A machine learning pipeline for extracting decision-support features from traffic scenes

Vitor A. Fraga[1], Lincoln V. Schreiber[1], Rafael Kunst[1], Jorge Luis V. Barbosa[1] and Gabriel de O. Ramos[1]

[1]*Graduate Program in Applied Computing, Universidade do Vale do Rio dos Sinos, São Leopoldo, Brazil*

Abstract

Traffic systems play a key role in modern society. However, these systems are increasingly suffering from problems, such as congestions. A well-known way to efficiently reduce this kind of problem is to perform traffic light control intelligently through reinforcement learning (RL) algorithms. In this context, extracting relevant features from the traffic environment to support decision-making becomes a central concern. Examples of such features include vehicle counting on each queue, identification of vehicles' origins and destinations, among others. Recently, the advent of deep learning has paved to way to efficient methods for extracting some of the aforementioned features. However, the problem of identifying vehicles and their origins and destinations within an intersection has not been fully addressed in the literature, even though such information has shown to play a role in RL-based traffic signal control. Building against this background, in this work we propose a deep learning pipeline for extracting relevant features from intersections based on traffic scenes. Our pipeline comprises three main steps: (i) a YOLO-based object detector fine-tuned using the UAVDT dataset, (ii) a tracking algorithm to keep track of vehicles along their trajectories, and (iii) an origin-destination identification algorithm. Using this pipeline, it is possible to identify vehicles as well as their origins and destinations within a given intersection. In order to assess our pipeline, we evaluated each of its modules separately as well as the pipeline as a whole. The object detector model obtained 98.2% recall and 79.5% accuracy. The tracking algorithm obtained a MOTA of 72.6% and a MOTP of 74.4%. Finally, the complete pipeline obtained an error of 7.53% in terms of origin and destination counts.

Keywords

object detection, object tracking, traffic scenes, intersections, deep learning

1. Introduction

Traffic congestion represents a major challenge in urban areas [1]. Intelligent Transportation Systems (ITS) emerged as a way to more efficiently handle traffic issues by means of information and communication technologies, as leveraged by data-driven artificial intelligence approaches [2]. In this context, obtaining data becomes a central concern in order to enable a more intelligent management of the traffic environment [3]. Such data is of great value for several purposes, namely for estimating the flow of vehicles, monitoring them, controlling traffic lights, among others.

Recently, reinforcement learning (RL) algorithms have shown potential for traffic lights

ATT'22: Workshop Agents in Traffic and Transportation, July 25, 2022, Vienna, Austria

vitor.augusto.fraga@gmail.com (V. A. Fraga); lincolnschreiber@gmail.com (L. V. Schreiber); rafaelkunst@unisinos.br (R. Kunst); jbarbosa@unisinos.br (J. L. V. Barbosa); gdoramos@unisinos.br (G. d. O. Ramos)

control [4]. Roughly, considering that traffic light control can be seen as a sequential decision-making problem, RL can be used to learn a behavior able to indicate which action should be taken for any given traffic condition. In this context, having access to relevant information describing traffic conditions is essential to enable RL adoption [5]. One such information refers to the origin and destination of each vehicle on the scene, which enables traffic light controllers to consider the traffic dynamics on the decision-making process.

The extraction of features from traffic scenes can be performed using different approaches. Recurrent approaches include [6]: lane sensors to count vehicles passing through a given location, connected GPS devices to keep track of vehicles' routes, laser radars to help reducing collisions, and so on. However, these approaches typically rely on expensive, intrusive infrastructure. On the other hand, recent advances in digital image processing leveraged by deep learning techniques have enabled the adoption of less invasive, more easily deployable approaches. A particularly interesting direction here refers to the use of intersection cameras coupled with computer vision techniques to automatically extract relevant traffic information [7].

The problem of extracting information from traffic scenes has been increasingly investigated in the literature [7, 8, 9, 10, 11, 12, 13]. Typically, existing works consider traffic scenes (e.g., images, videos) and extract relevant information using classification or object detection models, such as *YOLO* [14], *SSD* [15], *CornerNet* [16, 17], as well as two-stage approaches like *R-CNN* [18], *Fast R-CNN* [19], and *Faster R-CNN* [20]. However, these works fail to obtain more complex traffic information, like the origin and destination of the vehicles, which has shown essential for traffic light control [4, 5, 21]. More recently, other works [22, 23] took a step forward by also proposing tracking methods able to keep track of vehicles' trajectories along the traffic scenes. However, to the best of our knowledge, the identification of origins and destinations has been neglected in the the literature, which has hindered the applicability of RL-based traffic light control in real world.

Motivated by the need to obtain relevant information for supporting RL-based traffic lights control, in this work we introduce a complete pipeline for identifying and counting vehicles' origins and destinations from traffic scenes. Our pipeline includes three main steps. The first step employs a YOLOv4 network [14] for detecting vehicles in a traffic scene. Our model is pre-trained on the COCO dataset [24] and then fine-tuned for traffic scenes using aerial images from intersections, as available in the UAVDT dataset [25]. The second step of our pipeline consists in a tracking algorithm, which identifies vehicles along frames in order to recognize their trajectories throughout the intersection. Finally, as the last step, our pipeline identifies the origin and destination of each vehicle by analyzing the lane from which it has departed and the lane in which it arrived.

The proposed pipeline was assessed using previously unseen traffic scenes. The object detector step obtained a recall of 98.2% and accuracy of 79.5%. The tracking step yielded a multiple object tracker accuracy (MOTA) of 72.6% and precision (MOTP) of 74.4%. The complete pipeline obtained an average error as lower as 7.53% in terms of origins and destination counts. Hence, putting all together, our pipeline has shown to recognize vehicles, their trajectories and their origin and destinations, thus being able to properly summarize the origin-destination table for a given traffic intersection.

The main contributions of this work can be enumerated as follows:

- A customized YOLOv4 neural network [14] fine-tuned with the UAVDT dataset [25] for detecting vehicles at intersections. Our model also features larger input images, with a shape of 832 × 832.
- A vehicle tracking algorithm to identify the trajectory of a vehicle throughout an intersection by comparing its position along different frames.
- An algorithm for analyzing the vehicles' trajectories and extracting the number of vehicles belonging to each origin and destination lanes in the intersection.

The rest of this paper is organized as follows. Section 2 presents a brief overview of related work. Section 3 introduces the pipeline presented in this paper. Section 4 presents an empirical evaluation of our pipeline. Finally, Section 5 brings the concluding remarks.

2. Related Work

In this section, we briefly review the literature from two perspectives. As for the first perspective, we consider approaches related to multi-object tracking, which comprise the first and second stages of our pipeline. Considering the second perspective, we review works related to traffic flow control from video inputs.

In multi-object detection and object tracking. The detector aims at extracting information about objects of interest, such as their location in the scene. Currently, most methods that fall within this category are based on Convolution Neural Networks (CNNs). Roughly, CNN-based approaches can be divided into *one-stage*, such as *YOLO* [14], *SSD* [15], and *CornerNet* [16, 17], and *two-stage*, such as *R-CNN* [18], *Fast R-CNN* [19], and *Faster R-CNN* [20]. On the other hand, the tracker aims at identifying the same object throughout different scenes. To this end, the tracker typically relies on detections output by a detector along multiple frames. Object tracking can then be used for counting unique objects in a video, for example. Some of the approaches that can be used to perform tracking include IOU Tracker [26], SORT [27], and DEEP SORT [28].

When it comes to vehicle counting, however, it is necessary to go beyond. Once vehicles are detected and tracked throughout a traffic scene, specific methods are necessary to count the number of vehicles, to estimate the traffic flow, or even to identify the origin and destination of each vehicle in the scene. For this type of problem, [22] and [29] proposed the use of virtual lines, which enable trajectories to be associated with given origins and destinations.

In spite of the promising results achieved in the literature, to the best of our knowledge no previous work proposed a complete pipeline for extracting and analyzing the origins and destinations of vehicles within traffic intersections using a combination of deep learning approaches. In particular, none of them jointly: consider the use of aerial images of intersections, recognize vehicles, identify their paths, and quantify the number of vehicles for each combination of origin-destination pairs. The lack of solutions comprising all these aspects motivates the pipeline we propose in this work.

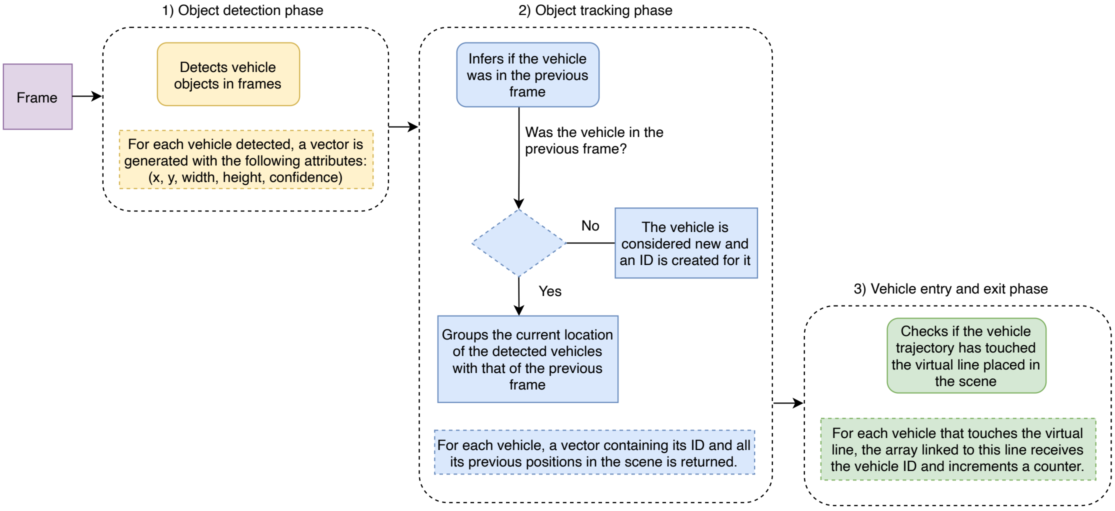

1) Object detection phase
Frame
Detects vehicle objects in frames
For each vehicle detected, a vector is generated with the following attributes: (x, y, width, height, confidence)
2) Object tracking phase
Infers if the vehicle was in the previous frame
Was the vehicle in the previous frame?
No
The vehicle is considered new and an ID is created for it
Yes
Groups the current location of the detected vehicles with that of the previous frame
For each vehicle, a vector containing its ID and all its previous positions in the scene is returned.
3) Vehicle entry and exit phase
Checks if the vehicle trajectory has touched the virtual line placed in the scene
For each vehicle that touches the virtual line, the array linked to this line receives the vehicle ID and increments a counter.

Figure 1: Overview of the proposed pipeline, comprising the vehicle detection, vehicle tracking, and origin-destination identification steps.

3. Proposed Pipeline

This paper introduces a deep learning pipeline for extracting the origin and destination of vehicles throughout a traffic intersection using aerial images. The underlying idea of our pipeline is that, by identifying a vehicle and keeping track of its trajectory, it is possible to identify the lanes through which it has entered and exited the intersection. That information can then be used to count the number of vehicles on each origin and destination within the intersection. In order to accomplish that task, our pipeline performs three tasks: (i) vehicle detection, (ii) vehicle tracking, and (iii) origin-destination identification. The overview of our pipeline is presented in Fig. 1.

Roughly speaking, our pipeline starts with a frame being received as input. The first phase is object detection one (Section 3.2), which detects the vehicles on the frame using a fine-tuned YOLOv4 network [14]. The network is fine-tuning using the UAVDT dataset, as described in Section 3.1. Once vehicles are detected, it is time for the object tracking phase (Section 3.3), where the positions of a vehicle within subsequent frames are used to form its trajectory. This step is performed using the tracking method proposed in [26]. Finally, the origins and destinations identification phase (Section 3.3) is responsible for detecting the entrance and exit points of each trajectory in order to allow summarizing the number of vehicles on each origin-destination pair within the intersection.

3.1. Dataset

In order to train and validate our pipeline, we first had to search a dataset featuring traffic footage with aerial angulation and appropriately annotated vehicles. Meeting these requirements provide sufficient conditions to perform object detection and tracking tasks using CNN architectures.

We consider the UAVDT dataset [25]. The UAVDT dataset features traffic scenes with diverse

Table 1
Subset of UAVDT dataset scenes used in our work for training and validation.

Name	Frames	View	Altitude	Car	Truck	Bus	Total vehicles	Unique vehicles	Type
M0101	407	Front+Side	Medium	5156	188	70	5414	20	Train
M0210	583	Bird	Medium	4725	1796	0	6521	33	Train
M0402	410	Side	Medium	10210	0	0	10210	45	Train
M0601	372	Front+Side	High	9908	150	365	10423	51	Train
M0606	1374	Front+Side	Medium	17071	224	459	17754	96	Train
M0701	1308	Bird	High	90563	2472	1008	94043	182	Train
M0702	777	Bird	Medium	42042	386	3699	46127	94	Train
M0703	683	Bird	Low	15235	643	628	16506	89	Train
M0801	298	Bird	Low	3832	108	207	4147	24	Train
M1201	1197	Front+Side	Medium	23080	4188	0	27268	59	Train
M0403	514	Front+Side	Medium	31403	0	0	31403	99	Validation
M0603	2035	Bird	Medium	41478	1748	1500	44726	71	Validation

heights (i.e., low, medium, high) and angulations (i.e., front, side, and bird). This dataset includes 50 different traffic scenes, totaling over 80,000 frames. However, in order to ensure that the whole intersection is captured in the scenes, we selected only a subset of scenes that are high enough to enable all incoming and outcoming lanes to be observed. This includes scenes with front or side views with at least medium altitude as well as bird view at any altitude. The final list of scenes is presented in Table 1. These scenes are essential to train the vehicle detector method, which we describe in detail in the next section.

3.2. Object Detection Phase

At this stage of the pipeline, the objective is to use the frames provided by the dataset as input to the detector and thus carry out the identification of vehicles in the scene. During this stage, the data is provided to the detector for use as input, and the result is the identification of vehicles.

In order to perform object detection, we employ the YOLOv4 network [14], which can be considered the state-of-the-art when real-time object detection architecture. The YOLOv4 network is pre-trained with the COCO dataset. However, for the particular case of traffic scenes, this is not enough to yield satisfactory performance. To this regard, we fine tuned the YOLOv4 network using the UAVDT dataset (described in the previous section) as detailed next.

We depart from the darknet implementation of YOLOv4[1] pre-trained on the COCO dataset. The loaded model had its first 136 layers frozen, while the remaining layers were unfrozen for training. In this way, we seek to accelerate the learning process while preserving the knowledge already acquired by the pre-trained model. We further customized the input shape of the network to 832 × 832 pixels (the original network had an input of 512 × 512 pixels) to improve the performance of the network on distant scenes. Finally, we set the batch size to 6000 samples (the original network uses 2000 samples) in order to take into account the RAM limitations of

[1] Available at https://github.com/AlexeyAB/darknet

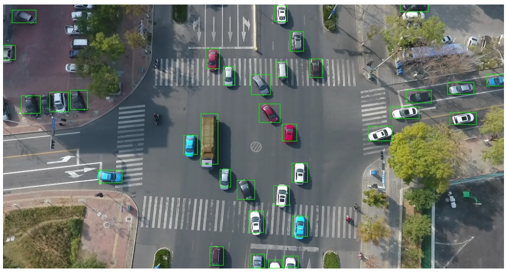
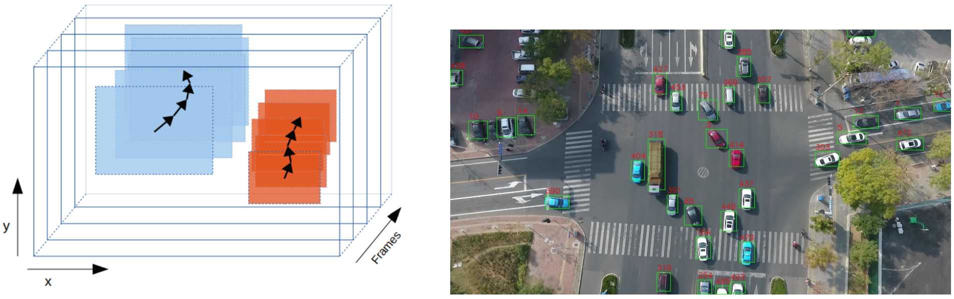


Figure 2: Output of the vehicle detection phase based on a frame from the M0603 scene.



(a)

(b)

Figure 3: Object tracking phase. Figure 3a shows a sketch of the method proposed by [26]. Figure 3b presents the output of the tracking phase based on a frame from the M0603 scene.

our setup. Fig. 2 illustrates the output of this phase.

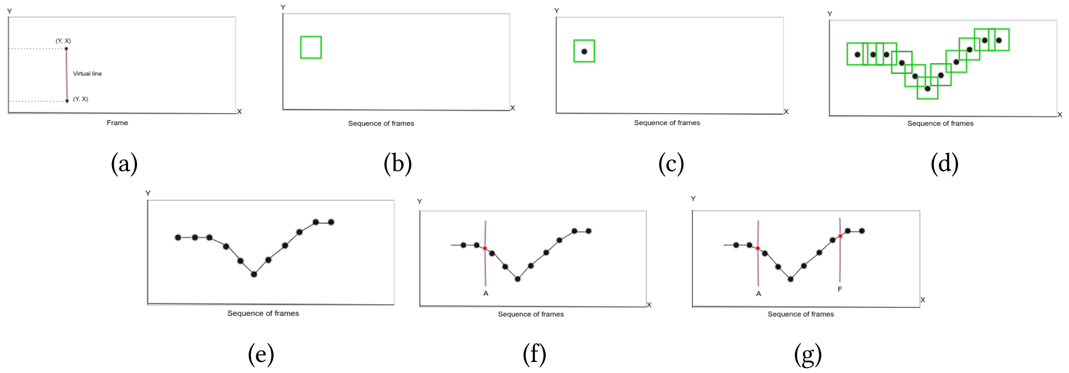
3.3. Object Tracking Phase

In the object tracking phase, each vehicle detected in the previous phase is tracked throughout a sequence of frames in order to enable the identification of its trajectory. This step is essential to enable the interpretation of the origins of destinations of the vehicles, as performed by the phase described in the next section.

We highlight that, although the object detection phase is the one responsible for identifying the vehicles, it is not able to tell whether an object appearing in subsequent frames represent the same vehicle. This is precisely the objective of the tracking phase, which needs to compare the position of vehicles in subsequent frames in order to have a unique correspondence.

In order to perform this task, we consider the tracker proposed in [26]. This method is based on the assumption that the detector produces one detection per frame for each vehicle to be tracked. In other words, for a given vehicle, the distance between its positions in subsequent frames should be small or should not exist. Furthermore, this method assumes that the bounding boxes representing the vehicle in two subsequent frames should have a high overlap, as measured by the Intersect Over Union (IoU) metric.

Figure 3a illustrates how the tracking technique works. In the figure, two objects are shown along a sequence of four frames. Each object is delimited by a bounding box and the arrow



(a) (b) (c) (d)

(e) (f) (g)

Figure 4: Origin and destination identification phase. (a) shows how the virtual lines are positioned in the scene. (b) illustrates the detection of a single vehicle in a single frame, and (c) its center. (d) presents the same vehicle being detected along multiple frames and their corresponding detection centers. (e) shows the detection centers being joined by a line, thus representing a trajectory. (f) represents the moment that the trajectory touches the virtual line A. (g) illustrates the moment that the trajectory touches virtual line F.

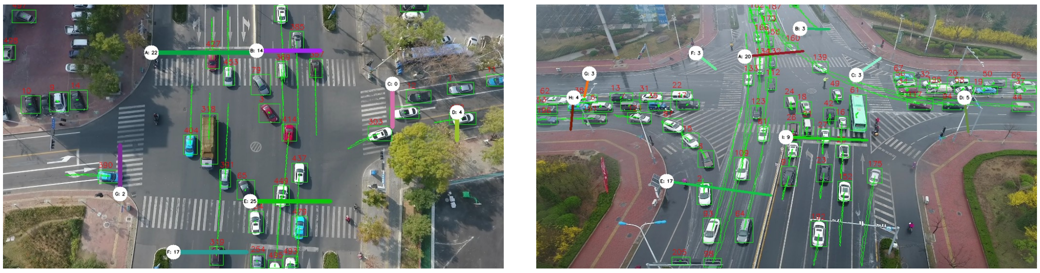
denotes its direction. Position are represented as x and y coordinates. As seen, for any object, the bounding boxes on subsequent frames present a high overlap. Building upon that observation, the trajectory of an object can be inferred by observing the position of its bounding box along frames.

In this sense, after receiving the vehicle detection coordinates, the tracking algorithm starts its verification to infer whether the vehicle is new on the scene or not. If the vehicle is new, an ID is assigned to it. Otherwise, the newly perceived coordinates are stored in an array containing all coordinates previously detected for that particular vehicle. Fig. 3b presents the result of the tracking method, where an ID is assigned to each vehicle identified in the previous stage.

3.4. Origin and Destination Identification

At this stage of the pipeline, the objective is to detect the points where a vehicle enters and exits the intersection. The idea is that, by detecting such points, one can infer the origin and destination of each vehicle. Recall that, for RL-based traffic light control, this information is of uttermost importance, as it allows the agent to properly model the flow of vehicles and make its decisions [4]. To this end, we define virtual lines in the frames to represent the entrance and exit points of the intersection, where each virtual line is composed of a pair of x and y coordinates. The sequence of steps of this phase are depicted in Fig. 4a.

In order to enable the identification of origins and destinations, virtual lines are placed at the entrances and exits of the intersections. Then, through the data provided by the tracking phase, it is possible to discover the entire trajectory of the vehicle along the scene based on the returned array of coordinates. For each coordinate, its center is calculated to generate the trajectory, as depicted in Fig. 4d. In this way, the trajectory is a set of centroids obtained from the array of coordinates, as seen in Fig. 4e.



(a)

(b)

Figure 5: Identification of the origins and destinations of the vehicles based on frames from the M0603 (Fig. 5a) and M0403 (Fig. 5b) scenes.

To be able to know where a vehicle entered and exited the intersection, we calculate the intersection between the vehicle's trajectory and the annotated virtual line, as shown in Figs. 4f and 4g. If the intersection is positive, we store the vehicle ID in an array associated with the respective straight segment. In addition, we increment a counter that shows how many vehicles have passed through that line.

In Fig. 5, we can observe the operation of the pipeline. In the figure, once vehicles and their trajectories are detected, it is possible to compute through which entrance and exit lines each vehicle goes through. Such information can then be used to count the number of vehicles on each origin and destination pair within the given intersection.

4. Experimental Evaluation

In this section, we present the experiments performed to assess our pipeline. The main objective here is to evaluate the performance of each phase of the pipeline, as well as of the complete pipeline. The evaluation methodology is presented in Section 4.1. Results are then detailed and discussed in Section 4.2.

4.1. Methodology

In order to test our pipeline, we selected the scenes M0403 and M0603 from the UAVDT dataset. For testing each phase of the pipeline, a different set of metrics is necessary. For multi-object tracking tasks, the following metrics can be used [30]:

- IDF1: Global min-cost F1 score.
- IDP: Global min-cost precision.
- IDR: Global min-cost recall.
- Recall: Number of detections over number of objects
- Precision: Number of detected objects over sum of detected and false positives.
- FP: Total number of false positives.
- FN: Total number of misses.
- IDs: Total number of track switches.

Table 2
Results for the vehicle detection and vehicle tracking methods. Arrows denote whether metrics should be maximized (↑) or minimized (↓). Best results are highlighted in bold.

Metrics	M0603	M0403	Average	State-of-art [31]
Frames	2035	514	2549	-
IDF1 ↑	82.5%	90.6%	**85.7%**	62.6%
IDP ↑	72.4%	86.1%	**77.5%**	76.00%
IDR ↑	95.9%	95.5%	**95.8%**	58.00%
Recall ↑	98.4%	97.8%	98.2%	-
Precision ↑	74.3%	88.3%	79.5%	-
FP ↓	15241	4087	19328	-
FN ↓	726	677	1403	-
IDs ↓	57	45	102	-
MOTA ↑	64.2%	84.7%	**72.6%**	43.1%
MOTP ↑	74.6%	74.1%	**74.4%**	78.5%

- MOTA: Multiple object tracker accuracy.
- MOTP: Multiple object tracker precision.

The primary metrics used to evaluate the vehicle detection method were precision and recall. In order to evaluate the vehicle tracking method, we employed MOTA and MOTP. The purpose of adopting these metrics is to highlight the model's success.

In order to evaluate the complete pipeline, we need a metric able to compare the number of vehicles on each origin-destination pair as output by our pipeline in comparison to the ground truth counting. To this end, we propose a new metric called OD Error, as shown below:

$$OD\ Counting\ Error = \sum_{i=0}^{n} \left| \frac{Detection_i - GT_i}{GT_i} \right|, \tag{1}$$

where n represents the number of virtual lines in the scene or the number of routes, and $Detection_i$ and GT_i denote the number of vehicles on virtual line i as detected by our pipeline and represented as ground truth, respectively.

Our pipeline was trained using *Google Colab Pro*, featuring a NVIDIA Tesla P100 GPU. The experiments, in turn, were performed on a standard computer with Intel Core I7 CPU with on 16GB of RAM and a NVIDIA GeForce RTX 2060 GPU.

In order to better assess the performance of our approach, we adopted [31] as a baseline method for object detection and tracking, which we consider to be the state of the art in the context of traffic. As for the origin-destination identification part, we include no comparison given the lack of works on the topic.

4.2. Results

We start the results section by analyzing the performance of our pipeline in its first two phases, the vehicle detector and the vehicle tracker. Table 2 presents the results obtained by our method and the baseline for the M0603 and M0403 scenes of the UAVDT dataset. As seen, the average

Table 3
Results for the number of vehicles passing through each virtual line for scenes M0403 and M0603.

Virtual line	M0603			M0403		
	GT	Detections	Error (N)	GT	Detections	Error (N)
A	24	24	0	28	27	1
B	26	26	0	5	3	2
C	0	0	0	3	3	0
D	5	5	0	8	8	0
E	36	36	0	22	22	0
F	25	25	0	3	3	0
G	3	3	0	3	3	0
H	-	-	-	5	5	0
I	-	-	-	9	9	0

results obtained by our pipeline outperform the current state-of-the-art approach by a good margin for most metrics. These results indicate that our pipeline is able to properly detect and track vehicles. This was made possible because we used a fine-tuned version of the YOLOv4 network, which yielded superior results than those obtained by the siamese network used in [31]. Moreover, since the our method yields better detections, the tracking phase ends up outperforming [31] as well.

The detector achieved an accuracy of 79.5%. This result directly influence the accuracy of the tracker, which ended up with an MOTP 74.4%. In order to improve the accuracy of the detector, it is necessary to train the model with more instances of the truck and bus classes, whose number was not sufficient in the UAVDT dataset. Throughout the experiments we observed that these specific classes end up failing more than cars. In addition, the use bird view images hinders the identification of smaller objects in the image. To improve this aspect, one could increase the size of the network at the time of training, which could enhance the accuracy at the cost of increasing the complexity of the object detection network.

Table 3 presents the results related to the number of vehicles that passed through each virtual line drawn in the scenes. Observe that the number of virtual lines is seven and nine for the M0603 and M0403 scenes, respectively. Results are compared to the annotated ground truth values. As seen, our method correctly detected all cases in the first scene and mistakenly detected only 3.38% of the cases in the second scene. We remark that the errors occurred because the detector failed to identify these vehicles. These errors could be reduced by improving the accuracy of the detector.

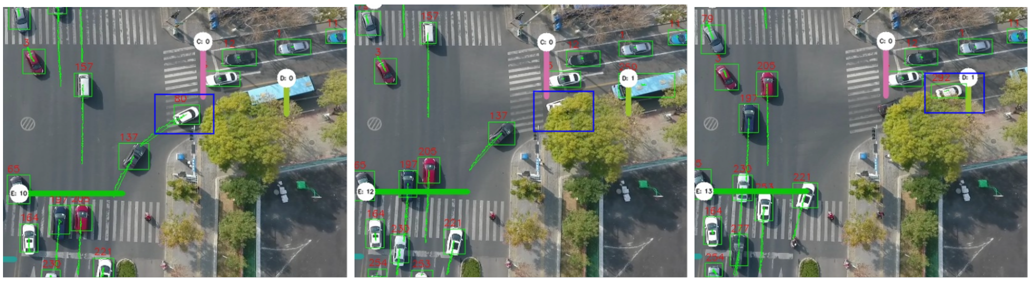
Table 4 presents the overall results of our pipeline on the M0603 scene by means of the OD Error metric. Recall that this metric shows for how many vehicles the pipeline failed to identify their complete trajectories. As seen in the table, our pipeline missed a single vehicle, which goes from A to F. The error happened because the detector failed to detect that vehicle in a single frame in the middle of the trajectory. As such, the tracker ended up assigning a different ID to the vehicle once it was detected again on the subsequent frame. From the table, it can also be observed that our pipeline presented a more substantial error for the E-D pair. In that case, the error happened because part of the scene is occluded, which prevented the detector from

Table 4
Origin-destination identification results for the M0603 scene.

Route	GT	Detections	OD Error by Route	-
A to F	22	21	1	-
E to B	26	26	0	-
E to D	5	0	5	-
G to F	3	3	0	-
OD Error % ↓	56	50	6	10.71%

Table 5
Origin-destination identification results for the M0403 scene.

Route	GT	Detections	OD Error by Route	-
A to D	5	5	0	-
A to E	12	12	0	-
C to B	3	2	1	-
F to G	3	3	0	-
OD Error % ↓	23	22	1	4.35%



(a) (b) (c)

Figure 6: Illustration of the loss of identification problem in the M0603 scene, where the vehicle detector fails to identify the vehicle in a single frame, thus leading the tracker to assign a new identifier to the vehicle once it is re-identified in later frames. In the figures, (a) a vehicle with ID 80 goes from point E to D where there is a tree occluding the view; then (b) the vehicle is lost by the vehicle detector; finally, (c) when the vehicle is re-identified as a new one.

detecting the vehicles on that part of the scene. This situation is illustrated in Fig. 6, where the vehicle with ID 80 becomes undetectable when passing close to a tree and then becomes detectable again, thus receiving a new ID.

Finally, Table 5 presents the overall results of our pipeline on the M0403 scene by means of the OD Error metric. As seen, only a single vehicle was not properly identified, which represents a percentage error of 4.35% in the identification of the complete trajectory of the vehicles. In this scene, no occlusion points exist. However, the error also happened due to a failure in the detector. After the vehicle started its trajectory entering C, the detector failed, as such the vehicle ended up receiving a new ID before ending its trajectory.

5. Conclusion

In this paper, we presented a complete pipeline to extract the origins and destinations of vehicles from scenes of traffic intersections. This information plays a role on enabling reinforcement learning agents to control traffic lights [5, 4, 21] and could also be extended to other traffic problems.

In general, the pipeline obtained promising results, achieving an average error rate as low as 7.53% in terms of origins and destinations identification. However, some aspects of the pipeline presented less impressive results when facing occlusions or even when facing trajectory interruptions. We achieved an average accuracy of 79.5%, which detracted from the tracker accuracy, which reached 74.4%.

As future work, we plan to train the model with a more significant number of instances to increase its accuracy and thus improve the tracker's accuracy. Likewise, our model could be extended to deal with multi-camera settings, which could enable overcome occlusion-related limitations. Another improvement that we are going to propose is a tracker method capable of re-identifying the vehicle when it passes through an occlusion point, which could mitigate discontinuities in the trajectory identification. Together, these changes would pave the way for a future deployment of our pipeline. Another interesting direction for future work refers to adopting some of the most recent object detection architectures, such as Vision Transformers [32] and SWIN Transformers [33], which could improve the vehicles' detection accuracy.

Acknowledgments

We thank the anonymous reviewers for their valuable feedback. This research was partially supported by Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001, Conselho Nacional de Desenvolvimento Científico e Tecnológico - CNPq (grants 306395/2017-7 and 314042/2021-0), Fundação de Amparo à Pesquisa do Estado do Rio Grande do Sul - FAPERGS (grants 17/2551-0001197-0, 19/2551-0001277-2, and 19/2551-0001334-5), and Fundação de Amparo à Pesquisa do Estado de São Paulo - FAPESP (grant 2020/05165-1).

References

[1] R. Arnott, K. Small, The economics of traffic congestion, American scientist 82 (1994) 446–455.

[2] A. L. Bazzan, F. Klügl, Introduction to intelligent systems in traffic and transportation, Synthesis Lectures on Artificial Intelligence and Machine Learning 7 (2013) 1–137. URL: https://doi.org/10.2200/s00553ed1v01y201312aim025. doi:10.2200/s00553ed1v01y201312aim025.

[3] A. Zear, P. Singh, Y. Singh, Intelligent transport system: A progressive review, Indian Journal of Science and Technology 9 (2016). doi:10.17485/ijst/2016/v9i32/100713.

[4] P. Mannion, J. Duggan, E. Howley, An experimental review of reinforcement learning algorithms for adaptive traffic signal control, Autonomic road transport support systems (2016) 47–66.

[5] H. Wei, G. Zheng, V. V. Gayah, Z. Li, A survey on traffic signal control methods, ArXiv abs/1904.08117 (2019). arXiv:1904.08117.

[6] J. Zhang, F.-Y. Wang, K. Wang, W.-H. Lin, X. Xu, C. Chen, Data-driven intelligent transportation systems: A survey, IEEE Transactions on Intelligent Transportation Systems 12 (2011) 1624–1639.

[7] N. Chintalacheruvu, V. Muthukumar, Video based vehicle detection and its application in intelligent transportation systems, Journal of Transportation Technologies 02 (2012) 305–314. URL: https://doi.org/10.4236/jtts.2012.24033. doi:10.4236/jtts.2012.24033.

[8] S.-C. Huang, B.-H. Chen, Highly accurate moving object detection in variable bit rate video-based traffic monitoring systems, IEEE Transactions on Neural Networks and Learning Systems 24 (2013) 1920–1931. URL: https://doi.org/10.1109/tnnls.2013.2270314. doi:10.1109/tnnls.2013.2270314.

[9] Y. Iwasaki, M. Misumi, T. Nakamiya, Robust vehicle detection under various environmental conditions using an infrared thermal camera and its application to road traffic flow monitoring, Sensors 13 (2013) 7756–7773. URL: https://doi.org/10.3390/s130607756. doi:10.3390/s130607756.

[10] A. Arinaldi, J. A. Pradana, A. A. Gurusinga, Detection and classification of vehicles for traffic video analytics, Procedia Computer Science 144 (2018) 259–268. URL: https://doi.org/10.1016/j.procs.2018.10.527. doi:10.1016/j.procs.2018.10.527.

[11] Y. Tang, D. Wu, Z. Jin, W. Zou, X. Li, Multi-modal metric learning for vehicle re-identification in traffic surveillance environment, in: 2017 IEEE International Conference on Image Processing (ICIP), IEEE, 2017. URL: https://doi.org/10.1109/icip.2017.8296683. doi:10.1109/icip.2017.8296683.

[12] H. Yang, S. Qu, Real-time vehicle detection and counting in complex traffic scenes using background subtraction model with low-rank decomposition, IET Intelligent Transport Systems 12 (2017) 75–85. URL: https://doi.org/10.1049/iet-its.2017.0047. doi:10.1049/iet-its.2017.0047.

[13] Y. Tang, Y. C. Xu, C. Z. Zhang, Robust vehicle detection based on cascade classifier in traffic surveillance system, The Open Automation and Control Systems Journal 6 (2014) 349–354. URL: https://doi.org/10.2174/1874444301406010349. doi:10.2174/1874444301406010349.

[14] C.-Y. Wang, A. Bochkovskiy, H.-Y. M. Liao, Scaled-yolov4: Scaling cross stage partial network, 2021. arXiv:2011.08036.

[15] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, A. C. Berg, Ssd: Single shot multibox detector, in: European conference on computer vision, Springer, 2016, pp. 21–37.

[16] H. Law, J. Deng, Cornernet: Detecting objects as paired keypoints, in: Proceedings of the European conference on computer vision (ECCV), 2018, pp. 734–750.

[17] H. Law, Y. Teng, O. Russakovsky, J. Deng, Cornernet-lite: Efficient keypoint based object detection, arXiv preprint arXiv:1904.08900 (2019).

[18] R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2014, pp. 580–587.

[19] R. Girshick, Fast r-cnn, in: Proceedings of the IEEE international conference on computer vision, 2015, pp. 1440–1448.

[20] S. Ren, K. He, R. Girshick, J. Sun, Faster r-cnn: Towards real-time object detection with

region proposal networks, Advances in neural information processing systems 28 (2015) 91–99.

[21] L. Schreiber, L. N. Alegre, A. L. C. Bazzan, G. de. O. Ramos, On the explainability and expressiveness of function approximation methods in rl-based traffic signal control, in: 2022 International Joint Conference on Neural Networks (IJCNN), IEEE, Padova, Italy, 2022. Forthcoming.

[22] Z. Dai, H. Song, X. Wang, Y. Fang, X. Yun, Z. Zhang, H. Li, Video-based vehicle counting framework, IEEE Access 7 (2019) 64460–64470.

[23] J. Wang, S. Simeonova, M. Shahbazi, Orientation-and scale-invariant multi-vehicle detection and tracking from unmanned aerial videos, Remote Sensing 11 (2019) 2155.

[24] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, C. L. Zitnick, Microsoft coco: Common objects in context, in: European conference on computer vision, Springer, 2014, pp. 740–755.

[25] D. Du, Y. Qi, H. Yu, Y. Yang, K. Duan, G. Li, W. Zhang, Q. Huang, Q. Tian, The unmanned aerial vehicle benchmark: Object detection and tracking, 2018. arXiv:1804.00518.

[26] E. Bochinski, V. Eiselein, T. Sikora, High-speed tracking-by-detection without using image information, in: International Workshop on Traffic and Street Surveillance for Safety and Security at IEEE AVSS 2017, 2017. URL: 1.

[27] A. Bewley, Z. Ge, L. Ott, F. Ramos, B. Upcroft, Simple online and realtime tracking, in: 2016 IEEE international conference on image processing (ICIP), IEEE, 2016, pp. 3464–3468.

[28] N. Wojke, A. Bewley, D. Paulus, Simple online and realtime tracking with a deep association metric, in: 2017 IEEE international conference on image processing (ICIP), IEEE, 2017, pp. 3645–3649.

[29] F. Liu, Z. Zeng, R. Jiang, A video-based real-time adaptive vehicle-counting system for urban roads, PloS one 12 (2017) e0186098.

[30] A. Milan, L. Leal-Taixe, I. Reid, S. Roth, K. Schindler, Mot16: A benchmark for multi-object tracking, 2016. arXiv:1603.00831.

[31] H. Yu, G. Li, W. Zhang, Q. Huang, D. Du, Q. Tian, N. Sebe, The unmanned aerial vehicle benchmark: Object detection, tracking and baseline, International Journal of Computer Vision 128 (2020) 1141–1159.

[32] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al., An image is worth 16x16 words: Transformers for image recognition at scale, arXiv preprint arXiv:2010.11929 (2020).

[33] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, B. Guo, Swin transformer: Hierarchical vision transformer using shifted windows, in: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2021, pp. 10012–10022.