Real-Time Traffic Monitoring and Status Detection with a Multi-vehicle Tracking System

Lu Wang(✉), Chan Tong Lam, K. L. Eddie Law, Benjamin Ng, Wei Ke, and Marcus Im

School of Applied Sciences, Macao Polytechnic Institute, Macao, China
{lu.wang,ctlam,eddielaw,bng,wke,marcusim}@ipm.edu.mo
https://www.ipm.edu.mo/esca/en/index.php

Abstract. With live street videos posted online, the Macao Government provides means to the general public to assess the latest road traffic conditions. After reviewing over these videos, a person may decide to change the travel route from the one he or she initially plans to take. To let road users make decisions better and faster, it would be desirable to design an automated software, being a component of an Intelligent Transport System, which offers proper suggestions to the users instantly upon analyzing all available live videos. In this paper, we propose to create a real-time road traffic condition estimation system. Its design is based on a combination of deep learning algorithms: the YOLOv5, DeepSORT, and the Non-Maximum Suppression algorithms. Putting together the YOLOv5 with our proposed two-stage NMS strategy, the improvement on the efficiency of object detection on live videos is noticeable. Our two-stage strategy removes the requirement to manually tune the NMS parameters continuously. With DeepSORT, we are able to track moving vehicles, and create motion trajectories, which we can use filtering strategy to assess the latest road traffic conditions. Since different lanes on a road may have different traffic situations, we separate the lanes based on angles and propose to use a lane status score independently for each lane. Through the experimental results, our system design could estimate the traffic status in real-time without requiring any manual parametric adjustments.

Keywords: Traffic transport systems · Multi-vehicle tracking · Road traffic status · Deep learning algorithms

1 Introduction

From an individual perspective, traffic congestion might affect a person's expected arrival time at a destination. It could lead to a consequential event such as missing a scheduled appointment, a gathering, or a musical show etc.

The work was supported by The Science and Technology Development Fund, Macao SAR (File no. 0001/2018/AFJ).

A. L. Martins et al. (Eds.): INTSYS 2021, LNICST 426, pp. 13–25, 2022.
https://doi.org/10.1007/978-3-030-97603-3_2

But collectively, from the financial perspective, the aggregate impact could affect the economy of a city [1], and it may be equivalent to virtually up to multi-billion dollars loss for a metropolitan city on estimation. Therefore, it is important to relieve or reduce the rate of occurrences of the traffic jams, especially it is beneficial to those who pay attentions to current road traffic conditions and scrutinize their own travel plans.

In order to subside the impacts on social, economic and touristic developments of a city, the Macao Government Transport Policy [1] attempting to optimize the road traffic conditions. The Macao Transport Bureau (DSAT) provides numerous live videos online [2], and people can go through them before making any travel plans. Browsing traffic video from multiple cameras is time consuming. It would be convenient if users could access traffic information digitally rather than on video.

In this paper, a novel vehicular monitoring system is proposed to identify the road traffic status in real-time. Our goal is to allow users to use our proposed system to dynamically and instantly determine if there is a traffic congestion at an intersection. With the rapid development of artificial intelligent technologies [3] and using the live video streams as inputs, we propose a road traffic monitoring system by applying deep learning algorithms on these input videos, we may be able to identify the road traffic status instantly, and generate helpful recommendations to users, if being asked. This system hence removes the needs of going through each online video sequentially by a human being in order to make the decision on the travel path afterwards.

Until today, there were a few traffic status detection systems proposed for Macao [4–6]. In [4], an economical real-time traffic congestion detection system was proposed. Its design was based on the low frame-rate online video provided by the Macao Government, but it used Haar-like features to detect vehicles. Then in [5], a real-time traffic analysis system was proposed to perform the correlation analysis, using the online traffic maps and the real-time information of available parking spaces from a government website. In another paper [6], an improved system was designed with neural network-based YOLOv3 [7] for vehicular detection and an mIOU (mean Intersection-Over-Union) method was proposed to estimate the current traffic state, but input images were from the low frame rate web cameras.

Recently, many street monitoring web cameras in Macao [1] have been upgraded from one frame every five seconds to about thirty frames per second. The frame resolutions are also improved. In other words, the input sources have been enhanced from still images to video streams. This permits us to examine more sophisticated deep learning algorithms regarding vehicular tracking [8], and traffic condition identifications through video analysis [9]. Indeed, deep learning-based algorithms are popular and essential for object detections. However, the resources required to track a large number of cameras simultaneously can be costly.

Frontier research works on both object detection [3,10] and tracking are crucial in making the multi-object tracking (MOT) operations successful. This will be our

research focus in this paper to carrying out MOT on road traffic through videos in order to determine whether the roads being monitored are congested or not.

2 System Architecture

Management of road traffic is always complicated. Drivers may react differently under the same traffic congestion scenarios. Some may try to switch lanes often, some may just stay put unchanged, some may decide to change routes completely, or some may go back where they start the journeys. In this section, we focus on designing a real-time multiple vehicles tracking system on streets with web cameras. The system operational model is shown in Fig. 1 below. In the system, the videos from web cameras are and will be cached in memory. The vehicle detection, identification and path tracking will be carried out in our proposed system. In fact, upon using the online videos from Macao Government, we can easily achieve one of our primary design goals, i.e., a low cost system in nature.

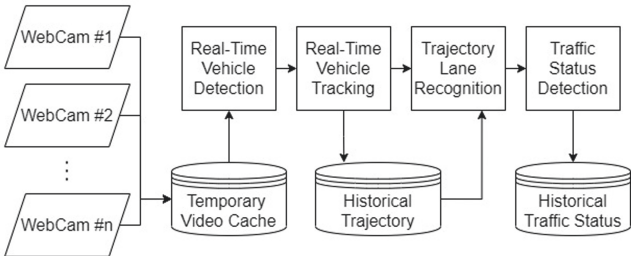

Fig. 1. System model and components.

The traffic monitoring system consists of two components [6]. One is to identify and track the motions of the vehicles. Another is to interpret the number of lanes, and their respective traffic status, such as the moving directions and status of traffic congestion.

Multi-lane roads are the main scenario we consider. A fixed lane mask is an intuitive solution that allows the system to consider one lane at a time. However, this requires marking each lane and cannot be adapted to lane changes. We therefore use recent historical vehicle trajectories to identify lanes, which can be adapted to different situations without the need for manual labelling.

Regarding the object detection mechanisms, the accuracy in general is not high if there are limited computational resources. The small-scale computation models may restrict the efficiency and degrade the overall system performance. The Non-Maximum Suppression (NMS) is the standard choice on finding the bounding boxes for object detections. However, the NMS parameters usually need to be manually adjusted for the actual situation. These manual adjustment requirements are due to different factors (for example, the lighting effect, etc.), and make NMS less effective on overall system performance.

The YOLOv5 (You Only Look Once) [11] is a PyTorch-based implementation of a set of pre-trained neural network models for fast object detections. In our proposed design, we integrate a YOLOv5 model with our recommended two-stage NMS strategy for rapid recognizing moving vehicles. The DeepSORT [12] is executed for vehicular trajectory tracking, and then we apply our filtering strategy in Sect. 3 to obtain the high-quality trajectories. The full algorithm is described in Algorithm 1.

Algorithm 1: Multi-vehicle Detection and Tracking System

Require: Video data $\mathcal{V}$
Loose NMS parameters p_{nms}
Min detection box size L_{box}
Min confidence threshold C_{min}
DeepSORT algorithm $F_{DeepSORT}$
Trajectory length threshold L_{traj}
Trajectory detected frame ratio threshold r
Trajectory jump threshold γ

while True **do**
  Update lane separate threshold set t_i from history
  Update traffic time set t_{min}, t_{max} for every lane from history
  Get next video segment $\mathcal{V}$ from cache
  Detection boxes $\mathcal{D} = \texttt{YOLOv5}(\mathcal{V})$
  // Two-stage NMS strategy
  $\mathcal{D} = NMS(\mathcal{D}, p_{nms})$ with loose parameters
  Filter detection, $\mathcal{D} = category\_filter(\mathcal{D})$
  $\mathcal{D} = box\_size\_filter(\mathcal{D}, L_{box})$
  $\mathcal{D} = confidence\_filter(\mathcal{D}, C_{min})$
  Trajectory $\mathcal{T} = F_{DeepSORT}(\mathcal{V}, \mathcal{D})$
  // Trajectory filter strategy
  Filter the trajectory $\mathcal{T} = traj\_len\_filter(\mathcal{T}, L_{traj})$
  $\mathcal{T} = frame\_ratio\_filter(\mathcal{T}, r)$
  $\mathcal{T} = jump\_traj\_filter(\mathcal{T}, \gamma)$
  For every lane, calculate the median passing times $\{t_{lane}\}$
  For every lane, current traffic score $\mathcal{S}_{lane} = (t_{lane} - t_{min})/(t_{max} - t_{min})$
end while

Through successful multi-vehicle tracking operations, we can differentiate different lanes on roads with the moving directions of the vehicles. Each lane must be handled separately. We should define indicators with less erroneous results caused by the camera shakes. The indicators being used are the interval transit time and the number of vehicles passing through in an interval. Since the online videos from web cameras are not annotated, we have to manually

annotate some videos in order to evaluate the performance of our system. The evaluation result shows that our system has good performance.

The operation details of our proposed system are outlined in Algorithm 1. And in summary, the flow of different operations are as followed. Since some cameras update video frames every 5 s, data from multiple cameras are first temporarily cached. Then the multi-GPU vehicle detection service processes the cached video in real-time. The YOLOv5l model we use can process up to 177 frames per second per GPU. With our two-stage NMS strategy, we obtain acceptable detection results without requiring manual optimization of the NMS parameters from multiple cameras. Then the DeepSORT algorithm is used to track the trajectory of all vehicles in real time, and its appearance model is also provided by a multi-GPU service to support multiple cameras. To avoid low quality trajectories affecting traffic status detection, we filter using our trajectory quality assessment score. By counting recent historical trajectories, we identify different lanes to evaluate traffic status individually. Our traffic status scores are determined by comparing historical traffic speeds with short-term average travel speeds.

3 Multi-vehicle Tracking Mechanisms

3.1 Real-Time Vehicle Detection with Multiple Cameras

The YOLO is a family of neural network models for object detections. It is well known due to its speed and accuracy [11]. Shortly after the release of YOLOv4 [10], the YOLOv5 [3] implementation is based on the PyTorch framework. Its pre-trained models are easy to use, and are friendly to different engineering applications.

Among the YOLO sub-models, the YOLOv5s model can reach 500 fps on V100 GPU [3]. This allows us to perform real-time object detection on multiple videos on a single GPU. Considering the speed and accuracy, we select the YOLOv5l model, and use a pre-trained model due to the fact that the online videos from Macao Government are not labeled.

In general, object detection model gives a higher number of detected object than the actual number. This leads to select the Non-Maximum Suppression (NMS) algorithm which can find the best detection box among all possible choices.

3.2 Two-Stage NMS Strategy

As aforementioned, the NMS algorithm parameters usually need to be optimised manually. In a single fixed dataset, it is inexpensive to adjust the parameters. But with multiple cameras in the real world, different scenes require different NMS parameters, and the optimal parameters also vary with the lighting. Therefore it is costly to manually adjust the NMS parameters in our system.

Some algorithms try to adjust the parameters automatically. The adaptive NMS [13] use a neural network to estimate the density scores, which was used to

adjust the threshold dynamically. This performed well in high density tracking. The disadvantage of this approach was that new models needed to be trained, which increased the computational power requirement for real-time detection and thus increasing cost.

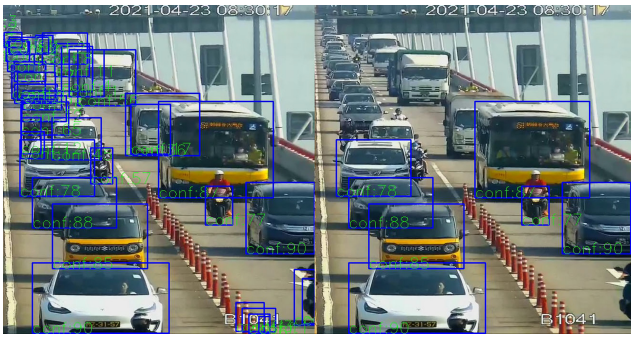

Fig. 2. Issues due to different NMS parameters, Left: false detections with loose threshold; Right: missed detection with tighter threshold.

In this section, we propose a two-stage NMS strategy to solve the parametric setting issue. This strategy is to run NMS with loosely set parameters, and then executes a heuristic filtering.

Usually, as shown in Fig. 2, NMS parameters can lead to missed detections if they are too tight and false detections if they are too loose. In fact, a missed detection can not be fixed because there is no detection information. On the other hand, false detections still have opportunities to be removed in future processing.

Through the experiments, we have obtained some falsely detected objects. Some of the falsely detection boxes are smaller in sizes. Falsely detected objects are usually the results of different classifications, and they appear discontinuously in continuous videos. Therefore, we first process the detection box using very loose NMS parameters and then filter it with the following strategy:

- Filter the detection boxes that are not belong to a vehicle categories,
- Filter according a detection box size threshold, L_{min},
- Filter according a confidence threshold, C_{min}.

3.3 Trajectory Filtering Strategy

We estimate the traffic status scores by vehicle trajectories. The DeepSORT [14] is an extension of the Simple Online and Real-time Tracking algorithm (SORT) [15]. Central to the effectiveness of the DeepSORT algorithm are the appearance and movement models. We select the DeepSORT because of its effectiveness and quickness.

In DeepSORT, We use kalman filter based motion model and pre-trained appearance model from the MOT16 challenge dataset [16]. Apart from estimating the traffic status, the dynamically established trajectories are also used to estimate and create lane information at low budget. Because we use smaller models, we are able to achieve acceptable accuracy rates faster than more complex or larger models. As shown in Fig. 3, some obtained trajectories from live videos are not suitable for traffic status estimations. The more complex scenario in Fig. 4 is used here, rather than the simpler one in Fig. 2.

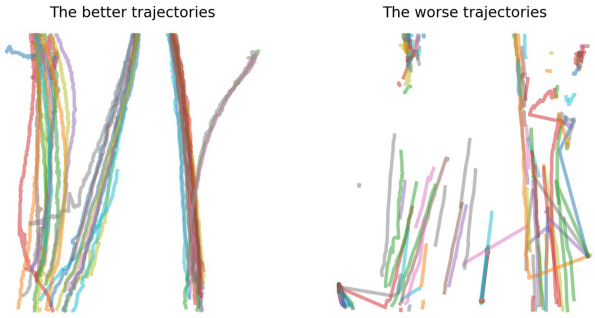

Fig. 3. Examples of created trajectories: each colored line is a trajectory.

Poorly created trajectories can lead to incorrect speed estimates. Therefore, we need to select trajectories with higher confidence. We define trajectory a set of points $\mathcal{T} = \{p_i\}, p_i = (i, x, y)$, where p_i is a point on the track detected from a video frame, the frame number is i, and the position of the point on the frame is (x, y). For a trajectory, i_0 is the first frame detected and i_n is the last frame detected. Then we add the following rules to determine whether a created trajectory can be accepted with high confidence:

- Trajectory length, $l = i_n - i_0$,
- Detected frame ratio, $r = \frac{|\mathcal{T}|}{i_n - i_0}$,
- A jumping trajectory, if exists, $|(p_{n-1}, p_n)|_2 > \gamma$, where γ is the distance threshold.

Through observations, the quality of the tracks at the edges is usually low, and those parts of the tracks near to the edges of frames are removed.

4 Traffic Status Estimation

4.1 Lane Separation

Different lanes in different directions on the same road may have different traffic status. Before calculating the traffic status, it is necessary to distinguish the traveling directions of traffic and calculate them separately. Although it is possible to

identify lanes by simply manually marking with lane masks, this is in fact more costly when there are numerous cameras. Moreover, fixed lane masks do not fit lanes with sudden traffic controls, e.g., lanes with stopped traffic temporarily.

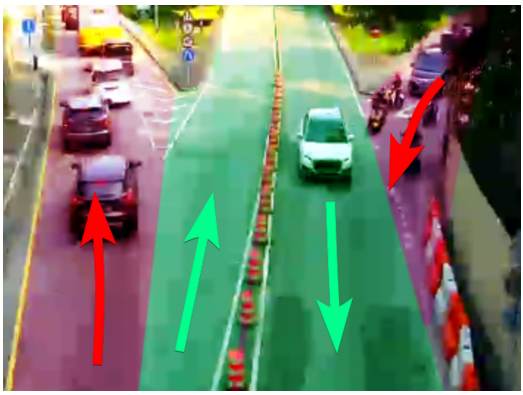

Fig. 4. Different traffic status for different lanes in the same directions: arrow indicates the travel direction, red indicates traffic congestion. (Color figure online)

Furthermore, the placements of the cameras are outside our control. If the light incidence angle of an camera is low, it may increases the degree of perspective of the camera. This allows parallel lanes of traffic to appear at different angles on the screen. We can use the track angle to determine different lanes. This process is automatic and therefore adapts automatically after a road change. By reducing the trajectory to a vector representation, $\langle p_0, p_n \rangle$, we can calculate its incidence angle, θ, with a unit vector, $\langle 0, 1 \rangle$.

Upon counting the trajectory angles over time, we can derive its distribution. We assume that the angle of travel for each lane always obeys a distribution, then the trajectory angles of N lanes should follow a set of N distributions. Based on this assumption, an angle-based lane separation mechanism is feasible. The minimal value of a trajectory angle distribution curve is used as the threshold for separating lanes. However, in practice, due to the limited number of lanes counted, the distribution function can be noisy and the statistics should be filtered before using the smallest value as the lane separation threshold (Fig. 5).

4.2 Traffic Status Score

We can quantify the traffic conditions of a lane by measuring the travel time of the trajectories, the travel time defined by the number of frames traversed. Historical data is being analyzed to determine the time required to go through a given lane on a video regularly. We use the previous day's lane tracking results to count the vehicle travel times, and take the median of the shortest 10% of the

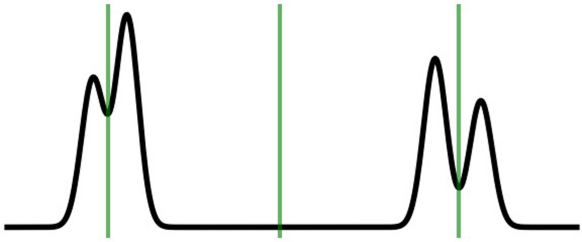

Fig. 5. Lane separation based on angle distribution: four lanes observed with the green line partitioning. (Color figure online)

travel times as the smooth travel times called t_{min}. Upon defining congested or slowly moving traffic, we define $t_{max} = \alpha * t_{min}$, where $\alpha = 1.6$ is set empirically.

We use the median travel time of all high-quality trajectories over a period of time t to calculate the traffic status score, which is defined as $\mathcal{S} = (t - t_{min})/(t_{max} - t_{min})$. If $\mathcal{S} < 0$, the traffic is clear or smooth. If $\mathcal{S} \geq 1$, there is traffic congestion.

5 Experimental Results

5.1 Vehicle Detection and Tracking

The inference speed is important to our system. Object detection models are often the bottleneck in MOT systems, because of the large computational cost associated with deep neural networks [17]. The appearance model is usually faster because it is smaller, which means that more images can be processed per batch using less inference time on the same device.

The deep neural network in our system is implemented by PyTorch, we use NVIDIA RTX2080Ti GPU for faster model inference. To find the best batch size for YOLOv5 on our device, we selected 9 different available batch sizes. The resolution of the selected video is 544×640 pixels, with 2500 frames. The results were shown in Fig. 6.

In contrast, YOLO was slower than expected, though we picked the faster YOLOv5l model. YOLOv5 achieved 196 fps (5.09 ms/frame) on our GPU. The object detection models usually generated many detection frames, but a vast majority of them were invalid. In order to reduce the usage of GPU transmission bandwidth, we used GPU for NMS calculations. Each GPU could support up to about 6 cameras for real-time vehicle detection, assuming that one camera generated 30 frames per second.

With GPU NMS, the inference speed was 177 fps (5.64 ms/frame). Regarding the NMS threshold issue, we used a two-stage NMS strategy. In the cases that it was impossible to pre-optimize the thresholds, our two-stage NMS strategy yielded acceptable results. It could filter a proportion of false detections as

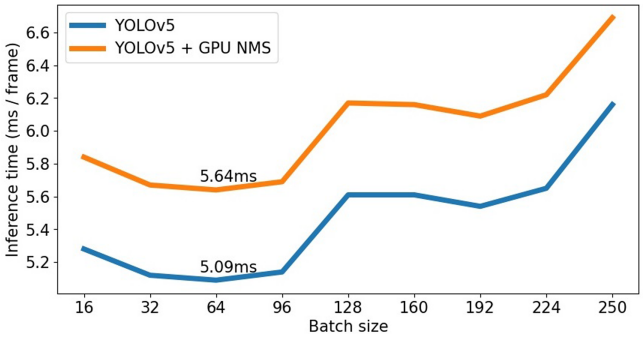

Fig. 6. YOLOv5 inference time with different batch sizes.

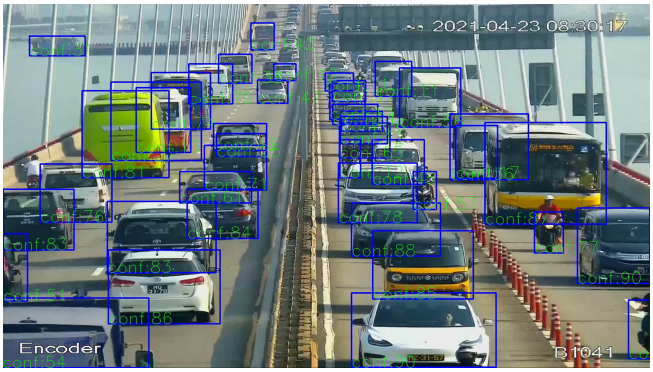

Fig. 7. Two-stage NMS result. Same scenario in Fig. 2

opposed to using a loose threshold only. The effect of the two-stage NMS strategy was shown in Fig. 7.

Through the DeepSORT algorithm, we could obtain many vehicle trajectories. However, not all trajectories were suitable for evaluating traffic status due to the fact that some trajectories were too short or broken. The ID switch, which was considered wrong in the field of target tracking, and had no influence on the detection of traffic status. Figure 8 showed the filtered trajectory, and each colored line represented a trajectory. The filtered trajectory on the right side was of higher quality than that on the left side. Our traffic status scores were calculated based on average travel time, and the high quality trajectory helped to obtain a more accurate estimate of traffic status.

5.2 Traffic Status Detection

Different lanes could have different traffic status. We used an angle-based approach to separate the lanes. In Fig. 9, an one-hour historical trajectory was used for lane separation. The blue bars indicated the number of occurrences of

Fig. 8. Vehicle trajectory. Left: raw data; Right: filtered. Same scenario in Fig. 4

different angles, the orange was the filtered smooth distribution function, and the green line showed the solved minima. These minima were the threshold values used for lane separations. In Fig. 10, the results of traffic lane separations in a camera were outlined.

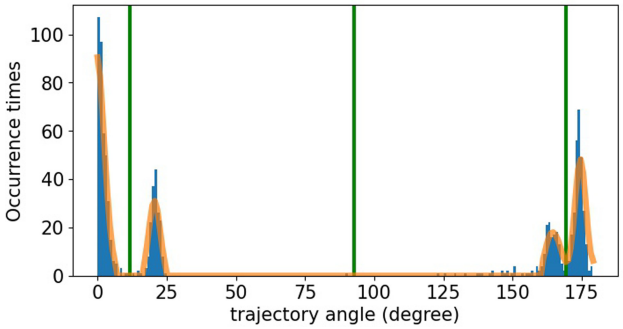

Fig. 9. Lanes by trajectory angle. Same scenario in Fig. 4

Our method allows for more flexible lane estimation than using the fixed lane. It can be deployed directly without additional annotation upon applying to a large number of cameras. However, it has a drawback that it takes some time to collect data for initialization and may not be put into use immediately.

After dividing the lanes, we can use traffic status score to estimate the traffic status. For the scenario in Fig. 4, we estimate the data for one lane for a day. The traffic status score of one lane from one camera was shown in Fig. 11. We randomly selected 500 videos to evaluate the accuracy of our method, these video clips have been manually tagged, half of these videos were traffic congestions. Our accuracy was 96.6%.

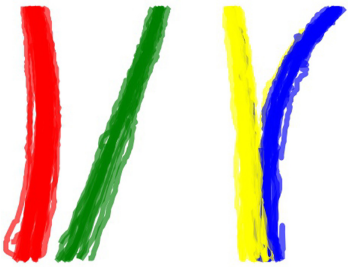

Fig. 10. Lane separation: a colored line represents a lane. Same scenario in Fig. 4 (Color figure online)

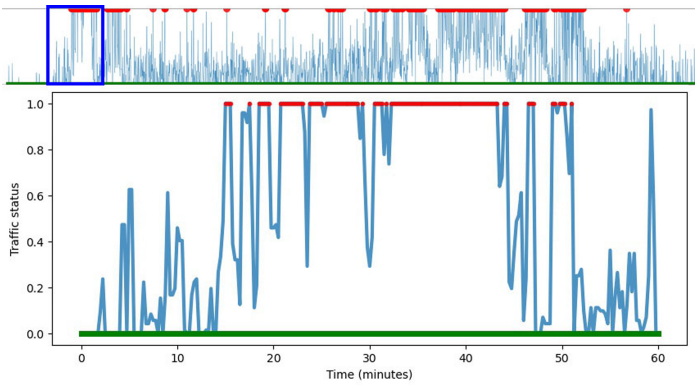

Fig. 11. Traffic status score of a lane, Top: a daily status score sheet; bottom: an hour score sheet (score 1 indicates traffic congestion).

6 Conclusion

Based on a set of deep learning algorithms, the YOLOv5, a 2-stage NMS, and the DeepSORT algorithms, we have proposed the design of a real-time multi-vehicle detection and tracking system. The system works with those online streaming videos provided by Macao Traffic Bureau. Through testing and observations, the proposed multi-vehicle tracking system works fast and satisfactorily. On average, it works harmoniously with 3 video cameras per GPU core. In our proposed design, the YOLOv5 model and a 2-stage NMS strategy are integrated to provide fast and acceptable detection results without the needs to run any manual NMS parametric adjustments. And the simple and fast DeepSORT algorithm is adopted for vehicular tracking. Through our proposed trajectory filtering strategy, high-quality trajectories can be generated for the road traffic status estimations. By calculating the travel times of vehicles per unit time, we obtain the traffic status scores for all lanes in videos in real-time. This information will be useful in future for finding recommended paths between any two locations in city. All in all, our proposed system works satisfactorily, and it can expands easily to handle large number of input videos.

References

1. Macao Transport Bureau: General traffic and land transport policy of MACAU (2011). http://www.dsat.gov.mo/ptt/pt/index.html. Accessed 30 May 2021
2. Macao Transport Bureau: Real-time road traffic videos (2021). http://www.dsat.gov.mo/dsat/realtime.aspx. Accessed 30 May 2021
3. Jocher, G., et al.: ultralytics/yolov5: v5.0 - YOLOv5-P6 1280 models, AWS, Supervisely and YouTube integrations (2021). https://doi.org/10.5281/zenodo.4679653
4. Lam, C., Gao, H., Ng, B.: A real-time traffic congestion detection system using on-line images. In: 2017 IEEE 17th International Conference on Communication Technology (ICCT), pp. 1548–1552 (2017)
5. Lam, C., Ng, B., Pun, I.: Analysis of traffic status using on-line traffic maps and real-time information of parking spaces. In: 2018 IEEE 18th International Conference on Communication Technology (ICCT), pp. 483–487 (2018)
6. Lam, C., Ng, B., Wang Chan, C.: Real-time traffic status detection from on-line images using generic object detection system with deep learning. In: 2019 IEEE 19th International Conference on Communication Technology (ICCT), pp. 1506–1510 (2019)
7. Redmon, J., Farhadi, A.: Yolov3: an incremental improvement. ArXiv abs/1804.02767 (2018)
8. Bui, K.H.N., Yi, H., Cho, J.: A multi-class multi-movement vehicle counting framework for traffic analysis in complex areas using CCTV systems. Energies (2020)
9. Datondji, S.R.E., Dupuis, Y., Subirats, P., Vasseur, P.: A survey of vision-based traffic monitoring of road intersections. IEEE Trans. Intell. Transp. Syst. **17**, 2681–2698 (2016)
10. Bochkovskiy, A., Wang, C.Y., Liao, H.: Yolov4: optimal speed and accuracy of object detection. ArXiv abs/2004.10934 (2020)
11. Redmon, J., Divvala, S., Girshick, R.B., Farhadi, A.: You only look once: unified, real-time object detection. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 779–788 (2016)
12. Wojke, N., Bewley, A., Paulus, D.: Simple online and realtime tracking with a deep association metric. In: 2017 IEEE International Conference on Image Processing (ICIP), pp. 3645–3649. IEEE (2017). https://doi.org/10.1109/ICIP.2017.8296962
13. Liu, S., Huang, D., Wang, Y.: Adaptive NMS: refining pedestrian detection in a crowd. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 6452–6461 (2019)
14. Wojke, N., Bewley, A., Paulus, D.: Simple online and real-time tracking with a deep association metric. In: 2017 IEEE International Conference on Image Processing (ICIP), pp. 3645–3649 (2017)
15. Bewley, A., Ge, Z., Ott, L., Ramos, F., Upcroft, B.: Simple online and real-time tracking. In: 2016 IEEE International Conference on Image Processing (ICIP), pp. 3464–3468 (2016). https://doi.org/10.1109/ICIP.2016.7533003
16. Milan, A., Leal-Taixé, L., Reid, I., Roth, S., Schindler, K.: Mot16: a benchmark for multi-object tracking. ArXiv abs/1603.00831 (2016)
17. Zou, Z., Shi, Z., Guo, Y., Ye, J.: Object detection in 20 years: a survey. ArXiv abs/1905.05055 (2019)