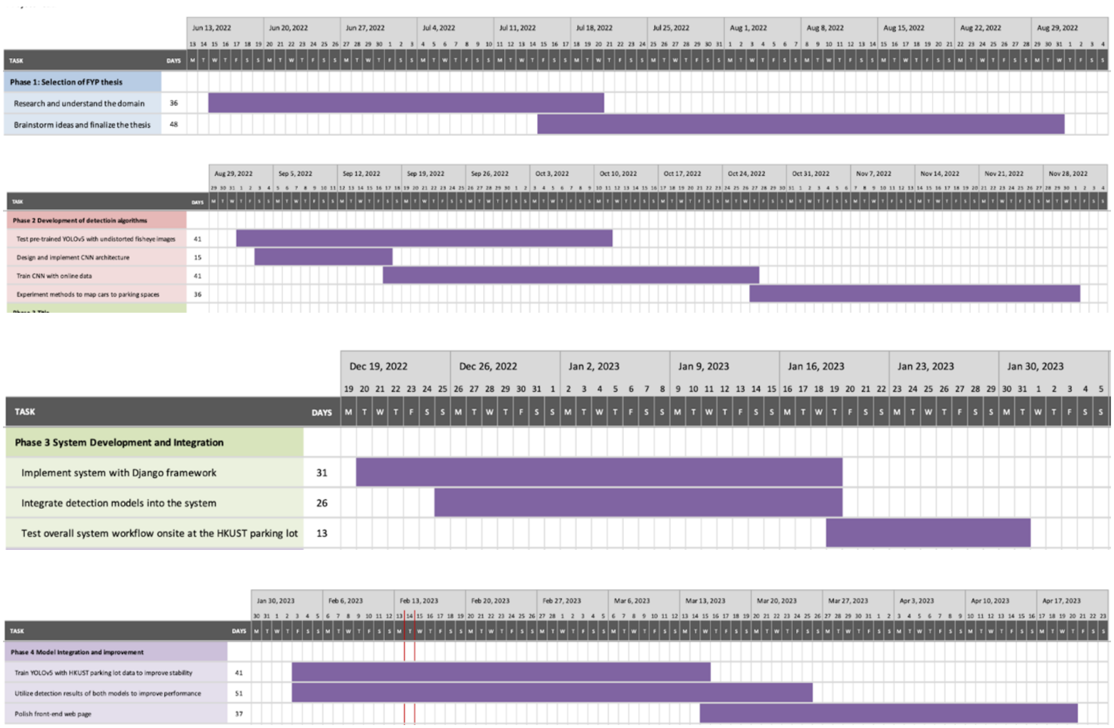



Fig. 10: Gantt chart of our project.

# 8. Appendix C: Hardware and Software Requirements

## 8.1 Hardware Requirements

For the whole system to operate, we so far used only one fisheye camera, but may need more for a larger capture area. We also used a router for data captured to be transmitted to the server for processing. And lastly, localhost, such as a laptop for undistorting images and making judgments on car occupancies by passing the undistorted images to the model we trained.

## 8.2 Software Requirements

The most crucial part of our FYP is the model we trained and the algorithm we used to undistort the raw image. We used a Linux environment (cloud server)/macOS or Windows (localhost) to host the models. For development, we used python to train the model. We also used the Django framework to integrate the system.