Prayer Hall Vacancy Detection

[1]Muhammad Naqib Syahmi bin Ab Razak, [1]Muhammad Imran bin Mohamad, [1]Dini Oktarina Dwi Handayani, [1]Zainab S. Attarbashi, [2]Sandra Hakiem Afrizal, [3]Zeldi Suryady
[1]*Kuliyyah of Information and Communication Technology, International Islamic University Malaysia, Selangor, Malaysia*
[2]*Public Health Master Programs, Faculty of Health Sciences and Technology, Universitas Binawan, Indonesia*
[3]*Telekom Malaysia Research and Development Sdn. Bhd (TM RnD) Malaysia*
naqib.syahmi@live.iium.edu.my, i.mohamad@live.iium.edu.my, dinihandayani@iium.edu.my, zainab_senan@iium.edu.my, sandra.hakiem@binawan.ac.id, zeldi.kamalurradat@tmrnd.com.my

2023 IEEE 9th International Conference on Computing, Engineering and Design (ICCED) | 979-8-3503-7012-6/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICCED60214.2023.10425587

***Abstract* - The issue of overcrowding in prayer rooms located in public areas, such as shopping centers, educational institutions, and other public spaces during prayer times, is a matter of concern nowadays. This condition has the potential to induce discomfort and adversely impact public health issues due to the increased risk of disease transmission, particularly respiratory illnesses. To address this issue, this paper presents a proof-of-concept system that uses computer vision techniques to detect, track, and count the number of people entering and exiting a prayer hall and subsequently count the total number of vacancies. The proposed system is based on the YOLOv5 object detection algorithm the Centroid Tracker object tracking algorithm and an existing object counting method. A custom dataset of 1,993 images of people entering and exiting a prayer hall was collected, cleaned, and annotated for use in training the system. The results of a sample video inference showed an average people detection mAP score of 98.9% and a people counting accuracy of 40% running on a Macbook Pro with an M1 chip. The counting results were displayed on a web app platform hosted on the Heroku cloud application platform.**

Keywords—proof-of-concept, computer vision, person detection, person tracking, person counting, vacancy counting, prayer hall, YOLOv5, Centroid Tracker, web app, Heroku.

I. Introduction

After the pandemic period, overcrowding in public areas including prayer rooms in shopping centers, educational institutions, and various communal spaces during prayer times, is a matter of significant concern. The consequences of this condition are many-sided, including discomfort amongst the congregation and adversely impacting public health [1]–[3]. This heightened risk primarily arises from the transmission of diseases, mainly respiratory illnesses due to a lack of distancing [4]–[6].

To address this issue, an effective measure to ensure the safe and optimise use of the prayer facility, particularly during peak prayer times is compulsory. Such improvements may include capacity limits, proper ventilation, enhanced sanitation protocols, and public awareness of the importance of healthy behavior in communal spaces [7]–[9]. By doing so, the adverse consequences of overcrowding can be reduced by public health promotions and maintain the solemnness of these communal prayer spaces. Thus, the present study aims to address the capacity limits issue of overcrowding in a medium-sized prayer hall by developing a proof-of-concept system that employs object detection and tracking algorithms to detect and track the number of individuals entering and exiting the prayer hall, subsequently displaying the available vacancies.

The system has been implemented and tested at the KICT men's prayer hall to optimize the utilization of prayer spaces and prevent overcrowding. The main objectives of this study are as follows: (1) to display the number of available vacancies in the prayer hall, (2) to evaluate the algorithms' capability in detecting and tracking individuals entering and exiting the prayer hall, and (3) to calculate the number of vacancies in the prayer hall. The system's ability to accurately detect and track individuals entering and exiting the prayer hall is important in ensuring compliance with safety regulations, such as preventing overcrowding. Furthermore, our system has the potential to enhance the overall experience of individuals needing to visit the prayer hall to perform their daily prayers by providing real-time information on the available vacancies in the prayer hall.

II. Literature Review

Several vacancy detection systems have been introduced in recent years and have been implemented and expanded in prominence in many areas, including parking lots, marketplaces, and many more. However, a comparable system has not been implemented in religious spaces, particularly Muslim prayer facilities. The evaluation of existing systems is necessary to acquire relevant information by evaluating the application requirements, which will help app developers in introducing a rather similar but improved system. Chandrasekaran, S., et al. [1] proposed employing a web-based application that allows users to check the occupancy of each parking block without requiring someone with technical expertise.

While this approach may be suitable for managing parking vacancy concerns, it has some security issues and cannot provide real-time occupancy detection due to the use of dynamic feeds. There have also been several studies that propose using deep learning algorithms to address parking occupancy issues [2][3][4][5]. While these deep learning techniques or methods are often the most efficient and fastest option for identifying car park vacancies, their effectiveness can be influenced by factors such as the time of day, environment, and many others. However, these techniques have the advantage of being able to recognize features automatically, without the need for human intervention, which simplifies and clarifies the implementation process.

Lastly, recent studies have suggested using YOLOv5, a state-of-the-art object detection algorithm, in combination with a robust object tracking algorithm like StrongSORT or DeepSORT, in order to improve the effectiveness of object counting [6][7]. However, these algorithms may not work as well in more complex cases involving two-way tracking, where the direction of an object's motion is difficult to track, such as in cases where objects are moving in opposite directions. (i.e. up and down, or left and right). Despite this, these techniques are still highly refined and widely used because of their fast and accurate performance.

III. Project Methodology

This section presents the methodology used for the development of our proposed prayer hall vacancy tracking system. It explains the implementation process from data collection up to deploying the system on the web, as well as describes the environmental setup used for testing the following system. This section also discusses the custom vacancy detection architecture that was created and how it was integrated with the YOLOv5 object detection algorithm and Centroid Tracker object tracking algorithm to track available vacancies by tracking people entering and exiting the prayer hall across different frames in the video sequences. The authors introduce a method for counting the number of people who cross through a designated region of interest, which indicates whether they have entered or exited the prayer hall and provides a count of the available vacancies in the prayer hall that can be displayed on the web.

3.1. System Architecture

Although many object tracking systems have been implemented in different environments in the past, such as in shops and offices, this system uses the latest advancements in AI. The authors utilized emerging object detection and tracking techniques that introduce new features and demonstrate significant improvements over previous techniques, and also introduce the system to a new environment, such as a prayer hall. Our proposed vacancy tracking system, shown in Figure 1, uses a custom vacancy detection algorithm based on the YOLOv5 object detection algorithm, combined with a Centroid Tracker object tracking algorithm, as well as incorporating an existing object counting method, and lastly, the results from the counting method will be displayed on the web.

The system consists of four main components, which are person (1) detection, (2) tracking, (3) counting, and (4) web deployment. The person detection component will be used for the detection of people, the people tracking component will be used to assign and track IDs for people, and the people counter component will be used to simulate, and count tracked people as they cross through a designated region of interest within and outside the prayer hall, and finally, the web deployment component will be used to display the count of available vacancies based on the current count of people tracked in the prayer hall (see figure 1).

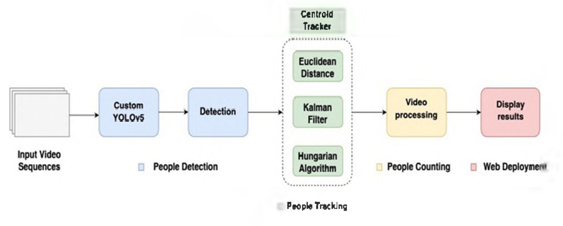


Fig. 1. System Architecture

3.2. Data Collection and Preparation

3.2.1 Data Collection

The dataset is made of video footage of the entrance to the KICT men's prayer hall. The data was collected over a week using two SriHome CCTV IP cameras, recording at a rate of 15 frames per second from noon to approximately 1.30 p.m. During this time, the prayer hall becomes quickly crowded as it is the time for Zuhr prayers, where students and staff will gather to perform their afternoon prayers. The cameras were placed roughly perpendicular to each other on the ceiling, with one inside and one outside the entrance to the prayer hall.

3.2.2 Data Augmentation

To prepare the data for further processing, the authors combined both video footages on top of each other to create a compiled video with a resolution of 1920 x 2160 pixels while maintaining its frame rate. We used Adobe Premiere Pro to edit both of the videos to ensure the people's motions were synchronized. This included making necessary adjustments to the timing and pacing of the videos so that they move in sync with each other. This step was crucial in creating smooth and cohesive data as shown in Figure 2-4.

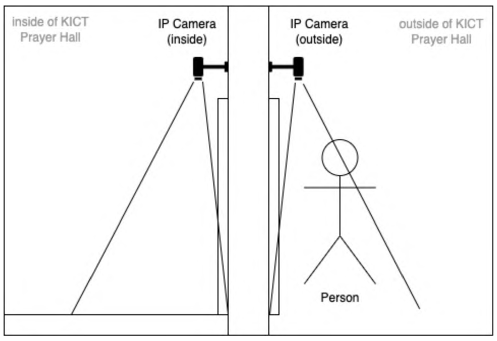


Fig. 2. IP Camera Placement For Data Collection

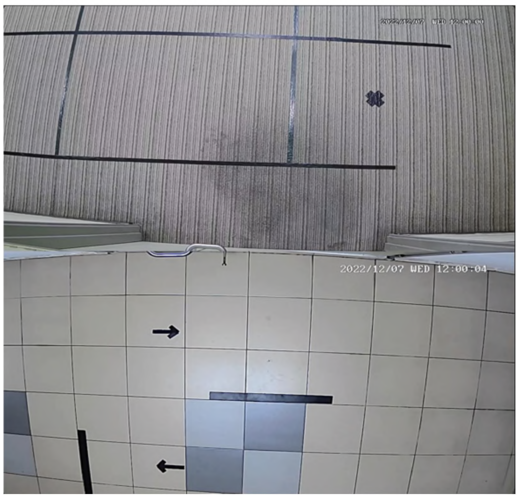
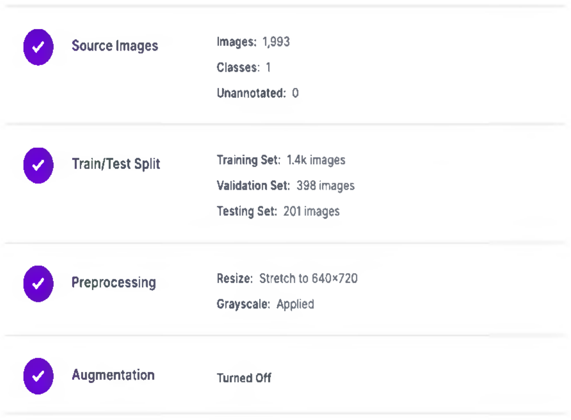


Fig. 3. Prayer Hall at Time (Noon)

Fig. 4. Prayer Hall at Time (1.30 PM)

3.2.3 Data Preparation

Each frame from the video footage was extracted into an image sequence, generating a total of 1,993 images of people entering and exiting the prayer hall and used to train YOLOv5 using our generated dataset on Roboflow, a computer vision framework that simplifies model training by combining the required techniques such as annotation, training and testing, preprocessing, and augmentation. The generated dataset was annotated, labeled, and exported in a YOLO text file format consisting of a single class, "Person", and is split into train, validation, and test sets with the following ratios 0.7, 0.2, and 0.1 respectively. To improve training accuracy, the size of each image was downscaled to 640 x 720 pixels while still maintaining its original ratio. Grayscale effects were also applied to all images. Figure 5 shows a summary of the required steps needed for model training, while Figure 6 shows a sample image annotated on Roboflow.



Source Images	Images: 1,993 Classes: 1 Unannotated: 0
Train/Test Split	Training Set: 1.4k images Validation Set: 398 images Testing Set: 201 images
Preprocessing	Resize: Stretch to 640×720 Grayscale: Applied
Augmentation	Turned Off

Fig. 5. Training Configuration on Roboflow

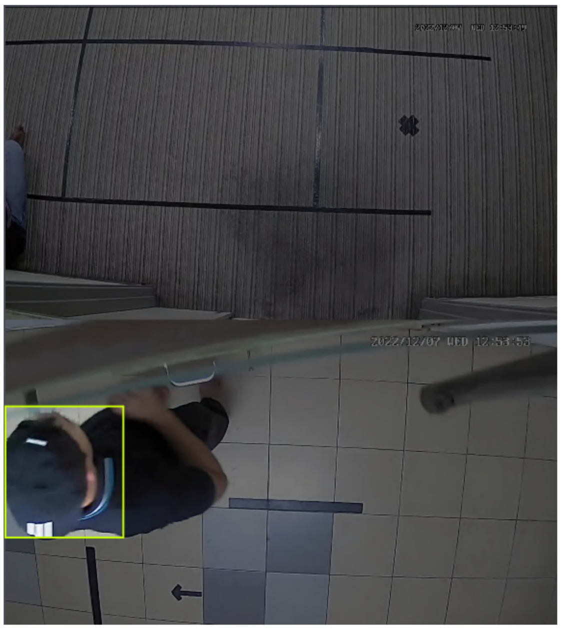


Fig. 6. Sample Image Annotated on Roboflow

3.3. Person Detection

For the system, the authors used a YOLOv5-based people detection algorithm. YOLOv5 is a popular object detection algorithm that is written and implemented in Python, using the PyTorch deep learning framework. It comes with five different model weights, which are the Nano, Small, Medium, Large, and Extra-Large, as shown in Figure 7. These weights differ in terms of the depth and width of the model, with the larger weights having a larger size and complexity than the smaller ones, but all of them have the same overall structure. The sample structures can also be customized by adding more neurons, increasing the number of hidden layers, and applying batch normalization or weight initialization. Thus, from the model training, the authors have made the custom trained YOLOv5m weights and modified the default structure of YOLOv5m by adding extra layers in order to improve the detection accuracy of fast-moving people, as shown in Figure 8.

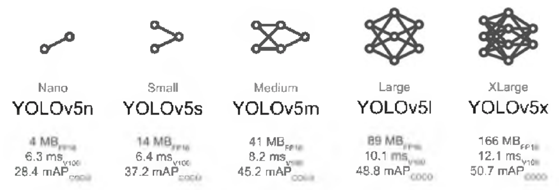
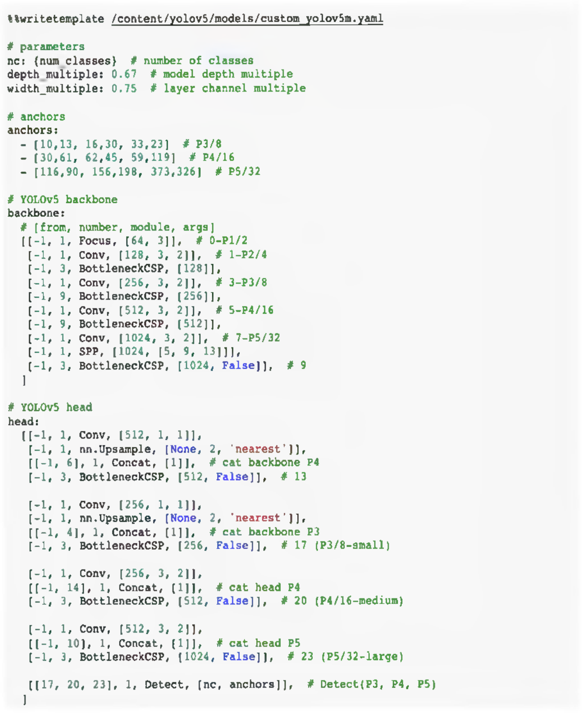


Fig. 7. Yollov5 Model Weights



```
%%writetemplate /content/yolov5/models/custom_yolov5m.yaml

# parameters
nc: {num_classes}  # number of classes
depth_multiple: 0.67  # model depth multiple
width_multiple: 0.75  # layer channel multiple

# anchors
anchors:
  - [10,13, 16,30, 33,23]  # P3/8
  - [30,61, 62,45, 59,119]  # P4/16
  - [116,90, 156,198, 373,326]  # P5/32

# YOLOv5 backbone
backbone:
  # [from, number, module, args]
  [[-1, 1, Focus, [64, 3]],  # 0-P1/2
   [-1, 1, Conv, [128, 3, 2]],  # 1-P2/4
   [-1, 3, BottleneckCSP, [128]],
   [-1, 1, Conv, [256, 3, 2]],  # 3-P3/8
   [-1, 9, BottleneckCSP, [256]],
   [-1, 1, Conv, [512, 3, 2]],  # 5-P4/16
   [-1, 9, BottleneckCSP, [512]],
   [-1, 1, Conv, [1024, 3, 2]],  # 7-P5/32
   [-1, 1, SPP, [1024, [5, 9, 13]]],
   [-1, 3, BottleneckCSP, [1024, False]],  # 9
  ]

# YOLOv5 head
head:
  [[-1, 1, Conv, [512, 1, 1]],
   [-1, 1, nn.Upsample, [None, 2, 'nearest']],
   [[-1, 6], 1, Concat, [1]],  # cat backbone P4
   [-1, 3, BottleneckCSP, [512, False]],  # 13

   [-1, 1, Conv, [256, 1, 1]],
   [-1, 1, nn.Upsample, [None, 2, 'nearest']],
   [[-1, 4], 1, Concat, [1]],  # cat backbone P3
   [-1, 3, BottleneckCSP, [256, False]],  # 17 (P3/8-small)

   [-1, 1, Conv, [256, 3, 2]],
   [[-1, 14], 1, Concat, [1]],  # cat head P4
   [-1, 3, BottleneckCSP, [512, False]],  # 20 (P4/16-medium)

   [-1, 1, Conv, [512, 3, 2]],
   [[-1, 10], 1, Concat, [1]],  # cat head P5
   [-1, 3, BottleneckCSP, [1024, False]],  # 23 (P5/32-large)

   [[17, 20, 23], 1, Detect, [nc, anchors]],  # Detect(P3, P4, P5)
  ]
```

Fig. 8. People Detection Model Architecture

3.4. Person Tracking

Once the people have been detected by the custom trained YOLOv5 algorithm, a Centroid Tracker object-tracking algorithm is used to track their movement as they enter and exit the prayer hall. The Centroid Tracker algorithm combines the Kalman Filter, Hungarian algorithm, and Euclidean Distance to track a person's motion. The Kalman Filter compares the current location of a centroid, represented by its bounding box coordinates, to its location in the previous frame and tries to determine whether it is the same person as they move through the video sequence. Euclidean distance is then used to account for any uncertainties made by the Kalman Filter by identifying pairs of centroids in subsequent frames that are considered the same person. Finally, once the person's location and motion have been derived, the Hungarian algorithm is used to assign a unique ID to each detected person and determine if the person in the current frame is the same as the person in the previous frame. Figure 9 discusses the flow of the Centroid Tracker algorithm.

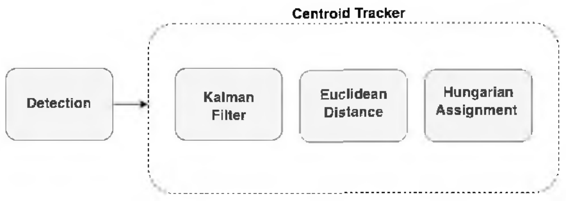


Fig. 9. Flow Centroid Tracker Algorithm

3.5. People Counting

Due to the limitations of our tracking algorithm when it comes to observing changes in the positions of centroids assigned to each detected person, this can result in inaccurate people counting due to ID switches throughout the video sequence. Therefore, to improve the robustness of our person counter, our system is not going to rely on the person tracker and ID assignment for the counting. We introduced a region of interest for accurately tracking people entering and exiting the prayer hall and count the total number of vacancies in the prayer hall. The region of interest works by dividing the scenes into two areas, IN area and OUT area as shown in Figure 10. IN area refers to the region that is located inside the prayer hall, whereas Figure 11 refers to the OUT area located outside the prayer hall. The person counting will be performed once a person has crossed either of these regions indicating whether they entered or exited the prayer hall.

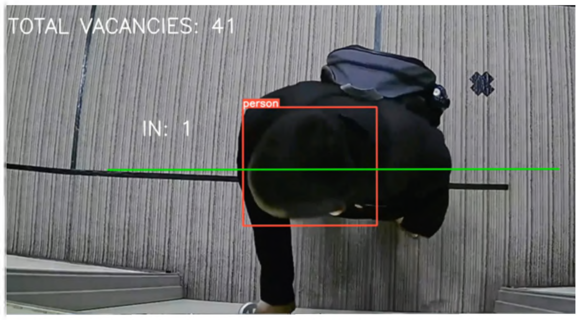


Fig. 10. In Area Used for Counting People

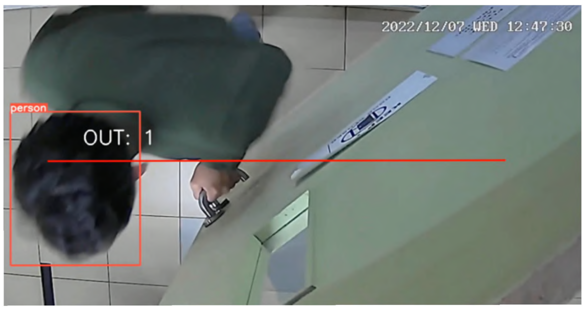


Fig. 11. Out Area Used for Counting People

3.6. Web Deployment

The output of the counting algorithm is sent to a Flask application that serves as the backend for the web server. The application, called 'app.py,' has two routes that handle web requests. The first route, '/', renders the index.html page, which is the interface of the system. The second route, '/api', provides the frontend with information in JSON format, including the number of vacancies in the prayer hall, the current date, the Hijri date, and the times for the five daily prayers (Fajr, Zuhr, Asr, Maghrib, and Isya). On the web interface, the website makes an API request to the server by using the route '/api' in order to fill in the necessary information in the html file. This request is done via an AJAX request function using the jQuery syntax.

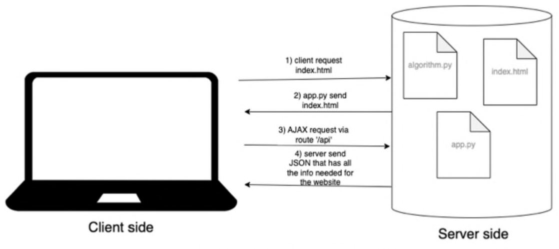

1) client request index.html
2) app.py send index.html
3) AJAX request via route '/api'
4) server send JSON that has all the info needed for the website
algorithm.py
index.html
app.py
Client side
Server side

Fig. 12. Example of Web Request

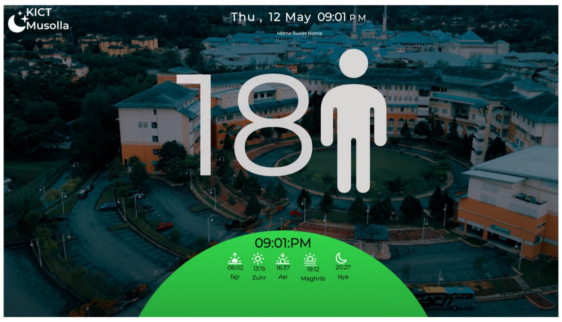


Fig. 13. Screenshot of The Web Interface

IV. Result

This section presents the performance evaluation of our proposed prayer hall vacancy tracking system, which was developed by utilizing a custom person detection algorithm and combined with a Centroid Tracker person tracking algorithm for detecting and tracking people entering and exiting the prayer hall and counting the total number of vacancies. It analyzes the performance of the trained person detection model and assess the efficiency of the person counting method. Finally, the results of our person detection model and total vacancy counting accuracy scores are presented and discussed.

4.1 Training Results

The person detection model was trained using 1,993 images, and its performance was validated using 398 images. The Mean Average Precision (mAP) and loss function plots were the main metrics used to evaluate the trained model. The model training results are shown in Table 1. The model was trained by modifying the YOLOv5m model architecture by adding more hidden layers and using training configurations from Roboflow. The model was trained for 100 epochs on a Macbook Pro with an M1 chip.

TABLE 1. TRAINED YOLOV5M PERSON DETECTION MODEL

Precision	Recall	mAP	Epochs
98.2 %	96.2 %	98.9 %	100

The training process took 1 hour and 14 minutes to complete, the loss function plots, shown in Figure 14, display three main types of losses which are box, object, and classification losses. The box regression loss indicates how accurately the algorithm was able to locate the center of a person and how well the predicted bounding boxes covered the detected person. The object loss reflects the probability that a person exists in a predicted region of interest, and the classification loss reflects the algorithm's ability to predict the person class given a detected person object.

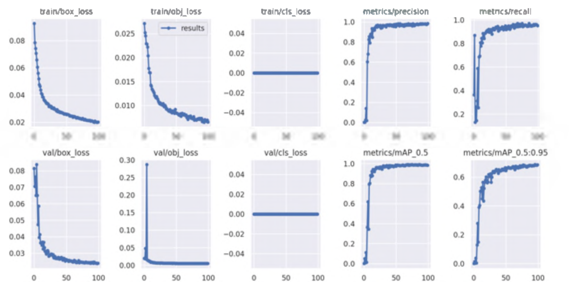


Fig. 14. Model Training Loss Function Plots The Validation Set

4.2 Experimental Results

The prayer hall vacancy detection system was tested using a web app platform that was deployed on the Heroku cloud application platform. The experimental setup was located at the KICT men's prayer hall at the International Islamic University Malaysia (IIUM) in Gombak, Selangor. The trained model's inference was run on a Macbook Pro with an M1 chip, which was then integrated with our web app platform to display the number of vacancies in the prayer hall. Figure 12 shows the deployed web app platform that was used for testing our system. The authors have observed and experimented with a sample video inference that has been collected. The video duration is 11:17 long and the inference results can also be accessed at the following link: tinyurl.com/b9j5tbsw.

V. Discussion

The loss function plots of the custom YOLOv5m model converged before reaching 50 epochs, with a loss of approximately between 0.04% and 0.05%. The mean average precision (mAP) is a measure of the model's performance in detecting people entering and exiting the prayer hall. Figure 11 shows that the mAP of the model gradually stabilized at around 97% before reaching 50 epochs of training. Training configurations from Roboflow were applied to the training images, which helped increase the recall of the model (the ability to correctly detect most of the people) and maintain a steady precision (a low number of false positive predictions of the people).

TABLE 2. PEOPLE COUNTING PERFORMANCE OF OUR SYSTEM

Video Name	Duration (minutes)	Class	States	Counting		Counting Error	Overall Accuracy
				Real	System		
Sample people entering and exiting the prayer hall - 115534_final2.mp4	11:17	Person	IN OUT	32 12	15 4	+17 +8	40%

The trained model's score was compared with vehicle counting models such as [6][7], which both used the YOLOv5 object detection architecture and DeepSORT object tracking algorithm to perform one-way tracking and counting of vehicles in highway scenes and achieved an average mAP score of 93%. In contrast, the average mAP score for tracking and counting people entering and exiting a prayer hall was 98%. As for the counting method, there are no existing systems with the level of complexity like the use case, which requires handling many variations of people entering and exiting the prayer hall that our system was not able to handle, such as when people enter or exit the prayer hall in groups or when people and exit at a fast pace, which our method was not able to detect. Thus, our proposed method had an average accuracy of 40%, which is not very accurate but still a good start since we are proposing a proof-of-concept system to demonstrate the potential of using current or future state-of-the-art object detection and tracking algorithms for detecting and tracking people entering and exiting a place, particularly in our use case of a prayer hall.

As a result, one of the limitations of our system that we have observed is that our object tracking algorithm (Centroid Tracker) is not able to accurately track the movement direction of fast-moving people, and it struggles to track the motion of multiple people entering or exiting the prayer hall at the same time. Next, the cameras used in our system are not positioned parallel to each other, causing variations in the video footage captured by each camera as the footage for the camera inside the prayer hall is slightly magnified while the camera outside the prayer hall is not magnified, which can negatively impact the efficiency of the object detection and tracking algorithm. Lastly, the wide-angle lenses on the cameras cause a slight curvature to the center of the video footage, which can make it difficult to accurately map the region of interest. Thus, these factors contribute to the system's accuracy in tracking the prayer hall vacancies.

VI. CONCLUSION

In conclusion, our proposed system, which was developed using a custom people detection algorithm based on the YOLOv5m model architecture and a Centroid Tracker object tracking algorithm, has not proven to be effective for the task of detecting tracking, and counting people and subsequently counting the total number of vacancies in a prayer hall. Additionally, our counting approach may not be the most efficient method for handling the many scenarios of people entering and exiting the prayer hall in order to produce accurate people counting results and avoid duplicate counts. Despite these limitations, our system was able to produce valid results due to the complexity of our case, and it can be improved in the future as a real-world application.

REFERENCES

[1] Chandrasekaran, S., Reginald, J. M., Wang, W., & Zhu, T. (2021). Computer Vision Based Parking Optimization System.

[2] Acharya, D., Yan, W., & Khoshelham, K. (2018). Real- time image-based parking occupancy detection using deep learning.

[3] Nyambal, J., & Klein, R. (2017). Automated parking space detection using convolutional neural networks. 1-6. 10.1109/RoboMech.2017.8261114.

[4] Farley, A., Ham, H., & Hendra. (2021). Real Time IP Camera Parking Occupancy Detection using Deep Learning, Procedia Computer Science, 17, 606-614. https://doi.org/10.1016/j.procs.2021.01.046

[5] Sai, N., V., R., A. (2019). Car Parking Occupancy Detection using YOLOv3.

[6] Saadeldin, A., Rashid, M, M. (2022). Video-Based Vehicle Counting and Analysis using YOLOv5 and DeepSORT with Deployment on Jetson Nano.

[7] Gao, B. (2022). Research on Two-Way Detection of YOLOV5s+Deep Sort Road Vehicles Based ON Attention Mechanism