Incident Detection on Junctions Using Image Processing
Murat Tulgac, Enes Yünci, Mohamad-Alhaddad and Ceylan Yozgatlıgil

Abstract—In traffic management, it is a very important issue to shorten the response time by detecting the incidents (accident, vehicle breakdown, an object falling on the road, etc.) and informing the corresponding personnel. In this study, an anomaly detection framework for road junctions is proposed. The final judgment is based on the trajectories followed by the vehicles. Trajectory information is provided by vehicle detection and tracking algorithms on visual data streamed from a fisheye camera. Deep learning algorithms are used for vehicle detection, and Kalman Filter is used for tracking. To observe the trajectories more accurately, the detected vehicle coordinates are transferred to the bird’s-eye view coordinates using the lens distortion model prediction algorithm. The system determines whether there is an abnormality in trajectories by comparing historical trajectory data and instantaneous incoming data. The system also works with a %97.3 success rate in detecting abnormalities on real data.

Index Terms—Anomaly detection, Deep learning, Traffic management.

I. INTRODUCTION

The demand for transportation rises year by year due to the increase in the population. As a result, traffic congestion occurs by exceeding the capacity of existing roads. Traffic congestions are examined in two groups as recurrent and non-recurrent ones. Recurrent congestions are based on a certain routine, and not affected by temporary events. Non-recurrent congestions are experienced due to weather or temporary events in traffic.

It is critical to inform the operators by identifying accidents, vehicle malfunctions, falling objects, or similar temporary events as soon as possible, especially at intersections. Fortunately, the study of smart monitoring systems has been a great interest in research and development. Remarkable progress has been achieved to deploy traffic monitoring systems that will detect abnormalities at intersections and notify who is in charge of taking action.

In this paper, we address the use of an omnidirectional (fisheye) camera for intersection monitoring. We provide a framework for incident detection which consists of three steps:

1) Detect vehicles in the camera Field of View (FoV)
2) Track the vehicles entering/exiting the FoV
3) ComAPre their followed paths trajectories with the normal path and detect any abnormality.

In order to detect an anomaly, normal behavior of the specific route must be characterized [1]. Afterward, by monitoring the system behavior continuously, the anomaly can be detected by defining time intervals that do not comply with normal behavior. In the proposed method, the difference between the normal behavior and the behavior of the entering vehicles is evaluated on the trajectories followed by the vehicles. For trajectory extraction, we utilized deep learning techniques to develop our vehicle detector, then we mapped the scene to the bird’s-eye view. Finally, we tracked the vehicles using Munkres and Kalman filter methods.

In this study, we propose a framework that detects trajectories on the images taken from a fisheye camera installed at the intersection and detects abnormalities by comAPring these trajectories with the past trajectories. This framework acts as a subsystem for Intelligent Traffic System (ITS). The flow chart of the proposed method is shown below in Figure 1.

Fig. 1: Flow chart of the proposed system.

The rest of the study is evaluated under four headings: Section-2 addresses the literature advancement in incident detection at intersections. Section-3 illustrates the synthetic and real data used in the study. In the Section-4, the proposed method is explained and sub-steps related to the method are presented. In Section-5, the results of the study on synthetic and real data, conclusions, and summary of the study are presented.
II. RELATED STUDIES

The omnidirectional cameras are known for their wider Field of View (FoV) than conventional cameras. Thus, using them in monitoring tasks is beneficial in terms of setup and maintenance costs. The studies addressing omnidirectional cameras have reported several challenges faced in capturing a large image and analyze the highly distorted objects at variable scales in real-time.

In their work, Lee et al. [2] have created a vehicle trip table estimation which tracks the origin-destination points of vehicles passing through the intersection. Their tracker relies on particle filters and motion dynamics and has suffered from appearance changes and different weather conditions which makes it limited in robustness. Wang et al. [3] have addressed the issues faced by relying on a fisheye camera for vehicle counting at an intersection, due to severe distortion and perspective effects. They came up with a method based on motion similarity and probabilistic weights to transfer the motion knowledge between long and short-term trajectories between points. Other attempts to surmount the distortion was to track motion based on background subtraction models [4, 5, 6].

Other studies have addressed the same problem with standard cameras (monocular vision), and remarkable outputs were reported so far in the literature. We found out that the researchers usually attempt to do background subtraction or feature extraction and tracking variants at the first step for detection purposes, then tracking the vehicle path with other perimeters [7, 8, 9, 10, 11].

To track the behavior, some studies attempt to rely on the vehicle size ratio in correlation with its direction, velocity, and acceleration obtained from the detection step [12] to understand the onsite situation, while others rely on kernel-based filters to determine whether the distribution of the detected pixels represents an accident or not [13].

The main difference between existing attempts and our approach is the lack of incident detection exclusiveness. The current studies focus on the situation where two vehicles may collide, while our approach can detect abnormality based on the passing vehicles’ trajectories characteristics. Another aspect is that we embed deep learning trends for ITS applications where addressed research has relied heavily on background subtraction which suffers in terms of robustness, especially for omnidirectional vision.

III. DATA

Two different data are used in the development and testing phases of the algorithm. These are simulation and real data of vehicles passing the intersection. An algorithm has been developed in which accurate results can be obtained for anomaly detection in both real and simulated data.

A. Simulation Data

Simulation data is obtained from the simulation program called VisSim [14]. This data is used in the creation of deviation detection algorithms in trajectory. ComAPred to the fisheye dataset, simulation data has a lower noise rate (Figures 9-10). This caused the trajectories of the vehicles in the simulation data to be much smoother than the actual data. Because of that, new tests were carried out on the fisheye dataset after implementing and testing the algorithm on simulation data to develop a more robust process.

Simulation data is created by simulating the 1-hour flow of a junction of a city in Turkey that is used while the fisheye camera dataset is created. Seven different scenarios (incident at different specific routes, no incidents) are created to test how the system will respond to different situations. In addition to these different scenarios, to characterize the normal behavior, the simulation scenario with no incident was run 30 times. Hence, 37 hours of simulation data are used. In this data, as with tracking algorithms, vehicle-based coordinate data are obtained, and consistency with real data is achieved. The simulation produces several vehicles $V_1, V_2, \ldots, V_n$ passing through the junction with different followed paths. Each followed path is sampled over several subsequent locations $L_{x,y}$ from the junction entrance until exit. $(x, y)$ are the Cartesian coordinates of the vehicle’s $V$ location $L$.

After the vehicle-based coordinate data is normalized on all data, the trajectories of the vehicles are sorted by time, and the differences from previous trajectories are observed. Related images for the intersection used in the study and simulation data are shown in Figure 2.

B. Fisheye Dataset

We had the access to the installed fisheye surveillance cameras mounted at 8 meters height from the ground on junctions from a city in Turkey. The coverage of the camera reaches a clear human observable view of around 75 meters from the junction entrance point. Each road can accommodate 3 lanes of vehicles that vary in size (car, truck, bus). The obtained dataset [15] is acquired during daytime and sampled at 10fps frames, resulting in 20,000 frames, a sample of the mentioned data is in Figure 3.

IV. PROPOSED METHOD

A. Vehicle Detection

Vehicle detection is a variant of the classic object detection problem in computer vision research. Several approaches have addressed detecting vehicles with different study scopes such as autonomous vehicles [16, 17, 18, 19, 20, 21, 22, 23, 24, 25] and smart cities models [26, 27, 28, 29, 30, 31, 32, 33, 34]. In our proposed framework, the first step is to detect vehicles at the junction, the capturing is via a static camera with a fisheye lens.

The fisheye lens has introduced a distortion problem in the vehicle shapes which would fail several techniques used in the research, where standard cameras where used. Some studies tried to overcome the distortion issue with different techniques. In one approach [35] they have trained a network over the mapping between a fisheye captured lines image and straight lines rectified image to overcome that issue, then let the network auto-rectify any new fisheye incoming image. Affine transformations have also proved to be a solution to overcome the lens effect [36, 37]. However, it is considered
to be a computationally expensive operation as a pre-process \[38\]. Another study \[39\] suggests working on the camera calibration itself. Although the found literature is massively informative, the nature of the open environment has limited us from having costly detection pre-processing computation or hardware calibration.

Here we follow the "use it as it exists" approach to come up with a detection algorithm. We target the fisheye camera as the feed source with no preprocessing taking place, thus eliminating any deployment ambiguities of configuring the algorithm at the post-training stage.

The above-mentioned fisheye data is split into train-test subsets 80\% by 20\% respectively. We trained several models on the raw images and tabulated the results in Table-1. The model configuration is cloned from \[40\]. It is found that model YoloV3 \[41\] overperforms YoloV3-tiny and the Faster-RCNN in the defined scores. Thus, it is selected for the vehicle detection stage. It is well known that relying on deep learning solutions for detection would leverage robustness, but penalize computation time and resources allocation \[42\].

To evaluate the model, we defined several parameters. For measuring how much punctual the model is we used PASCAL \[43\] average precision formula at 3 different Jaccard index (a.k.a IoU) thresholds of interest 50, 75 and 95 \%. The thresholds are picked to get a better understanding of model performance since any vehicle’s size in fisheye camera changes massively. For the inference time measurement, the runtime CPU is INTEL core i7-8750H. Due to the existence of several parameters, we defined a formula (Equation-1) to pick up the best model, as illustrated in Table-1.

\[
\text{score} = 0.4 \times \text{mAP}_{\text{IoU}50\%} + 0.3 \times \text{mAP}_{\text{IoU75\%}} + 0.1 \times \text{mAP}_{\text{IoU95\%}} + 0.2 \times (1 - \text{inference})
\]

**TABLE I: Vehicle detection models**

| Model      | mAP_{\text{IoU}50\%} | mAP_{\text{IoU}75\%} | mAP_{\text{IoU}95\%} | Inference (msec) | Score (%) |
|------------|------------------------|------------------------|------------------------|------------------|-----------|
| YoloV3     | 95.6                   | 89.9                   | 80.7                   | 250              | 88.28     |
| YoloV3-tiny| 89.5                   | 80.9                   | 55                     | 80               | 83.97     |
| Faster-RCNN| 85                     | 75                     | 70                     | 420              | 75.1      |

**B. Fisheye to Bird’s-eye View Conversion**

The wide field of view (FoV) of a fisheye camera benefits in better image coverage compared to a standard surveillance camera \[36, 44\]. After the vehicles are detected in the fisheye view, it is required to obtain vehicles’ trajectories. To limit the computation resources required for this step, it is favored to do a simple pixel transformation mapping on the initial frame and use the obtained transformation parameters to project the trajectories into the bird’s-eye view. To do so, as a post-processing step, a geometrical rectification procedure takes place as follows:

1) Define \(L_f(x_c, y_c)\) as the center point (in pixels) of the vehicle’s \(V\) bounding box resulted from the detector at fisheye view
2) Manually dictate the initial frame key points for rectification to bird eye view
3) Obtain the transformation map matrix
4) Apply the transformation map matrix against vehicle location \(L_f\) to get \(L_b\) as the vehicle \(V\) location at the bird’s-eye view; we denote \(L_b\) as our \(L\) for the sake of simplicity.

The algorithm of distortion model estimation is described in Equation-2:

\[
D(P_f, P_c) = K(z) \times D(P_b, P_c).
\]

![Fig. 2: Intersection footage. (a) Drone recorded image of the intersection used in the study. (b) Simulated image of the intersection. (c) Routes in the simulation data.](image)

![Fig. 3: Example of the fisheye data.](image)
where $D(a,b)$ is defined as $\begin{pmatrix} a_x - b_x \\ a_y - b_y \end{pmatrix}$. $P_c$ is defined to be the center point of distortion which is the center of the image frame, $P_f$ is the point $P$ in the fisheye view while $P_b$ is the point $P$ in the bird’s-eye view. In order to obtain the trajectory point $P_b$ of $P_f$ it is required to solve the Equation-3 for $K(z)$ where $K(z)$ is the function approximating the distortion as a Taylor series defined as:

$$K(z) = q_0 + q_1z + q_2z^2 + q_3z^3 + \cdots$$  \hspace{1cm} (2)

The $z$ is the Euclidean distance between $P_c$ and $P_f$:

$$z = \sqrt{(P_{cx} - P_{f_{x}})^2 + (P_{cy} - P_{f_{y}})^2}.$$  \hspace{1cm} (3)

Solving for $K$ requires obtaining the distortion parameters set $(q_0, q_1, q_2, \ldots)$. This is done at the initially obtained frame, where a line of points $P_f$ is drawn on the fisheye image following the path of the street, as in Figure 4. Lastly using the obtained distortion parameters we are able to project each vehicle location $L_f$ to $L_b$.

![Image](image.png)

Fig. 4: On the left is a fisheye frame having a drawn line of points $P_f$ is defined to follow the street path, on the right is the rectified bird’s-eye version of the frame.

### C. Tracking the vehicle trajectory

To define the vehicle trajectory it is required to track each passing vehicle throughout the junction. Research community has addressed object tracking problem with geometrical algebra \cite{48, 49, 50, 51, 52} as well as recent deep learning-based solution \cite{53, 54, 55, 56, 57}. The requirement to have a lightweight solution for trajectory extraction emphasizes going with the Munkres-Kalman combination, thus achieving high accuracy and computationally efficient solution.

Several studies have addressed object association using the Munkres algorithm. In the study of body parts association of COCO 2016 challenge Cao et al. have suggested the Munkres algorithm for optimal results \cite{49}. Others like Mozaffari et al. have used it in Unmanned Autonomous Vehicle (UAV) deployment and association problems in order to keep track of the devices \cite{58, 59, 60, 61, 62}.

The Munkres is to link identified objects at frame $F_{t-1}$ and frame $F_t$. To utilize it in our vehicle association and have a track of the vehicles’ Spatio-temporal status, we have favored the conditional context to be the Jaccard Index score. Although some addressed studies have preferred to use Euclidean distance \cite{48} the fact of vehicles can get too close may trick the algorithm at that point.

$$J_{t-1,t} = \begin{pmatrix} j_{1_{t-1},1_t} \\ j_{2_{t-1},2_t} \\ j_{3_{t-1},3_t} \end{pmatrix}$$  \hspace{1cm} (4)

where $j_{i_{t-1},i_t} = Jaccard(V_{t-1}, V_t)$ and $V$ is the vehicle detected at location $L$ in the previous step. The method will associate the vehicles by obtaining the maximum IoU $Max(J_{t-1,t})$, as illustrated in Figure 5.

![Diagram](diagram.png)

Fig. 5: Vehicle $V$ trajectory tracking procedure

After assigning the vehicles, it is required to track their path throughout the junction. The more the path tracking mechanism is consistent, the better the proposed method performance. Kalman filter has proved to be a state of the art tracking method, deployed in several studies \cite{48, 49, 50, 51, 52}. We use Kalman filter to estimate vehicle $V$ next location $L_t$ based on $L_{t-1}$ and smooth the trajectory produced in case the detector failed to detect $V$ at a frame $F$. The vehicle location $L$ is defined for the tracker when the $V$ enters the camera FoV and is dismissed when it leaves the FoV. The output of this stage can be formulated as below:

$$\hat{V} = Kalman(Munkres(V_{0:L}))$$  \hspace{1cm} (5)

where $V \in \{V_{0:L0}, V_{1:L1}, \ldots\}$ and $\hat{V}$ denotes the set of tracked vehicles in the scene at time $t$.

### D. Route Classification

At the initial stage, each route in the junction is drawn manually over the image frame. The route line consists of 8 to 10 connected points are fitted to 2 to 7-degree polynomials as shown in Figure 6-a and 6-b.

After that, the mean absolute errors are calculated to correlate the vehicle $V$’s trajectory with the defined routes, with the formula in Equation-7. Finally, the vehicle $V$ belongs to the route with the lowest error.
Fig. 6: An example route. (a) The points for the route. (b) The related line of the route.

\[
error = \frac{\sum_{i}^{n} | P_{byi} - \hat{y}_{i} |}{n}. \tag{6}
\]

This process is done for each route. An example route classification process is shown below in Figure 7. The original and bird-eye view of fisheye frame and classified routes are shown in Figure 8.

Fig. 7: A sample of route classification for one route. Red ones are the followed trajectory and blue line is the manually dictated route.

Fig. 8: Example of a fisheye frame and bird eye view. (a) The fisheye frame. (b) The bird-eye view. (c) Route directions. (d) Classified Routes. Having the average error value below the error threshold is recorded for each route class.

2) Runtime: The next day, after determining the class of the incoming data by using the minimum mean square error, it is merged with the data of the past 5 vehicles in the same class. Merged data is modeled by the polynomial functions with different degrees from 1 to 20, the average error value is calculated, and the lowest degree is compared with the previous day’s recorded lowest degree to determine whether there is a deviation in the route. Examples of the deviations in the route for simulation data and fisheye data are shown in Figures 9 and 10.

If the degree difference between the lowest degrees is more than 2, it is classified as an anomaly. This threshold value is decided by the box-plot extreme value detection procedure. All quartiles of the simulation data are calculated. Then, the interquartile range (IQR) and whisker lengths are obtained as follows: The upper whisker of the box plot is the largest data point smaller than 1.5*IQR above the third quartile, and the lower whisker of the box plot is the smallest data point larger than 1.5*IQR below the first quartile. If the newly observed data is below the lower whisker or above the upper whisker, it is labeled as an anomaly. In our simulation data, the threshold value to decide anomaly is calculated as the 2-degree difference between the fitted polynomial functions. The IQR method does not depend on any loops and is, therefore, faster and more easily scaled [58]. An example of how this anomaly detection works can be seen in Figure 11. Here, the second vehicle on the route made an accident, and this method catches it immediately.

To support the decision of degree threshold, a simulation scenario that has no incident was run for 30 times to
characterize the normal behavior of the trajectories. Several resources [59, 60] were used to decide how many times the simulation should be run. After simulating for 30 times, two route types were selected to analyze the degree change between simulations. Selected routes and related results are shown below in Figure 12 and Table 2. As it is seen from the results, the chosen difference threshold value is supported by the simulation results.

V. CONCLUSION

In this study, a framework for incident detection at an intersection is developed. An omnidirectional (fisheye) camera acts as a data source, and based on the vehicles’ trajectories, anomaly occurrence decision is made. The evaluation measures for algorithms are shown below in Table 3. Since the proposed method includes merging the incoming data with the data of the past 5 vehicles, there are wrongly classified vehicle trajectories because of taking the average residue values for these 6 vehicles. However, we found that all routes with
deviations are successfully detected.

| Algorithm | Anomaly Detection (Synthetic data) | Precision(%) | Recall(%) | F1-Score(%) |
|-----------|-----------------------------------|--------------|----------|-------------|
|           | Anomaly Detection (Synthetic data - Vehicle-based) | 97.2 | 98 | 98 |
|           | Anomaly Detection (Synthetic data - Sensor-based) | 95.8 | 97.1 | 96.5 |

The proposed method does not include u-turns. Therefore, these turns are not taken into account for the evaluation. It is planned to carry out a future study including the u-turns. It is also planned to identify and analyze the roads joining the intersection where the incident takes place.

REFERENCES

[1] A. Soule, K. Salamatian, and N. Taft, “Combining filtering and statistical methods for anomaly detection,” in Proceedings of the 5th ACM SIGCOMM conference on Internet measurement - IMC ’05, 2005.

[2] S. Lee and H. Baik, "Origin-Destination (O-D) Trip Table Estimation using Traffic Movement Counts from Vehicle Tracking System at Intersection," IECON 2006 - 32nd Annual Conference on IEEE Industrial Electronics, Paris, 2006, pp. 3332-3337, doi: 10.1109/IECON.2006.347812.

[3] W. Wang, T. Gee, J. Price and H. Qi, "RealTime Multi-vehicle Tracking and Counting at Intersections from a Fisheye Camera," 2015 IEEE Winter Conference on Applications of Computer Vision, Waikoloa, HI, 2015, pp. 17-24, doi: 10.1109/WACV.2015.10.

[4] T. F. Gee and J. R. Price, “Omnidirectional imaging and computer vision for transportation applications: From conception to deployment,” AlidisCorp, Knoxville, TN, USA, Report, 2011.

[5] A. Ghorayeb, A. Potelle, L. Devendeville, and E. M. Mouaddib, “Capteur omnidirectionnel Optimal pour le diagnostic de la circulation dans les carrefours urbains,” in ORASIS’09 - Congrès des jeunes chercheurs en vision par ordinateur, 2009.

[6] M. Gressmann, G. Palm and O. Löhrlein, "Surround view pedestrian detection using heterogeneous classifier cascades," 2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC), Washington, DC, 2011, pp. 1317-1324, doi: 10.1109/ITSC.2011.6082895.

[7] T. Furuya and C. J. Taylor, “Road intersection monitoring from video with large perspective deformation,” Proc. 21st World Congress on Intelligent Transportation Systems, Sep. 2014, pp. 1–12.

[8] Y. Ki and D. Lee, "A Traffic Accident Recording and Reporting Model at Intersections," in IEEE Transactions on Intelligent Transportation Systems, vol. 8, no. 2, pp. 188-194, June 2007, doi: 10.1109/TITS.2006.890070.

[9] M. Molinier, T. Häme, and H. Ahola, “3D-connected components analysis for traffic monitoring in image sequences acquired from a helicopter,” in Image Analysis, Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 141–150.

[10] C. Caraffi, T. Vojíř, J. Třefný, J. Šochman and J. Matas, "A system for real-time detection and tracking of vehicles from a single car-mounted camera," 2012 15th International IEEE Conference on Intelligent Transportation Systems, Anchorage, AK, 2012, pp. 975-982, doi: 10.1109/ITSC.2012.6338748.

[11] N. Saunier and T. Sayed, "A feature-based tracking algorithm for vehicles in intersections," The 3rd Canadian Conference on Computer and Robot Vision (CRV'06), Quebec, Canada, 2006, pp. 59-59, doi: 10.1109/CRV.2006.3.

[12] S. Yogamani et al., “WoodScape: A multi-task, multi-camera fisheye dataset for autonomous driving,” arXiv [cs.CV], pp. 9308–9318, 2019.

[13] Y. Chen, Y. Yu and T. Li, “A vision based traffic accident detection method using extreme learning machine,” 2016 International Conference on Advanced Robotics and Mechatronics (ICARM), Macau, 2016, pp. 567-572, doi: 10.1109/ICARM.2016.7606983.

[14] M. Fellendorf and P. Vortisch, “Microscopic traffic flow simulator VISSIM,” in Fundamentals of Traffic Simulation, New York, NY: Springer New York, 2010, pp. 63–93.

[15] “Video & Image Processing Cup,” IEEE Signal Processing Society, 26-Oct-2020. [Online]. Available: https://signalprocessingsociety.org/community-involvement/video-image-processing-cup. [Accessed: 02-Dec-2020].

[16] M. Bertozzi, A. Broggi, A. Fascioli and S. Nichele, “Stereo vision-based vehicle detection,” Proceedings of the IEEE Intelligent Vehicles Symposium 2000 (Cat. No.00TH8511), Dearborn, MI, USA, 2000, pp. 39-44, doi: 10.1109/IVS.2000.898315.

[17] Xin Li, XiaoCao Yao, Y. L. Murphey, R. Karlsten and G. Gerhart, "A real-time vehicle detection and tracking system in outdoor traffic scenes," Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004., Cambridge, 2004, pp. 761-764 Vol.2, doi: 10.1109/ICPR.2004.1334370.

[18] P. Dollár, R. Appel, S. Belongie and P. Perona, “Fast Feature Pyramids for Object Detection,” in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 36, no. 8, pp. 1532-1545, Aug. 2014, doi: 10.1109/TPAMI.2014.2300479.

[19] Zehang Sun, G. Bebis and R. Miller, “Monocular pre-crash vehicle detection: features and classifiers,” in IEEE Transactions on Image Processing, vol. 15, no. 7, pp. 1929-1939, July 2006, doi: 10.1109/TIP.2006.877062.

[20] M. Bertozzi, A. Broggi, A. Fascioli and S. Nichele, “Stereo vision-based vehicle detection,” Proceedings of the IEEE Intelligent Vehicles Symposium 2000 (Cat. No.00TH8511), Dearborn, MI, USA, 2000, pp. 39-44, doi: 10.1109/IVS.2000.898315.

[21] S. Yogamani et al., “WoodScape: A multi-task, multi-camera fisheye dataset for autonomous driving,” arXiv [cs.CV], pp. 9308–9318, 2019.

[22] G. Hartmann and R. Klette, “Cylinder Sweep: Fishye Images into a Bird ’ s eye View,” 2011. https://www.overleaf.com/project/5ec39941f2f48b00013b66c8
[23] A. Petrovskaya and S. Thrun, “Model based vehicle detection and tracking for autonomous urban driving,” Auton. Robots, vol. 26, no. 2–3, pp. 123–139, 2009.

[24] Zehang Sun, G. Bebis and R. Miller, "On-road vehicle detection: a review," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 28, no. 5, pp. 694-711, May 2006, doi: 10.1109/TPAMI.2006.104.

[25] R. Brehar and S. Nedevschi, “Scan window based pedestrian recognition methods improvement by search space and scale reduction,” in 2014 IEEE Intelligent Vehicles Symposium Proceedings, Jun. 2014, pp. 529–534. doi: 10.1109/IVS.2014.6856571.

[26] H. Hu, Q. Cai, D. Wang, J. Lin, M. Sun, P. Krahenbuhl, T. Darrell, and F. Yu, “Joint Monocular 3D Vehicle Detection and Tracking,” in Proc. Int. Conf. on Comput. Vis., 2019, pp. 5390-5399.

[27] R. Guerrero-Gómez-Olmedo, R. J. López-Sastre, S. Maldonado-Bascón, and A. Fernández-Caballero, “Vehicle tracking by simultaneous detection and viewpoint estimation,” in Natural and Artificial Computation in Engineering and Medical Applications, Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 306–316.

[28] S. Kul, S. Eken, and A. Sayar, “Distributed and collaborative real-time vehicle detection and classification over the video streams,” Int. J. Adv. Robot. Syst., vol. 14, no. 4, p. 172988141772078, 2017.

[29] J. Choi, “Realtime On-Road Vehicle Detection with Optical Flows and Haar-like feature detector,” University of Illinois at Urbana-ChamAPign, Computer Science Research and Tech Reports, 2006.

[30] Xin Li, XiaoCao Yao, Y. L. Murphey, R. Karlsen and G. Gerhart, “A real-time vehicle detection and tracking system in outdoor traffic scenes,” Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004., Cambridge, 2004, pp. 761-764 Vol.2, doi: 10.1109/ICPR.2004.1334370.

[31] C. Ozkurt and F. Camci, “Automatic traffic density estimation and vehicle classification for traffic surveillance systems using neural networks,” Math. Comput. Appl., vol. 14, no. 3, pp. 187–196, Dec. 2009.

[32] N. Seenouvong, U. Watchareerutai, C. Nuthong, K. Khongsomboon and N. Ohnishi, "A computer vision based vehicle detection and counting system," 2016 8th International Conference on Knowledge and Smart Technology (KST), Chiangmai, 2016, pp. 224-227, doi: 10.1109/KST.2016.7440510.

[33] A. Somov, C. Dupont and R. Giaffreda, "Supporting smart-city mobility with cognitive Internet of Things," 2013 Future Network & Mobile Summit, Lisboa, 2013, pp. 1-10.

[34] H. U. Unlu, P. S. Niehaus, D. Chirita, N. Evangelion and A. Tzes, "Deep Learning-based Visual Tracking of UAVs using a PTZ Camera System," IEECON 2019 - 45th Annual Conference of the IEEE Industrial Electronics Society, Lisbon, Portugal, 2019, pp. 638-644, doi: 10.1109/IECON.2019.8927731.

[35] Z.-C. Xue, N. Xue, and G.-S. Xia, “Fisheye distortion rectification from deep straight lines,” arXiv [cs.CV], 2020.

[36] L. Alvarez, L. Gomez, and J. R. Sendra, “Algebraic lens distortion model estimation,” Image process. line, vol. 1, pp. 1–10, 2010.

[37] A. Furnari, G. M. Farinella, A. R. Bruna and S. Battiatto, "Affine Covariant Features for Fisheye Distortion Local Modeling," in IEEE Transactions on Image Processing, vol. 26, no. 2, pp. 696-710, Feb. 2017, doi: 10.1109/TIP.2016.2627816.

[38] H. V. Luong and J. Kim, “A Massively Parallel Approach to Affine Transformaation in Medical Image Registration,” 2009 11th IEEE International Conference on High Performance Computing and Communications, Seoul, 2009, pp. 117-123, doi: 10.1109/HPCC.2009.13.

[39] X. Ying, Z. Hu, and H. Zha, “Fisheye lenses calibration using straight-line spherical perspective projection constraint,” in Computer Vision – ACCV 2006, Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 61–70.

[40] M. H. Yap et al., “Deep learning in diabetic foot ulcers detection: A comprehensive evaluation,” arXiv [cs.CV], 2020.

[41] J. Sang et al., “An improved YOLOv2 for vehicle detection,” Sensors (Basel), vol. 18, no. 12, p. 4272, 2018.

[42] J. Zhang, M. Huang, X. Jin, and X. Li, “A real-time Chinese traffic sign detection algorithm based on modified YOLOv2,” Algorithms, vol. 10, no. 4, p. 127, 2017.

[43] M. Everingham, S. A. Eslami, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, “The pascal visual object classes challenge: A retrospective,” International Journal of Computer Vision, vol. 111, no. 1, pp. 98–136.

[44] T. Sato, A. Moro, A. Sugahara, T. Tasaki, A. Yamashita, and H. Asama, “Spatio-temporal birds-eye view images using multiple fish-eye cameras,” Proceedings of the 2013 IEEE/SICE International Symposium on System Integration, 2013.

[45] F. Lueetteke, X. Zhang and J. Franke, "Implementation of the Hungarian Method for object tracking on a camera monitored transportation system,” ROBOTIK 2012; 7th German Conference on Robotics, Munich, Germany, 2012, pp. 1-6.

[46] Z. Cao et al., “Realtime multi-person 2D pose estimation using part affinity fields,” in Proc. CVPR, 2017, pp. 1302–1310.

[47] M. Mozaffari, W. Saad, M. Bennis and M. Deb, "Mobile Unmanned Aerial Vehicles (UAVs) for Energy-Effective Internet of Things Communications," in IEEE Transactions on Wireless Communications, vol. 16, no. 11, pp. 7574-7589, Nov. 2017, doi: 10.1109/TWC.2017.2751045.

[48] B. Sahbani and W. Adiprawita, "Kalman filter and Iterative-Hungarian Algorithm implementation for low complexity point tracking as part of fast multiple object tracking system," 2016 6th International Conference on System Engineering and Technology (ICSET), Bandung, 2016, pp. 109-115, doi: 10.1109/ICSEngT.2016.7849633.

[49] A. Yilmaz, O. Javed, and M. Shah, “Object tracking: A survey,” ACM Comput. Surv., vol. 38, no. 4, p. 13, 2006.
and P. H. S. Torr, “Fully-convolutional Siamese networks for object tracking,” in Lecture Notes in Computer Science, Cham: Springer International Publishing, 2016, pp. 850–865.

[51] Y. Chen, X. Yang, B. Zhong, S. Pan, D. Chen, and H. Zhang, “CNNTracker: Online discriminative object tracking via deep convolutional neural network,” Appl. Soft Comput., vol. 38, pp. 1088–1098, 2016.

[52] U. R. Jeereddy, “Pedestrian detection on dewarped fish-eye images using deep neural networks,” 2019.

[53] H. W. Kuhn, “The Hungarian method for the assignment problem,” Nav. Res. Logist., vol. 52, no. 1, pp. 7–21, 2005.

[54] X. Li, K. Wang, W. Wang, and Y. Li, “A multiple object tracking method using Kalman filter,” The 2010 IEEE International Conference on Information and Automation, Jul. 2010.

[55] Y. t. Chan, A. g. c. Hu and J. B. Plant, “A Kalman Filter Based Tracking Scheme with Input Estimation,” in IEEE Transactions on Aerospace and Electronic Systems, vol. AES-15, no. 2, pp. 237-244, March 1979, doi: 10.1109/TAES.1979.308710.

[56] A. Salhi, and A. Y. Jammoussi, “Object tracking system using Camshift, Meanshift and Kalman filter,” World Academy of Science, Engineering and Technology, vol. 64, pp. 674-679, 2012.

[57] S.-K. Weng, C.-M. Kuo, and S.-K. Tu, “Video object tracking using adaptive Kalman filter,” J. Vis. Commun. Image Represent., vol. 17, no. 6, pp. 1190–1208, 2006.

[58] J.W. Tukey. Exploratory Data Analysis. Addison–Wesley, 1977.

[59] L. T. Truong, M. Sarvi, G. Currie and T. M. Garoni, "How Many Simulation Runs are Required to Achieve Statistically Confident Results: A Case Study of Simulation-Based Surrogate Safety Measures," 2015 IEEE 18th International Conference on Intelligent Transportation Systems, Las Palmas, 2015, pp. 274-278, DOI: 10.1109/ITSC.2015.54.

[60] J. D. Wiegand and C. Y. D. Yang, “Traffic simulation runs: How many needed?,” Public Roads, vol. 74, no. 4, 2011.

Mohammad Alhaddad received a B.Sc degree in mechatronics engineering from Eastern Mediterranean University in 2017. He is with the ISSD image processing team since 2018 with a focus on machine intelligence, software development, and design aspects of machine learning-based projects. He has over 3 years of research experience in the sustainability aspects of IoT systems and has worked on a Tunnel surveillance deep learning empowerment project funded by the EU in 2019-2020.

Ceylan Yozgatlıgil received the M.Sc. degree from the Department of Statistics, METU, in 1999, and the Ph.D. degree in Statistics from the Temple University in 2007. She is currently an Associate Professor in the Department of Statistics at METU. Her research interests include time series analysis, temporal aggregation of time series, applied statistics, supervised learning via machine learning algorithms, statistical applications in meteorological variables, temporal clustering methods, statistical inference.

Murat Tulgac received the B.S. degree in electrical and electronics engineering from Anadolu University, Eskişehir, Turkey, in 2018. He is currently pursuing the M.S. degree in electrical and electronics engineering with TOBB ETU, Ankara, Turkey. He is also working as Data Scientist at ISSD. His research interests include machine learning, time series analysis, and distributed computing.

Enes Yüncü received the B.S. degree in electrical and electronics engineering and the M.S. degree in cognitive science from the METU, Ankara, Turkey, in 2010 and 2013, respectively. He is currently working as Technical Lead at ISSD. His research interests include deep learning, computer vision, and machine learning.