Efficient Passenger Counting in Public Transport Based on Machine Learning

Chonlakorn Wiboonsiriruk
Dept. of Electrical Engineering
Faculty of Engineering
Kasetsart University
Bangkok, Thailand
chonlakorn.wi@ku.th

Ekachai Phaisangittisagul
Dept. of Electrical Engineering
Faculty of Engineering
Kasetsart University
Bangkok, Thailand
fengecp@ku.ac.th

Chadchai Srisurangkul
National Metal and Materials Technology Center
National Science and Technology Development Agency
Pathum Thani, Thailand
chadchas@mtec.or.th

Itsuo Kumazawa
Dept. of Information and Communications Engineering
School of Engineering
Tokyo Institute of Technology
Tokyo, Japan
kumazawa.i.aa@m.titech.ac.jp

***Abstract*—Public transportation is a crucial aspect of passenger transportation, with buses playing a vital role in the transportation service. Passenger counting is an essential tool for organizing and managing transportation services. However, manual counting is a tedious and time-consuming task, which is why computer vision algorithms are being utilized to make the process more efficient. In this study, different object detection algorithms combining with object tracking are investigated to compare passenger counting performance. The system employs the EfficientDet algorithm, which has demonstrated balanced performance in terms of speed and accuracy. Our results show that the EfficientDet can accurately count passengers in varying conditions with an accuracy of 94%.**

***Index Terms*—computer vision, object detection, passenger counting, public transportation**

I. Introduction

Passenger counting is a critical aspect of public transportation management, as accurate counts are essential for various purposes such as resource allocation, route planning, scheduling, and fare collection. Manual counting, which involves physically counting passengers, is often labor-intensive, time-consuming, and prone to errors, particularly in busy or crowded situations.

In recent years, computer vision algorithms have emerged as promising solutions for automating passenger counting in public transport systems. Previous approaches have included background subtraction methods, which analyze the difference between consecutive frames to detect moving objects [1], but they often struggle with complex scenes and lighting variations, leading to inaccurate counts. Another approach involves using feature-based techniques, where handcrafted features are extracted from images and used for counting. However, these methods are sensitive to changes in viewpoint and lighting conditions, limiting their effectiveness in real-world scenarios [2].

Video-based systems using computer vision algorithms have shown the most promise in accurately and efficiently counting passengers in real-time. Deep learning techniques, particularly in the context of object detection, have demonstrated superior performance compared to other methods [3]. These models can learn complex representations of passengers, enabling robust detection and tracking even in challenging conditions. However, accurately and efficiently counting passengers in public transport systems using these algorithms remains an ongoing research problem [4].

In this study, we aim to investigate and compare the performance of different deep learning models for passenger counting in public transport systems. Our objective is to develop a system that can significantly improve passenger management and optimize bus services. We propose a passenger counting system that utilizes deep learning algorithms optimized through transfer learning and a tracking algorithm to improve the accuracy of passenger counting in public transport systems. The proposed system is evaluated based on its ability to accurately and efficiently count passengers in varying conditions, such as crowded or low-light environments. We compare the performance of different deep learning models and evaluate the time and accuracy for each deep learning model. To achieve this objective, Section II will review related works in passenger counting. Section III will describe the proposed passenger counting system, including data preprocessing, model training, and evaluation. The experimental results and analysis will be described in section IV. Finally, Section V will conclude the paper and discuss future directions.

II. Related Works

A. Object Detection

Object detection is a fundamental computer vision task that involves the identification and localization of objects within digital images or video frames. The primary objective of object detection is to discern the presence of specific objects in an image and precisely locate them by defining bounding boxes around them. There are several popular approaches to object detection, including traditional computer vision techniques like Haar cascades [5] and Histogram of Oriented Gradients (HOG) [6], as well as deep learning-based approaches [7]-[10]. In this paper, we will focus on deep learning-based approaches.

EfficientDet is a single neural network, a cutting-edge object detection method, as shown in Figure 1, to recognize objects of various shapes and scales. For the reason to balance the trade-off between accuracy and efficiency, it employs a compound scaling algorithm. In order to estimate the bounding boxes and class probabilities of an object in the images, EfficientDet first applies a backbone neural network to extract characteristics from the input image. For the purpose of to increase the accuracy of object recognition, EfficientDet additionally utilizes a BiFPN (Bidirectional Feature Pyramid Network) to combine information across different level of the backbone network layers.

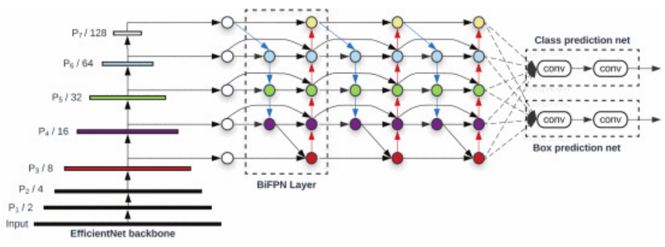

Fig. 1: EfficientDet architecture [7].

SSD (Single Shot MultiBox Detector) is an object detection algorithm as shown in Figure 2. It operates by utilizing a single neural network to identify objects within an image. This process involves dividing the input image into a grid composed of cells. Within each cell, the algorithm predicts the respective object's bounding boxes and associated class probabilities [8]. To achieve this, SSD employs a sequence of convolutional layers that extract pertinent features from the input image. Subsequently, a collection of convolutional filters is applied to forecast the bounding boxes and class probabilities for objects within each cell. To refine its results, SSD incorporates a technique known as non-maximum suppression. This method effectively removes redundant bounding boxes with lower confidence scores, ensuring that only the most reliable and accurate ones are retained. Additionally, SSD integrates anchor boxes into its process to enhance the precision of bounding box predictions.

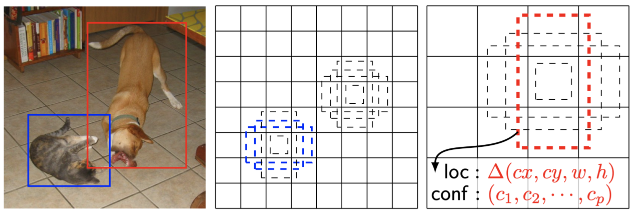
loc : $\Delta(cx, cy, w, h)$
conf : $(c_1, c_2, \cdots, c_p)$

Fig. 2: SSD architecture [8].

Faster R-CNN (Region-based Convolutional Neural Network), as shown in Figure 3, represents a two-stage approach for object detection. It employs a region proposal network (RPN) to generate potential candidate regions containing objects. Subsequently, a second network is employed to classify and refine these proposals [9]. The RPN operates by sliding a compact network across the convolutional feature map of the input image, producing a collection of object proposals. The subsequent network takes these proposed regions as input and yields the final predictions, including bounding boxes and class probabilities for the detected objects within the image. To enhance the precision of bounding box predictions, Faster R-CNN incorporates anchor boxes.

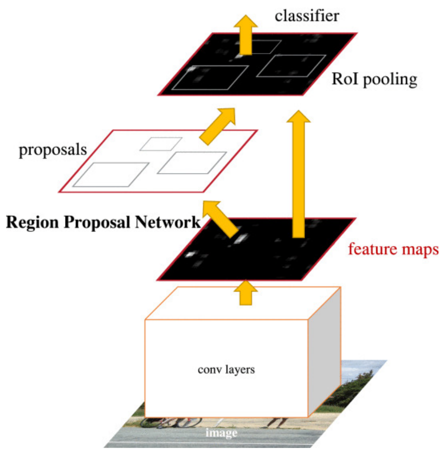

Fig. 3: Faster R-CNN architecture [9].

YOLOv3 (You Only Look Once version 3), as shown in Figure 4, stands as a real-time object detection algorithm that capitalizes on a single neural network to achieve object identification within images. The methodology involves partitioning the input image into a grid comprised of cells, wherein predictions for object bounding boxes and their associated class probabilities are made for each cell [10]. YOLOv3 integrates a sequence of convolutional layers to abstract salient features from the input image. Subsequently, a set of convolutional filters are employed to forecast bounding boxes and class probabilities for objects situated within each cell. The algorithm additionally leverages anchor boxes to enhance the precision of its bounding box predictions. Furthermore, YOLOv3 employs a feature pyramid network to effectively detect objects across various scales, thereby elevating the overall accuracy of object detection.

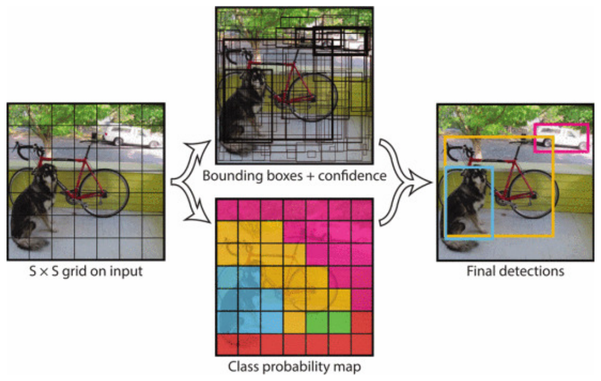

Fig. 4: YOLOv3 architecture [10].

YOLOv3 is an improvement over its predecessors [11], YOLOv2 and YOLOv1, in several ways. Firstly, YOLOv3 uses a feature pyramid network to detect objects at different scales, which improves the accuracy of object detection. YOLOv2 also used a feature pyramid network, but YOLOv1 did not. Secondly, YOLOv3 uses anchor boxes to improve the accuracy of the bounding box predictions. Anchor boxes were not used in YOLOv1, and YOLOv2 used a different method for bounding box prediction. Finally, YOLOv3 has better accuracy than its predecessors, achieving state-of-the-art performance on several object detection benchmarks.

B. Object Tracking

Object tracking refers to a computer vision method employed to track the motion of particular objects within a series of frames or a video. It entails recognizing and pinpointing a specific object of interest as it moves throughout the video over time. Various algorithms and techniques exist for object tracking, such as Kalman Filters [12] and Optical Flow [13]. In this study, centroid tracking is utilized due to its advantages in terms of real-time performance, object localization, and resilience to changes in appearance.

The centroid tracker algorithm is a widely employed technique for both detecting and tracking objects. Its operation involves identifying the centroid coordinates of an object's bounding box and utilizing this information to trail the object through consecutive frames. Multiple research studies have demonstrated the effectiveness of this method [14]-[16]. The basis of the centroid tracker rests on associating object IDs with centroid pairs that exhibit the closest Euclidean distance [15]. Furthermore, certain investigations have integrated the Kalman filter [12] to anticipate the centroid's location for each track within the ongoing frame, thus allowing the adjustment of the associated bounding box [14]. In similar vein, alternative research [17] has explored ellipse fitting as a means to estimate both the dynamic state of the centroid and the physical extension of the object.

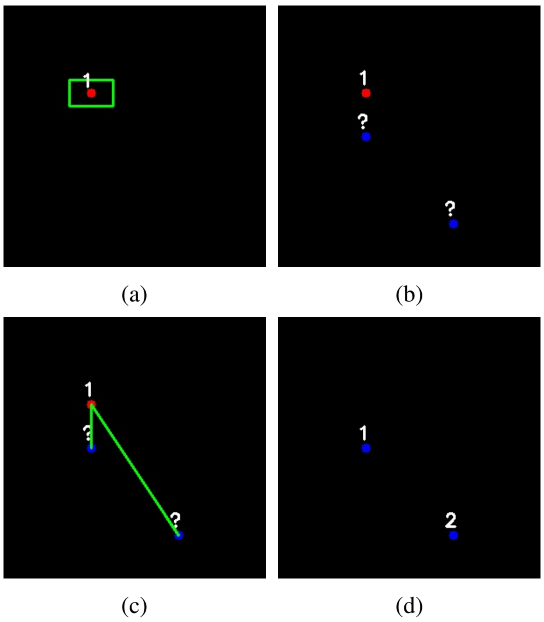

Fig. 5: Centroid tracker algorithm.

The basic centroid tracking algorithm is shown in Figure 5. Once we obtain the bounding box, we calculate the centroid point for each bounding box, as shown in Figure 5 (a). In Figure 5 (b), our objective is to assign IDs to the new centroid point (indicated in blue). To assign IDs, we compute the Euclidean distance, as shown in Figure 5 (c), between the previous centroid (indicated in red) and the new centroid. The centroid with the closest distance is assigned the same ID, while a new ID is assigned to another centroid, as shown in Figure 5 (d).

III. Methodology

The proposed methodology consists of following processes.

A. Preprocessing

We utilized a comprehensive dataset that was acquired from a open-source repository [17]. The dataset itself comprised a collection of overhead perspective videos, as shown in Figure 6, which were meticulously recorded to capture the boarding and exiting behaviors of bus passengers. To accomplish this, a Kinect V1 Camera was strategically positioned at the top of the bus entrance, enabling a comprehensive view of the entire scene.

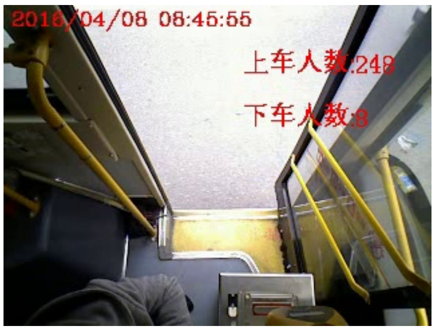

Fig. 6: The perspective of the camera.

Furthermore, in order to facilitate in-depth analysis and examination of specific activities and behaviors related to the passengers' heads, the dataset underwent a meticulous annotation process. Each image within the dataset was meticulously labeled to highlight and delineate the regions corresponding to the heads of the individuals. These annotations, which are visually shown in Figure 7, provide a precise reference for the head positions and movements of the bus passengers throughout the boarding and exiting process.

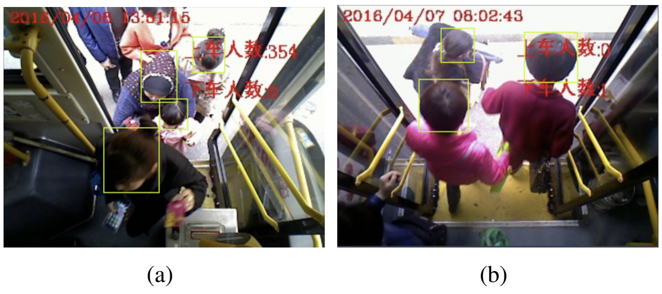

Fig. 7: Examples of bus passenger used in this study.

The proposed passenger counting algorithm consists of three main components: passenger detection, passenger tracking, and passenger counting. These components work together to create an effective system for analyzing passenger movement in a given area. The workflow of the system is shown in Figure 8. the video is initially fed into the detection model to identify passengers, and the bounding box information from the detection step is utilized to track the same passengers across consecutive frames. Finally, if a passenger crosses the counting zone, the system increments the passenger count.

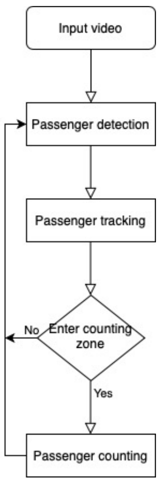

Fig. 8: The purposed passenger counting system.

B. Dectection

In each model employed within the algorithm, a transfer learning approach is utilized to enhance model accuracy. Transfer learning involves leveraging the knowledge and pre-trained weights from a base model, which has been trained on a large-scale dataset, and fine-tuning it for the specific task at hand. This process allows the model to benefit from the features learned from diverse data, resulting in improved generalization to new and unseen data, ultimately leading to enhanced performance.

Once the models are trained or pre-trained, the algorithm proceeds to load input pictures or frames into the respective models. These models, equipped with their learned knowledge, process the images and perform object detection, with a specific focus on detecting individuals within the images.

During the processing stage, the models analyze the images and generate bounding boxes around the detected individuals. These bounding boxes represent rectangular regions that accurately localize the positions of the individuals within the frame, as shown in Figure 9. The use of bounding boxes ensures precise identification and tracking of individuals within the images.

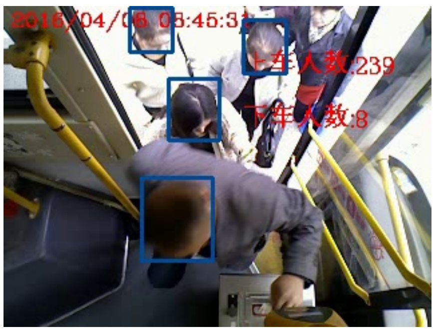

Fig. 9: Passenger detection.

To ensure the reliability of detections, the algorithm incorporates a filtering step. This step involves eliminating or filtering out bounding boxes with low-confidence scores from further consideration. Low-confidence scores typically indicate uncertain or less reliable predictions. By excluding these bounding boxes, the algorithm prioritizes more reliable predictions, thereby minimizing false positives or poorly detected regions.

C. Tracking and Counting

Centroid tracking technique is a widely used method for object tracking. It can handle occlusions, where people may temporarily block or overlap with each other. By analyzing the change in centroid positions, the algorithm can correctly associate the centroids with the corresponding individuals, even when they are partially obscured.

Centroid tracking is an object tracking algorithm that utilizes the centroids, which represent the center points, of bounding boxes obtained from object detection. The distance computation, as shown in Figure 10, is performed using a Euclidean distance. This metric measure the spatial separation between two centroids in different frames.

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{1}$$

where (x_1, y_1) and (x_2, y_2) are the coordinates of the current and previous centroids, respectively.

If the distance between the centroids is within the specified threshold, the algorithm associates the current centroid with the previous centroid, indicating that the object is likely the same. This association allows for consistent tracking of objects across frames. On the other hand, if the distance exceeds the threshold, the algorithm assumes that the object has moved significantly or that it might be a different object.

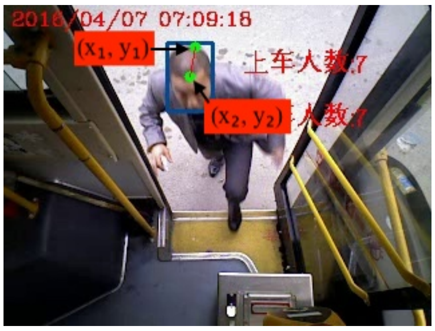

Fig. 10: Passenger tracking.

The proposed system combines object tracking with the use of a virtual line, as shown in Figure 11 (a), as a reference point to estimate the number of passengers boarding or exiting from a bus. We try to find the best position for the system, and during our research, we discovered that a particular location is well-suited for our requirements. It is important to avoid extreme brightness or darkness as they can make it difficult to detect objects effectively. When a passenger crosses the virtual line, as shown in Figure 11 (b), the algorithm increments the count of passengers. To determine whether it is an entry or an exit, the algorithm analyzes the position of the first passenger that appears in the frame. By tracking this initial position, the system can infer whether the passenger is entering or exiting the bus based on their movement relative to the virtual line.

IV. Experimental Results

The experiment aimed to investigate the performance of different object detection models in the task of counting the

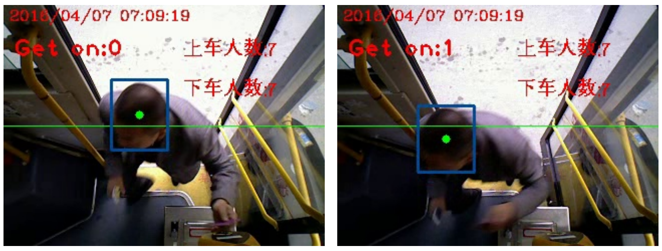
(a) (b)

Fig. 11: Passenger counting.

number of passengers getting on and off a bus. The dataset used in the experiment consisted of 5,464 videos, which were captured from bus surveillance cameras. The dataset is sourced from an open-source repository [17].

To conduct the experiment, the dataset was divided into a training set and a test set. Specifically, 4,370 videos were allocated for training, while 1,094 videos were reserved for testing. The training set was used to train each of the object detection models, teaching them to detect and classify individuals as they entered or exited the bus.

The experiment was performed on Google Colab, utilizing a CPU with a clock speed of 2.2GHz and a Tesla T4 GPU. These computational resources were leveraged to facilitate the training phase of the object detection models using the provided dataset.

Following the training phase, the models were tested using 10 randomly selected test videos. These videos were chosen to represent various scenarios, including different lighting conditions and varying passenger densities.

TABLE I
Experimental results of different object detection models in passenger counting

Object detection model	***Real In/Out***	***Count In/Out***	***Accuracy***	***FPS***
EfficientDet	140/121	128 / 118	94.26%	26
SSD		124 / 118	92.73%	35
Faster R-CNN		127 / 117	93.49%	8
YOLOv3		129 / 120	95.41%	16
YOLOv3 tiny		117 / 112	87.74%	59

The experimental results, as shown in Table 1, indicate that the YOLOv3 model achieves the highest accuracy, reaching an impressive 95.41%. However, its corresponding frames per second (FPS) value is 16, rendering it unsuitable for real-time applications.

In terms of real-time processing, the EfficientDet, SSD, and YOLOv3 tiny models exhibit more promising characteristics. Among these models, EfficientDet stands out as the most balanced model, achieving high accuracy while maintaining a reasonable FPS rate.

Eventhough Faster R-CNN displays slightly different accuracy compared to EfficientDet and SSD, it falls short due to its significantly lower FPS. Consequently, Faster R-CNN cannot effectively handle real-time processing.

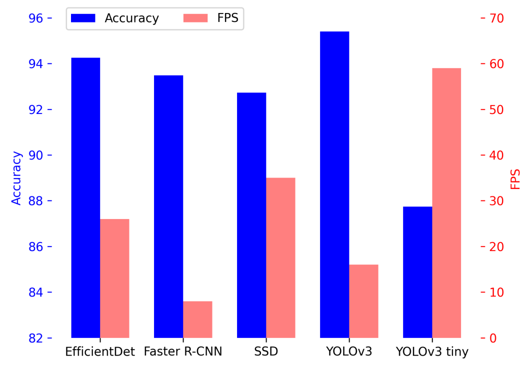

Fig. 12: Comparison of Model Performance for Passenger Counting: Accuracy and FPS.

Based on Figure 12, EfficientDet show promising result that achieves high accuracy while maintaining a relatively high fps. It uses a compound scaling method that optimizes the model architecture and input resolution to achieve high accuracy with fewer parameters and computations. This makes it more efficient and faster than other models while maintaining high accuracy.

YOLOv3 tiny, on the other hand, has the lowest accuracy among the models listed. This is because it is a smaller and simpler version of YOLOv3, which sacrifices accuracy for speed and efficiency. YOLOv3 tiny uses fewer layers and smaller input resolution, which reduces the accuracy of the model. Additionally, it has a higher fps because it uses fewer computations and parameters, but this comes at the cost of lower accuracy.

V. Conclusion

In this study, we compared the performance of several popular object detection models for real-time video processing. Based on our evaluation, EfficientDet emerged as the most favorable choice, achieving an impressive accuracy of 94.26% while maintaining a frame rate of 26 fps. These results demonstrate that EfficientDet surpasses the required threshold of 25 fps, making it well-suited for real-time object detection applications. However, it's important to consider other factors such as model size and specific use cases when selecting the most appropriate model for a given scenario.

References

[1] J. -W. Perng, T. -Y. Wang, Y. -W. Hsu and B. -F. Wu, "The design and implementation of a vision-based people counting system in buses," 2016 International Conference on System Science and Engineering (ICSSE), Puli, Taiwan, 2016, pp. 1-3.

[2] C. -H. Chen, Y. -C. Chang, T. -Y. Chen and D. -J. Wang, "People Counting System for Getting In/Out of a Bus Based on Video Processing," 2008 Eighth International Conference on Intelligent Systems Design and Applications, Kaohsuing, Taiwan, 2008, pp. 565-569.

[3] D. Baumann, M. Sommer, Y. Schrempp and E. Sax, "Use of Deep Learning Methods for People Counting in Public Transport," 2022 International Conference on Connected Vehicle and Expo (ICCVE), Lakeland, FL, USA, 2022, pp. 1-6.

[4] S. Khan, M. H. Yousaf, F. Murtaza, and S. Velastin, "PASSENGER DETECTION AND COUNTING FOR PUBLIC TRANSPORT SYSTEM," NED University Journal of Research, vol. XVII, pp. 35–46, Mar. 2020.

[5] P. Viola and M. Jones, ”Rapid object detection using a boosted cascade of simple features,” Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001, Kauai, HI, USA, 2001, pp. I-I.

[6] M. Li, Z. Zhang, K. Huang and T. Tan, ”Estimating the number of people in crowded scenes by MID based foreground segmentation and head-shoulder detection,” 2008 19th International Conference on Pattern Recognition, Tampa, FL, USA, 2008, pp. 1-4.

[7] M. Tan, R. Pang, and Q. Le, “EfficientDet: Scalable and Efficient Object Detection,” Jun. 2020, pp. 10778–10787.

[8] W. Liu et al., “SSD: Single Shot MultiBox Detector,” in Computer Vision – ECCV 2016, Springer International Publishing, 2016, pp. 21–37.

[9] S. Ren, K. He, R. Girshick, and J. Sun, Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. 2016.

[10] J. Redmon, S. Divvala, R. Girshick and A. Farhadi, ”You Only Look Once: Unified, Real-Time Object Detection,” 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016, pp. 779-788.

[11] J. Redmon and A. Farhadi, YOLOv3: An Incremental Improvement. 2018.

[12] D. Vignarca, J. Prakash, M. Vignati and E. Sabbioni, ”Improved Person Counting Performance Using Kalman Filter Based on Image Detection and Tracking,” 2021 AEIT International Conference on Electrical and Electronic Technologies for Automotive (AEIT AUTOMOTIVE), Torino, Italy, 2021, pp. 1-6.

[13] A. Tokta and A. K. Hocaoglu, ”A Fast People Counting Method Based on Optical Flow,” 2018 International Conference on Artificial Intelligence and Data Processing (IDAP), Malatya, Turkey, 2018, pp. 1-4.

[14] H. Vadlamudi, “Evaluation of Object Tracking System using Open-CV In Python,” International Journal of Engineering Research and, vol. V9, Sep. 2020.

[15] venkata sriharsha Kuncham, N. Mahamkali, and V. Ayyasamy, “Moving Object Detection and Tracking Based on the Contour Extraction and Centroid Representation,” 2017, pp. 212–219.

[16] J. Singh, “Tracking of Moving Object Using Centroid based Prediction and Boundary Tracing Scheme,” International Journal of Image, Graphics and Signal Processing, vol. 9, no. 8, pp. 59–66, Aug. 2017.

[17] S. Sun, N. Akhtar, H. Song, C. Zhang, J. Li and A. Mian, ”Benchmark Data and Method for Real-Time People Counting in Cluttered Scenes Using Depth Sensors,” in IEEE Transactions on Intelligent Transportation Systems, vol. 20, no. 10, pp. 3599-3612, Oct. 2019.