Detection of Operators' Inspection Quality in Car Factory

Cheng-Wei Lin (林正偉)[1] **Chiou-Shann Fuh (傅楸善)[1]**

[1]Department of Computer Science and Information Engineering

National Taiwan University, Taipei, Taiwan 10617

[1]r10922102@ntu.edu.tw **[2]fuh@csie.ntu.edu.tw**

論文摘要

本論文提出電腦視覺、影像處理與深度學習的解決方案，用於檢測汽車製造廠內操作員動作檢測與品質確保。論文研究目的在於，建立與維護一個系統，用以確認汽車製造廠內操作員確實把每台汽車應檢的部分(檢測點)確實檢測。

我們設置了兩台攝影機，安裝在生產線的兩邊，用來偵測汽車和作業員的位置。汽車工廠定義了汽車的檢測點。在初始時，這些點未檢測，標記為灰色。當作業員的手套碰到檢測點時，該點會轉成綠色，標記為已檢測。在汽車通過檢測區域後，我們會保存檢測結果，並開始下一輛車的檢測流程。

我們系統中用到了深度學習和影像處理演算法。首先使用物件偵測演算法找到車輪，接著將檢測點利用對車輪的相對位置投影在車身上。接著使用人體姿態估計演算法 Blazepose，偵測作業員的姿勢。當偵測到作業員手套碰到檢測點時，該點視為已檢測。

關鍵詞：物件偵測、人體姿態估計、機器學習、影像處理

Abstract

In this paper, we propose a computer vision, image processing, and automatic optical inspection solution to detect operator motion and quality assurance in the car factory. Our goal is to create a system to check the operators have inspected all inspection points on each car.

We set up two cameras on both sides of production line to detect car and operator locations. Inspection points defined by car factory are mapped onto the car. Initially, these points are uninspected and marked by gray. When the operator's glove touches the inspection point, the inspected points are changed into green as inspected. After the car passes through the inspected area, we save inspection result, and inspect next car.

Deep learning and image processing algorithm are used in our system. We first use object detection algorithm to detect the wheel and map inspection points onto the car by relative position to wheels. Then, we find the operator by pose estimation algorithm, Blazepose. When the glove touches inspection point, then that point will be recorded as inspected (changed from gray to green).

Keywords: Object Detection, Human Pose Estimation, Machine Learning, Image Processing.

1. Introduction

On the production line of car, the common approach to check whether the car has defect is that every car will be transported through the production line conveyor belt. In the process of the inspection, the operator needs to establish a standard operating procedure to ensure that the inspection process of each vehicle is the same, and every quality point is touched, so that the quality of each vehicle can be guaranteed to be consistent. Operators will be assigned to inspection station on production line. When the vehicle reaches the working range, operators will check the vehicle with a standard operating procedure and touch every quality point. This can effectively improve yield rate.

We aim to establish inspection system for quality assurance. This inspection system will confirm whether each vehicle on the production line has been detected by the operator at the inspection points. Each vehicle has pre-defined quality inspection points, and each operator has a Standard Operating Procedure (SOP) for inspecting the vehicle. After the inspection procedure, the vehicle will have an inspection score.

Some problems need to be considered in our system. First, because the camera must be installed on the production line, and the vehicle will move on the production line. The working range is fixed. The most suitable angle and location for camera must be selected to maximize the depth of the effective image, that is, to maximize the time that we can see the vehicle [1].

Second, we take real-time video and analyze each frame in the video. Because the vehicle will move on the production line, there are some differences between frames, for example: the reflection of the car door is different when the car moves on the production line. Besides, the car door becomes smaller and smaller due to the depth of field.

Third, there would be several operators on the production line, and the movement of the operators may occlude the detection results in the video.

We aim to overcome these difficulties and achieve real-time detection and tracking of vehicles on the production line.

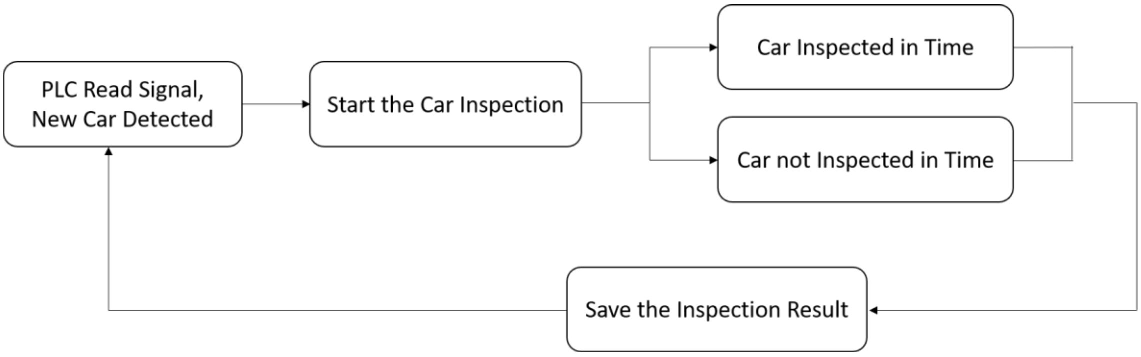

Figure 1 System flow.

Figure 2: The scene of the car on the production line [1].

2. Related Works

Since we aim to check whether the operator inspects car defect correctly in real time. The task can be divided into two parts: 1. Find the location of inspection points on the car. 2. Track the pose of operator.

2.1 Object Detection

To find the location of inspection points, we need to know the location of the car first, so the object detection model is used. Object detector is usually categorized into two kinds: One-stage object detector and two-stage object detector. One-stage object detector performs region prediction and classification using a single network only; two-stage object detector performs these operations using two different networks. It predicts the bounding-box with the feature of image first, and then classifies all of the bounding-boxes. Thus, one-stage object detector usually performs faster.

Considering training time and inference speed, we decided to use YOLOv3 (You Only Look Once version 3), a single stage real-time object model.

2.2 Human Pose Estimation

We track the operator with pose estimation algorithm to detect operator in image by predicting key-points of body parts. Openpose is known to be a state-of-the-art method which achieves successful accuracy in this task and won the COCO (Common Object in Context) Key-Point Detection Challenge 2016. However, its computation cost is heavy, and the frame rate (0.4FPS) is not high enough for our requirement. Due to the need of inference speed and light-weight architecture, an on-device model, BlazePose is more suitable in this task. BlazePose uses an encoder-decoder network architecture to predict heatmaps for all joints, followed by another encoder that regresses directly to the coordinates of all joints. In contrast to OpenPose, BlazePose shows slightly worse performance than the OpenPose model, but much faster than OpenPose.

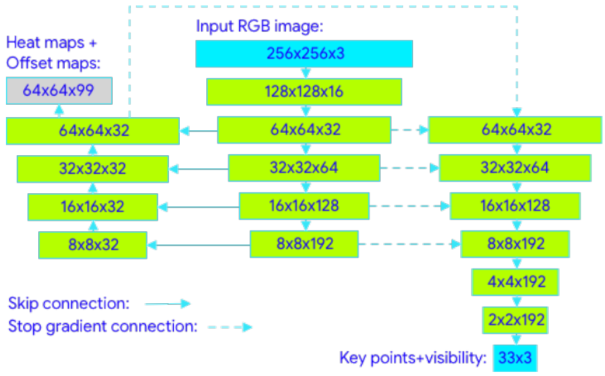

Figure 3: Network architecture of BlazePose [2].

3. Methodology

To detect the inspection quality of operators in car factory, we develop and maintain the whole system and install it on the production line. We set up a camera on each side of the production line, and make the working range as long as possible. Next, we create a Graphic User Interface (GUI) to enable operators and managers to use this system. Finally, we propose an algorithm to detect the inspection quality, achieve the goal of our system in Figure 1.

3.1 System Design and Flow

First, we install a sensor on the ceiling of production line. The sensor is controlled by Programmable Logic Controller (PLC) widely used for the control of manufacturing processes. When a car passes through the sensor, PLC will send a signal to the computer in Figure 5.

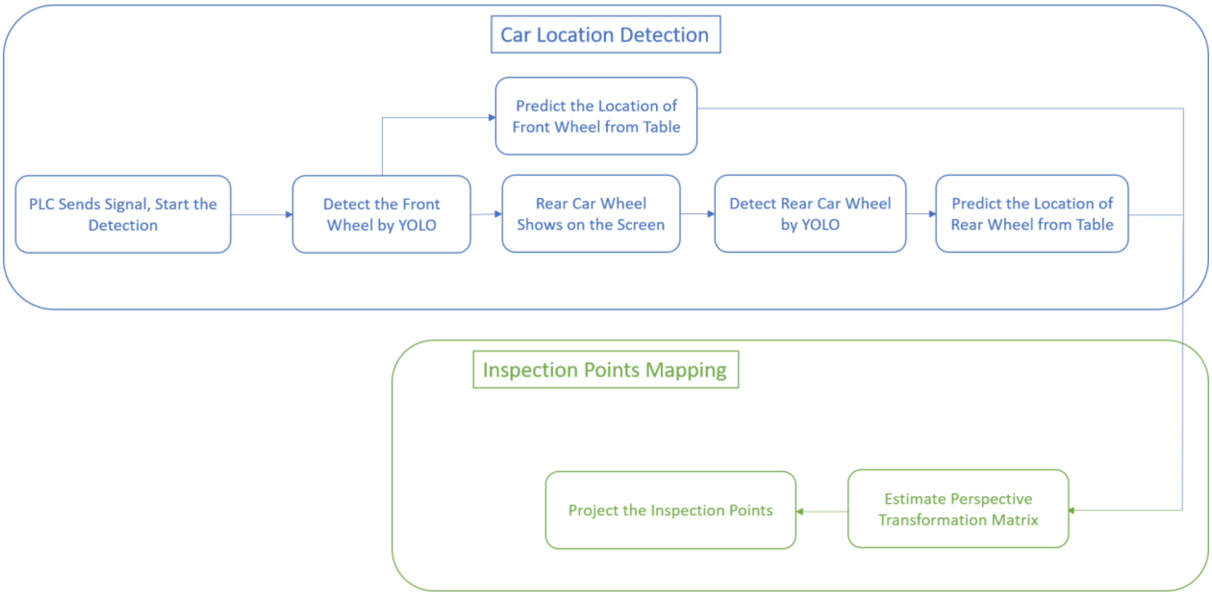

Figure 4: Car detection algorithm.

When computer receives the signal indicating one car passes through the production line, it queries the car number from their server, and starts to record the video of inspection process for this car, and starts the inspection algorithm. The inspection process will end when the inspection time of this car reaches the time limit. After inspection process, the video and the inspection result will be saved in the folder with respect to this car. When PLC senses another car on the car production line, another inspection process will be triggered and start to record and apply the algorithm again.

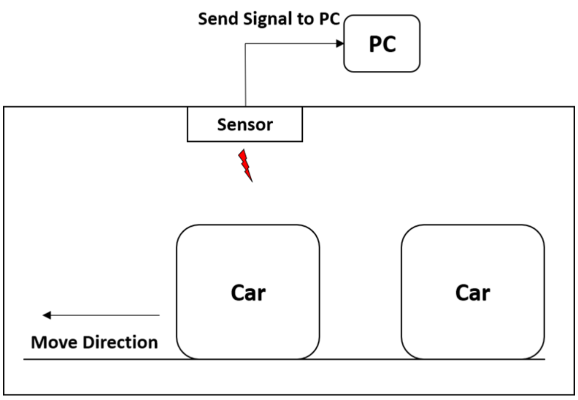

Figure 5: The location of sensor [1].

3.2 Graphic User Interface

To simplify our system for production line workers, we develop a GUI to meet their different requirements in Figure 6.

Car inspection system video is shown in the middle of the GUI. The start button is to activate the car inspection algorithm. The close button is to close all the service and the program. The continue / pause button is to continue or pause the procedure.

Our system will query the car number at the beginning of the inspection process. However, due to schedule variation, the car number may not always be correct. If the car number is incorrect, then the manager has to change the car number manually. Input correct car number in the textbox and press car number change button, then the car number will be modified.

The grids in lower left record the inspection result of last 100 cars. The inspection score is divided into 3 levels. The highest level (100%) will be recorded by green, the middle level (60%-99%) will be recorded by yellow, the lowest level (0%-59%) will be recorded by red. Press the button and the system will display the inspection snapshot of that car.

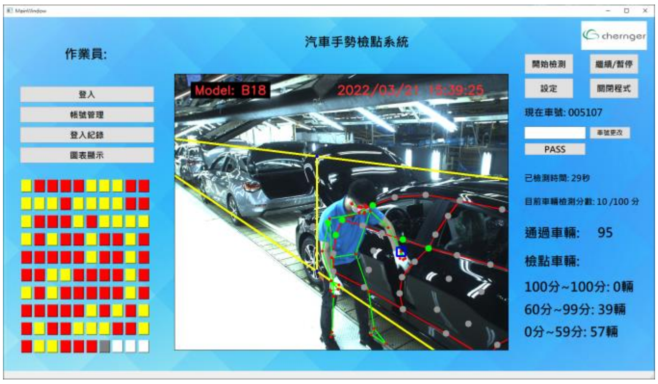

Figure 6: Main interface of our GUI.

Displaying chart button shows the chart with real-time statistics. The chart is implemented with multi-thread and will be updated every second, showing production line statistics of that day in Figure 8.

Setting button is to modify thresholds of three levels. By default, the lowest score level is defined as 0% to 59%. The middle one is defined as 60% to 99%. The highest score level is 100%.

Figure 7: Inspection snapshot.

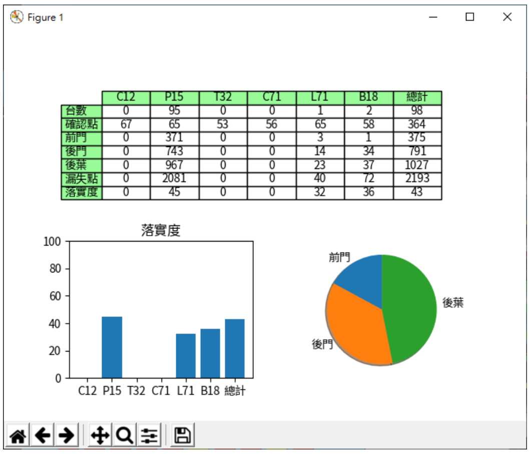

Figure 8: The real-time statistics chart.

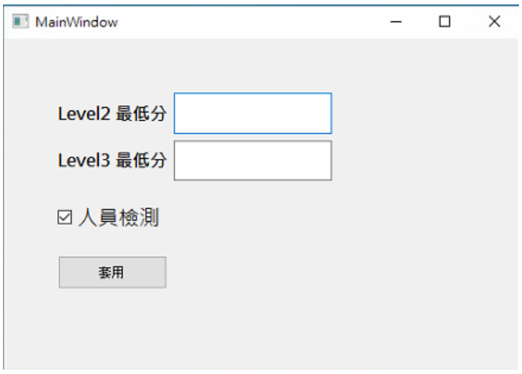

Figure 9: Setting window.

3.3 Car Detection Algorithm

After signal trigger, we first detect the wheel with object detection algorithm. We crop and use bottom-right quarter from image because car wheel will first appear in bottom-right quarter. We use YOLOv3 to detect each frame to check whether wheel appears in video. When YOLOv3 detects wheel in 10 continuous frames, our algorithm finds the wheel.

When we find the wheel, we do not detect it afterwards. Instead, we create a table of how wheel will move on the production line to make our system more robust. Because YOLOv3 may fail to detect wheel while the wheel is occluded. Since the speed of production line is constant, computing the wheel location with look-up table is stable and can reduce the computation cost.

Then, we map the inspection points on the car. Our algorithm supports 6 car models: P15, T32, L71, B18, C12, and C71. Inspection points corresponding to each model are defined by car factory. During inspection, our system will query the car number and its model from manufacturer's web service.

Knowing wheel location and the car model, we can map the inspection points on the car by relative position to wheels. With the location of wheels and auxiliary lines estimated from direction of production line, we can get a trapezoid with respect to lateral face of the car. With four points on trapezoid and their corresponding anchors, we can estimate the perspective transformation matrix. Then, we project our pre-defined anchors on that trapezoid with that matrix.

Figure 10: Inspection point definition.

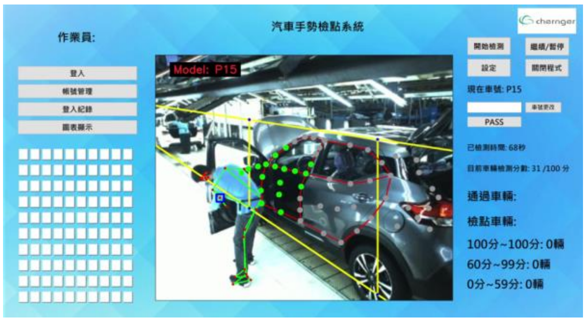

Figure 11: Inspection points on the car.

3.4 Operator Detection Algorithm

Simultaneously, we detect the operator with a human pose estimation algorithm, BlazePose. BlazePose predicts coordinates of 33 key points. The topology on human body is the superset of those used by BlazeFace, BlazePalm, and COCO (Common Object in COntext).

Operator inspects the car with his left hand. Thus, we track Key point 15, the left wrist, to judge his inspection. When operator's hand touches the inspection point, it will be considered detected.

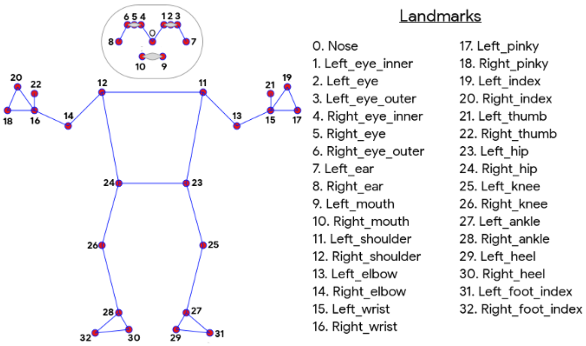

Figure 12: 33 key points topology of BlazePose.

After finishing inspection procedure of that car, we calculate the inspection score by the ratio of inspected points and total points.

3.5 Inspection Result Recording

We save the video of inspection process in a folder corresponding to that car, named with its car model and car number, for example 005107. A car is divided into 6 inspection parts: 1. Left front body + left front door; 2. Right front body + right front door; 3. Left rear door; 4. Right rear door; 5. Left rear body; 6. Right rear body. We record the inspection score of each part, and the total score. Therefore, they can understand which part is worse inspected, and care more about that part next time.

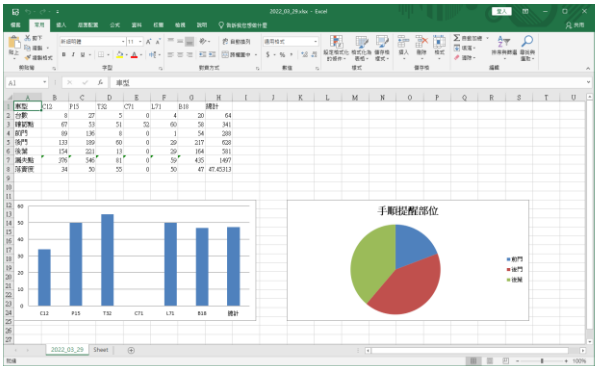
Figure 13: The record of inspection result.

3.6 Mirror the Algorithm to Right-Side

Our detection algorithm is based on the images collected from left camera. The procedure of right-side is similar to left-side. The only difference is location and direction of production line, which can be solved by image mirroring and some parameters tuning.

Thus, in right-side, we mirror images before processing, apply the same algorithm as left-side with some parameters tuned, and then mirror it again to display on monitor. Therefore, we can use the same algorithm with some minor difference for both sides.

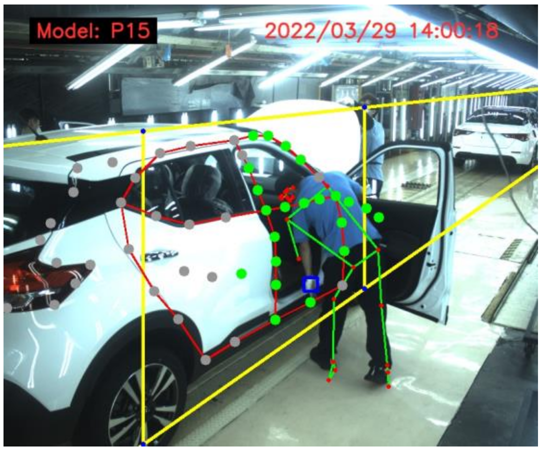

Figure 14: Detection algorithm run on right-side.

4. Results

4.1 Car Detection Algorithm

First, we tried to detect the car parts with object detection algorithm purely. However, it performed poorly in our experiments. Car door detection may fail due to variation of car models and colors. Wheel detection is stable, but cannot detect the occluded objects.

Compared with detecting car parts using object detection algorithm purely, our algorithm only detects the wheel at beginning, and then track it by look-up table, thus is more stable and more accurate, and can track the occluded wheels.


Figure 15 Left: Successful left front wheel detection; Right: Failed left front wheel detection.

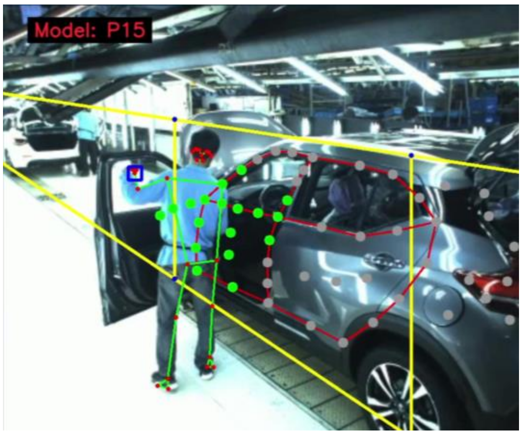

Figure 16: Our algorithm successfully tracks the left front wheel (blue dot on yellow line on conveyor belt) with look-up table, even during occlusion.

4.2 Human Pose Estimation Algorithm

BlazePose is a fast on-device pose estimation algorithm. Its rapid inference performs accurately most of the time.

However, it may fail while multiple people stand together in an image. Further, it incorrectly detects noise as a person (i.e. false alarm) sometimes.

The second problem is easy to solve since it only occurs while there are no people in the image, and the pose will be unusual. Thus, we can easily remove the wrong detection with a rule base defined by prior knowledge.

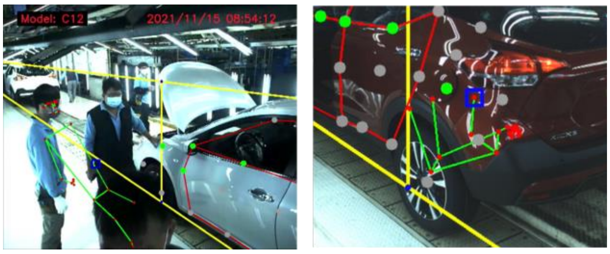

Figure 17: Examples of wrong detection.

Tested on the video captured from production line, precision and recall of detecting left hand achieves 95% and 71% respectively, and the inference speed is 0.06s per image. With this result, this algorithm can detect the operator accurately in real time.

4.3 Time Performance

Our system can achieve 10 FPS (Frames Per Second) in the inference stage on GeForce RTX™ 2080 Ti GPU (Graphic Processing Unit). With this inference speed, our system successfully satisfies the requirements on the production line.

5. Conclusion

We create a system installed on each side of production line in car factory to detect the inspection quality. We develop our detection algorithm based on left-side camera, and then apply it on right-side with image mirroring and some parameters tuning. This system collects the images from cameras on each side of production line, and analyzes them in real-time, confirms whether each inspection point is inspected.

Next, we build a GUI to let operators and administrators access this system easily, so they can supervise the production line situation. Besides, we record the inspection progress and results for each car, organize them into a statistics chart, make them understand the inspection quality and improve it.

Finally, we implement an algorithm to analyze each frame captured from production line. Our algorithm can deal with the occluding problem in production line, successfully track the wheels and map the inspection points on the car with perspective transformation. Then, we track the operators with pose estimation algorithm BlazePose.

With the location of inspection points and operator pose, we can judge the inspection quality and record the inspection result. Therefore, our system successfully achieves the purpose of quality management and assurance for car factory.

誌謝

This research was supported by the Ministry of Science and Technology of Taiwan, R.O.C., under Grants MOST 109-2221-E-002-158-MY2 and MOST 108-2221-E-002-140, and by Test Research, Jorgin Technologies, III, Chernger, Jeilin Technology, Otus Imaging, D8AI, PSL, and LVI.

References

[1] Y. H. Tsai, “Vehicle Door and Interior Defect Inspection and Quality Assurance,” Master Thesis, Department of Computer Science and Information Engineering, National Taiwan University, 2021.

[2] V. Bazarevsky, I. Grishchenko, K. Raveendran, T. Zhu, F. Zhang, and M. Grundmann, “BlazePose: On-device Real-time Body Pose Tracking,” https://arxiv.org/abs/2006.10204, 2020.

[3] J. Redmon and A. Farhadi, “YOLOv3: An Incremental Improvement,” https://arxiv.org/abs/1804.02767, 2018.

[4] A. Bochkovskiy, C. Y. Wang, and H. Y. M. Liao, “YOLOv4: Optimal Speed and Accuracy of Object Detection,” https://arxiv.org/abs/2004.10934, 2020.