Unsupervised detection of inflows and outflows in intersection traffic videos

Integrated Computer-Aided Engineering
XX(X):1–16

DOI: 10.1177/ToBeAssigned
www.sagepub.com/
SAGE

Ariadna Jiménez-Partinen[1,2,3], José D. Fernández-Rodríguez[1,2,3], Jesús Benito-Picazo[1,2,3], Miguel A. Molina-Cabello[1,2,3], Rafaela Benítez-Rochel[2,3] and Ezequiel López-Rubio [1,2,3]

Abstract

As traffic cameras become prevalent, the automatic analysis of traffic scenes presents new opportunities and challenges. Advances in deep learning allow for automated characterization of traffic in such videos. This work aims to understand traffic flow without human supervision, focusing on the localization of road intersections. For this purpose, a three-stage method is proposed that uses a deep neural network for vehicle detection, an object tracker to recover vehicle trajectories, and unsupervised machine learning to detect potential incoming and outgoing traffic flows. The approach has been tested on a variety of real and synthetic videos, with satisfactory results across different camera positions, traffic patterns, and weather conditions. As a key part of the methodology, five options for clustering starting and ending track points were tested. These options included a basic strategy based on predefined spatially localized clusters, and the K-means algorithm with two methods to determine the optimal number of clusters: the Elbow method and the Silhouette score. Additionally, Mean Shift and the Density-Based Spatial Clustering of Applications with Noise were evaluated. An exhaustive analysis of the proposed clustering methods was conducted, including runtime at each stage, performance metrics , and the addition of noise to simulate tracker failures. The results demonstrated the feasibility of the proposed methodology and concluded that Mean Shift is the most suitable clustering method due to its balance of high performance, low runtime, and stable behaviour against abnormal trajectory points.

Keywords

Unsupervised learning, Object tracking, Object detection, Video surveillance, Deep learning

1 Introduction

Technological improvements in video cameras, massive and fast computer storage, and network connectivity have enabled the deployment of ever larger amounts of traffic cameras for video surveillance, monitoring, and collection of data, both linear road segments and, more typically, in intersections and roundabouts (1; 2; 3). Systems for traffic surveillance are deployed for a range of goals, including recording and monitoring how vehicles use and share the road network (to identify antisocial and/or dangerous behaviors), detecting and predicting congestion, collisions, and other traffic-altering anomalies, and collecting statistics about traffic flow. Research in traffic surveillance can be classified in various areas, such as vehicle detection (4; 5), scene analysis (6; 7), traffic tracking (8; 9; 10) and monitoring (11; 12), detection (13) and management of emergency situations (14; 15), detection of traffic events (16; 17), detection of empty parking slots (18), etc.

Artificial intelligence, and especially deep learning, is a technology that is currently accelerating the automation of many kinds of tasks across many disciplines and industries, especially processes that require unsupervised, automated detection with reduced labelling, such as detection of screen printing defects (19) and lightning prediction (20). In the context of traffic surveillance, the application of deep neural networks, mainly convolutional neural networks (CNNs), to the large (and growing) amount of traffic video data enables the implementation of automated intelligent systems to monitor and predict traffic flows.

In many cases, the design of intelligent systems for traffic applications is relatively straightforward for video feeds from linear road segments, but becomes significantly more challenging in urban scenarios, such as traffic intersections and roundabouts. Vehicle detection and tracking is more difficult in these scenarios, since the vehicles have more complex spatio-temporal trajectories, with frequent accelerations and slowdowns, waiting periods to cross intersections or at traffic lights, frequent lane changes and turnings. In these urban settings, vehicles are also frequently hidden from the camera viewpoint by other vehicles, trees

[1]ITIS Software, University of Málaga, C/ Arquitecto Francisco Peñalosa, 18, Málaga, 29071, Spain
[2]Instituto de Investigación Biomédica de Málaga y Plataforma en Nanomedicina-IBIMA Plataforma BIONAND, C/ Severo Ochoa, 35, Málaga TechPark, 29590, Spain
[3] Department of Computer Languages and Computer Science, University of Málaga, Bulevar Louis Pasteur, 35, Málaga, 29071, Spain
Email: {ariadna, jpicazo, josedavid}@uma.es, {miguelangel, benitez, ezeqlr}@lcc.uma.es
ORCID for each author:
- Ariadna Jiménez-Partinen: 0000-0002-4668-2946
- José D. Fernández-Rodríguez: 0000-0003-3702-2230
- Jesús Benito-Picazo: 0000-0001-9015-5804
- Miguel A. Molina-Cabello: 0000-0002-8929-6017
- Rafaela Benítez-Rochel: 0000-0001-8500-0488
- Ezequiel López-Rubio: 0000-0001-8231-5687

or buildings, either partially or completely, making traffic tracking even more challenging (21).

Recently, some researchers have applied deep learning to overcome specific difficulties in traffic analysis at intersections. For example, Abdeljaber *et al.* (28) developed a CNN-based tool for the automatic extraction of traffic trajectories at road intersections, and Tak *et al.* (29) applied YOLOv4 (a deep learning model) for vehicle detection with a camera installed at an intersection. Other researchers have applied clustering to whole traffic trajectories for various tasks, such as the classification of the different types of trajectories in an intersection (30), the inference of traffic rules at an intersection from GPS tracks (31), or tracking of vehicles through multiple traffic cameras with overlapping fields of view (32).

In the context of automatic video analysis at intersections, some recent work has been published. Pan *et al.* (22) introduced the Physics-Guided Spatio-Temporal Graph Neural Network (PG-STGNN) framework, designed to capture complex spatio-temporal dependencies in traffic networks for traffic flow prediction, i.e., flow count. A multi-intersection-aware traffic flow prognostication architecture was proposed by Shen *et al.* (23), which employs recent information from nearby roads using a relevance vector machine (RVM). Jakubec *et al.* (24) implement a YOLO-based framework to automatically analyze traffic flow, including speed and vehicle gaps, within the observed intersection area. Some approaches are composed of ensembling tasks, for instance, Tang *et al.* (25) develop a three-stage system: detection with YOLOv5, tracking with the MS-SORT model, and, finally, integrating trajectories with a re-identification (ReID) method based on ResNet-50 architecture to match trajectories across different surveillance videos, with the aim of analyzing intersection traffic conditions. The work of Azimjonov *et al.* (26) presents a vision-based, real-time traffic-monitoring system for collecting statistics on vehicle movements at intersections. The object-tracker and data-association algorithms use bounding-box properties to estimate vehicle trajectories, aiming to compute vehicle counts and instantaneous and average speeds. Song *et al.* (27) proposed a K-means trajectory clustering method based on NURBS curve fitting to obtain traffic flow parameters. The B-spline quadratic interpolation function is used to fit a smooth NURBS curve to the vehicle trajectory, and the K-means clustering algorithm measures the minimum distance, using the first and last endpoints to automatically divide the intersection area to count the vehicle flow.

In Table 1, the main characteristics of reported studies are summarized. The related works, which focused on intersection video analysis, share some similarities with the present study, such as the implementation of a three-stage method that includes detection and tracking. However, they differ in the final and crucial step, which defines the purpose of the system. While previous studies have primarily focused on predicting vehicle flow, analyzing intersection conditions, and collecting vehicle movement statistics– mostly utilizing private datasets– this work focuses on identifying the entry and exit points at intersections through vehicle tracking points clustering. Some of these studies also incorporate unsupervised approaches, which, alongside the reported results, demonstrate the viability of applying these methods to the addressed problem. Additionally, while many studies are limited to specific locations, the proposed method has been designed for general intersection locations and conditions using open-access datasets. Considering the drawbacks and gaps identified in the literature, the proposed methodology offers an innovative, reproducible approach that, as far as the authors know, has no comparable frameworks.

The methodology presented in this paper is intended to facilitate automated analysis of traffic trajectories at intersections with traffic surveillance cameras by applying deep learning-based methods to reduce human supervision. The proposed three-stage method automatically detects points of traffic inflow and outflow at the borders of the scene watched by a video camera. The first stage involves detecting vehicle positions in each video frame using the YOLOv5x6 model. At the second stage, vehicle trajectories are reconstructed using the Norfair tracking method. Finally, the key task is addressed by applying unsupervised clustering to these trajectories to automatically identify the traffic flows entering and exiting the intersection.

In turn, this enables the analysis of road intersection videos without the need for manual labeling of the roads and traffic lanes visible in the intersection. Further analysis of these videos (not addressed in this work) may be enabled by the automated detection of these points of traffic inflows and outflows.

While this work is focused on unsupervised detection of points of entry and departure of traffic flows in video sequences, we have previously published work on other aspects of the processing of traffic videos, from basic detection and tracking of vehicles (33) to unsupervised clustering of vehicle types (34) and detection of anomalous vehicle trajectories (35).

The rest of this work is organized as follows: Section 2 provides a detailed description of the methodology used for detecting potential incoming and outgoing traffic flows in traffic videos at intersections. Section 3 describes the settings items and the data set used for the experiments and the assessment of the performance of the proposed method. Finally, conclusions are drawn in Section 4.

2 Methodology

The following method is proposed for identifying vehicle entry and exit points in traffic videos of an intersection. Initially, an object detection deep neural network, denoted as $\mathcal{F}$, is used to obtain a collection of object detections, S_t, from the current video frame, $\mathbf{X}_t$, at each time step t:

$$S_t = \mathcal{F}\left(\mathbf{X}_t\right) =$$

$$= \{(a_{t,i}, b_{t,i}, c_{t,i}, d_{t,i}) \mid i \in \{1, ..., N_t\}\} \quad (1)$$

where the total number of object detections is represented by N_t. For the i-th object detected at time t, the upper left corner of its bounding box is denoted by $(a_{t,i}, b_{t,i})$, and the lower right corner by $(c_{t,i}, d_{t,i})$.

Next, the output from the object detection network is fed into a tracking method, $\mathcal{G}$. This method takes the current set of object detections, S_t, along with the previously tracked

Table 1. Summary of studies on automatic video analysis at intersections.

Reference	Task	Method	Dataset	Location
Pael *et al.* (22)	Predict traffic flow (Regression)	Physics-Guided Spatio-Temporal Graph Neural Network (PG-STGNN)	Private	1 location from Yizhuang District of Beijing, China
Shen *et al.* (23)	Short-term Traffic Flow Forecasting (Regression)	Relevance Vector Machine (RVM)	Private	6 locations from Minnesota, EE.UU
Jakubeck *et al.* (24)	Traffic analysis (timestamps, speeds, gaps, and directions)	YOLO + ByteTrack + Traffic analysis module	Private	1 location from Zilina, Slovak Republic
Tang *et al.* (25)	Long-term traffic conditions (trajectories integration)	YOLO + MS-SORT + ResNet-50	Private	2 locations from Yiyang city, Hunan, China Province.
Azimijov *et al.* (26)	Real-time vehicle movements statistics (total counting, instantaneous, and average speed)	YOLO + Own tracker + data association module	Available	5 locations from the Netherlands, Sweden, Turkey, Japan, and Ukraine
Song *et al.* (27)	Vehicle flow counting	NURBS (Non-Uniform Rational B-Splines) curves + K-means	Private	3 unknown locations
Our proposal	Unsupervised inflows and outflows detections	YOLO + Norfair + Clustering method	Open-access	14 general locations

objects, Q_{t-1}, and produces the updated set of tracked objects, Q_t:

$$Q_t = \mathcal{G}\left(S_t, Q_{t-1}\right) = \\ = \{(\alpha_{t,j}, \beta_{t,j}, \gamma_{t,j}) \mid j \in \{1, ..., M_t\}\} \quad (2)$$

where the total number of object tracks is denoted by M_t. The centroid of the j-th tracked object at time t is represented by $(\alpha_{t,j}, \beta_{t,j})$, and $\gamma_{t,j}$ is an integer that uniquely identifies the tracked object.

At the start of the video, there are no tracked objects, so the set is initially empty:

$$M_0 = 0, \; Q_0 = \emptyset \quad (3)$$

The set R_t includes all tracked objects from the beginning of the video up to the current time t:

$$R_t = \bigcup_{\tau=1}^{t} Q_\tau \quad (4)$$

Then, the set V_t, which contains the first and last occurrences of all tracked objects up to the current time t, is calculated:

$$V_t = \{(\alpha_{\tau,j}, \beta_{\tau,j}, \gamma_{\tau,j}) \in R_t$$

$$|\tau = \min\{\tau' \mid (\alpha_{\tau',j}, \beta_{\tau',j}, \gamma_{\tau',j}) \in R_t\} \cup$$

$$\{(\alpha_{\tau,j}, \beta_{\tau,j}, \gamma_{\tau,j}) \in R_t$$

$$|\tau = \max\{\tau' \mid (\alpha_{\tau',j}, \beta_{\tau',j}, \gamma_{\tau',j}) \in R_t\} \quad (5)$$

Subsequently, the clustering method is applied for clustering the set V_t. In the case of the K-means clustering algorithm, the Elbow method is applied to determine the optimal number of clusters, k. The Elbow method identifies the value of k corresponding to the point of maximum curvature on the Mean Quantization Error (MQE) curve as a function of k:

$$MQE_{k,t} = \\ = \frac{1}{|V_t|} \sum_{(\alpha_j, \beta_j, \gamma_j) \in V_t} \min_{h \in \{1,...,k\}} \left\| (\alpha_j, \beta_j) - \left(\bar{\alpha}_{h,k}, \bar{\beta}_{h,k}\right) \right\|^2 \quad (6)$$

where $|\cdot|$ represents the cardinality of a set, $\|\cdot\|$ denotes the Euclidean norm of a vector, and $\left(\bar{\alpha}_{h,k}, \bar{\beta}_{h,k}\right)$ is the center of the h-th cluster derived by the k-means algorithm for a total of k clusters. Please note that the elbow of the curve is found without introducing any kind of visual assessment into the procedure, but algorithmically finding the point of maximum curvature as it is described in Equation 6. See Figure 1 for a visual example.

Let us denote the value chosen by the Elbow method as $\hat{k}$. The corresponding set of cluster centers is then given by:

$$C_t = \left\{ \left(\hat{\alpha}_h, \hat{\beta}_h\right) \mid h \in \left\{1, ..., \hat{k}\right\} \right\} \quad (7)$$

The set C_t is viewed as a compact representation of the observed centroids within R_t.

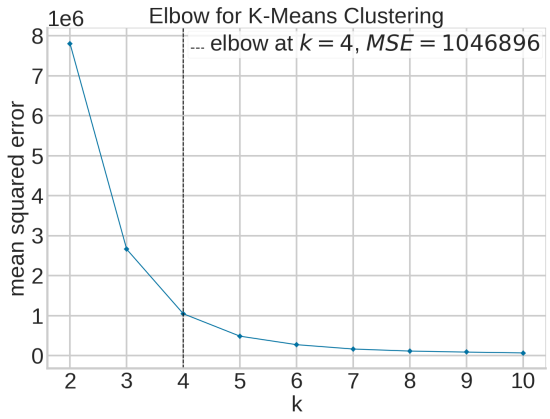

Figure 1. Determining the number of clusters using the elbow method for the video Seq1_SK_1 processed with YOLOv5x6. For each possible number of clusters k, the mean squared error (MSE) is computed, and the point of maximum curvature is chosen as the best number of clusters. This graph is purely illustrative: the elbow point is algorithmically computed as the point of maximum curvature, without any kind of visual assessment.

When vehicle detection and tracking maintain sufficiently low error rates, the cluster centroids in C_t indicate where vehicles enter and exit the camera's view.

3 Experimental Results

This section presents the results obtained by applying the previously described methodology in a series of experiments with various videos of road intersections from the perspective of a traffic camera.

3.1 Methods

The system is implemented using *OpenCV* to read image frames from recorded videos or, if available, to retrieve a live stream from a traffic camera. For vehicle detection, the deep learning network YOLOv5 is used. Specifically, of the range of publicly available sub-models, YOLOv5x6 is used. This sub-model is trained on the COCO dataset (36) and achieves an mAP@0.5 of 72.7 (37). Objects of COCO classes *car*, *motorcycle*, and *truck* are considered to be vehicle detections. Any other detections are discarded. Vehicle detections are fed to Norfair, a publicly available state-of-the-art object tracker with standard assignment heuristics and Kalman filters (38). Default parameters are used for these components. It should be noted that this method is best suited when the point of view of the camera is high enough to minimize vehicle-vehicle occlusions that might result in tracking failures.

For each vehicle trajectory, the starting and ending points are retrieved and clustered using three state-of-the-art clustering algorithms: K-means (39), Mean Shift (40; 41), and Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise (DBSCAN) (42) employing the SCIKIT-LEARN (43) Python library. Additionally, an alternative method based on split frames into quadrants (Quadrant-based) has been included to compare results between a simple spatial-rule-based method and an unsupervised method, which involves methodological complexity and higher computational cost. The Quadrant-based strategy divides the frame into four parts, each corresponding to a predefined cluster. The choice of a 2×2 grid is due to the common placement of the intersection in the picture.

In the case of the K-means algorithm (42), two methods are used in order to estimate the optimal number of clusters: the Elbow method and the Silhouette Score. The Elbow method consists of repeatedly computing clustering with different cluster numbers, measuring clustering performance (see Section 2 for details), and selecting the point of maximum curvature in the clustering metric. However, the Silhouette score is a distance-based criterion that selects the cluster number that maximizes the Silhouette coefficient, a normalized difference between the intra-cluster distance and the nearest inter-cluster distance. Please note that the other two clustering algorithms (Mean Shift and DBSCAN) do not require specifying the number of clusters.

The Mean Shift clustering algorithm is an iterative method aiming to discover "blobs" in a smooth density of samples (41). This algorithm identifies clusters of arbitrary shape and variable size without requiring a priori specification of the number of clusters. It is a centroid-based algorithm, which works by updating candidates for centroids to be the mean of the points within a given region by using a kernel function that determines the weight of nearby points for re-estimation of the mean. The implementation used in this work is from the Scikit-learn package for Python (44).

Finally, the Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise or DBSCAN, is a clustering method that relies on a density-based notion of clusters, which is designed to discover clusters of arbitrary shape (42). The DBSCAN algorithm has 2 hyperparameters to be adjusted in order to get significant results. The first one is the $epsilon$ parameter, which defines the radius of a circular (or spherical, in higher dimensions) neighborhood around each point. In our study, we use the method described by Ester *et al.* (42) to estimate $epsilon$, which involves applying the Elbow method to the K-nearest neighbours algorithm. The second parameter that can be adjusted is the $Min\ points$ parameter which is the number of points a core point must have within its $epsilon$ neighbourhood. In (42) is demonstrated how the optimal value for this parameter in bi-dimensional distributions is 4. Therefore, *Min points* has been adjusted to 4 for the experiments carried out.

The goal of this clustering stage is to obtain cluster centroids that mark the positions of roads and/or road lanes at a road intersection. The aim of using all these methods – Quadrant-based, K-means (Elbow), K-means (Silhouette), Mean Shift, and DBSCAN – is to compare clustering performance and to validate the three-stage methodology proposed using several strategies. The overall proposed methodology is illustrated in Figure 2.

Please note that all values described in this pipeline (also in the formal description in Section 2) are in pixel space (from vehicle bounding boxes to bounding boxes' centers, tracking and clustering of starting and ending points), without translating them to non-distorted geometric coordinates over the road plane. This means that, depending on the lens geometry and the camera pose, pixel distances

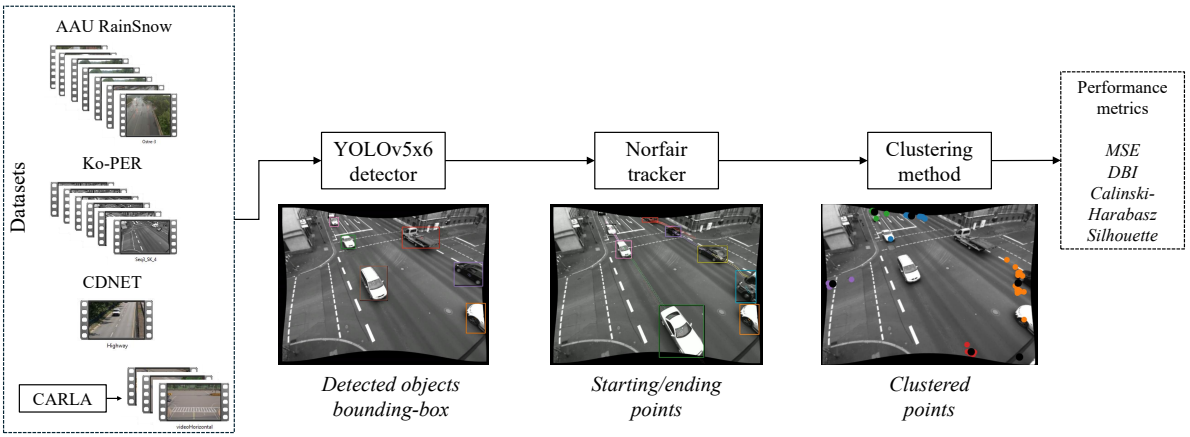

Figure 2. Methodology pipeline.

of similar magnitude in different parts of the scene will actually represent very different real distances over the road plane. This will influence clustering results, as portions of the road that are far away from the camera (on in greatly compressed areas of the image plane) will naturally present different clustering from similar areas that are near the camera. However, this work is mostly concerned with unsupervised detection of entry/exit points in traffic scenes in qualitative terms. For this, clustering in the pixel space presents reasonably good results. Furthermore, detecting both arbitrary camera poses (relative to the road) and arbitrary lens geometries, and undoing them effectively, consistently and without errors, is very difficult unless using supervised methods, like the one used by García-González et al. (45). In contrast, working in pixel space can be effective even when measuring relative distances and velocities between trajectories, as shown by Fernández-Rodríguez et al. (35). For these reasons, and noting that we are more concerned with qualitative results about entry/exit points in traffic scenes, in our methodology we use coordinates in pixel space.

3.2 Datasets

To evaluate the proposed methods, a diverse set of videos from several well-established open-access datasets has been utilized. A total of 14 videos have been chosen, those videos whose scenes met the criteria in which the model was implemented, from three state-of-the-art datasets depending on the purpose. To validate the proposed method in bad weather conditions, seven videos from the AAU RainSnow Traffic Surveillance Dataset (46) were chosen: Hadsundvej-1, Hadsundvej-2, Hasserisvej-1, Hasserisvej-2, Hasserisvej-3, Hjorringvej-2, and Ostre-3. For each intersection, all videos are from the same camera, but were captured at different times and under varying weather conditions, including rain and snow.

To test the proposed method with different viewing angles of the same vehicles, another six videos were taken from the Ko-PER Intersection dataset (47): Seq1_SK_1, Seq1_SK_4, Seq2_SK_1, Seq2_SK_4, Seq3_SK_1, and Seq3_SK_4. Of these, videos whose names differ only in the ending number were recorded at the same time in the same intersection, but from cameras in different places and orientations. All videos depict four-way intersections, but some recordings do not include incoming/outgoing vehicles in one of the four ways.

Finally, to check the behaviour of the model out of the scope conceived, road intersections, a non-branching road segment video was added: the video titled *Highway* from the 2014 CDNET dataset (48). Additionally, three videos were generated using CARLA (49), each showcasing different perspectives of the same three-way intersection. These synthetic videos were designed to feature scenarios such as speeding, tailgating, and unnecessary lane changes amid heavy traffic, in order to test the robustness of the proposed method when applied to videos depicting vehicles engaged in hazardous, unconventional behaviours.

3.3 Performance Metrics

Given that our study explores five distinct clustering methods (Quadrant-based, K-means (Elbow), K-means (Silhouette), Mean Shift, and DBSCAN), it is essential to provide a thorough and structured presentation of the experimental results to adequately capture the unique characteristics and performance of each approach. Clustering algorithms often exhibit significant differences in how they handle data, particularly in terms of how they define clusters, their sensitivity to parameters, and their ability to manage noise or outliers. These variations are likely to influence the outcomes of our experiments, and as such, warrant individual attention and detailed analysis.

Given that the key part of the methodology proposed relies on the clustering task, it is essential to consider the clustering performance measures obtained from the underlying clustering processes. These measures provide critical insights into the quality and effectiveness of the clustering methods used, helping to validate and interpret the results. By evaluating the performance of the clustering processes, we can better assess the reliability and significance of the conclusions drawn from the data. Therefore, several well-known clustering performance measures have been considered:

- Mean Squared Error (MSE) quantifies the compactness of clusters by measuring the average squared distance between points and their respective centroids, with lower values indicating tighter, more coherent clusters.

- Davies-Bouldin Index (DBI) (50) assesses the separation between clusters by considering both intra-cluster similarity and inter-cluster differences, with lower values signaling better-defined clusters.
- The Calinski-Harabasz index (50) measures the ratio of between-cluster dispersion to within-cluster dispersion, where higher values indicate better clustering structure.
- The Silhouette score (51) gauges the separation between clusters by comparing the average intra-cluster distance with the nearest inter-cluster distance, providing insight into how well clusters are formed. The higher the score, the better the clustering.

Together, these performance measures offer a comprehensive understanding of clustering quality, enabling more robust and reliable conclusions in any clustering study. Furthermore, the required runtimes have been recorded in order to take computation time into account as a key constraint for validating and supporting the methodology in real-world scenarios.

3.4 Results

To facilitate a comprehensive and nuanced comparison of these methods, the experimental results section will be addressed by performance metrics, allowing thorough comparison of each strategy and ensuring that each method's results are presented in a manner that supports careful reflection on its strengths, limitations, and overall effectiveness. This structure not only enhances clarity but also provides the necessary context for understanding the conditions under which each method excels or faces challenges, thereby offering valuable insights into their comparative performance.

Table 2. For each video (first column), number of frames (second column) and per-frame milliseconds spent in detection (YOLOv5x6, third column) and tracking (Norfair, fourth column).

Video	Frames	Detection	Tracking
CARLA1	3839	37.068	0.684
CARLA2	3900	37.150	0.707
CARL32	3629	37.084	0.507
Highway	1700	37.135	0.428
Seq1_SK_1	2419	37.035	1.177
Seq1_SK_4	2419	36.816	0.594
Seq2_SK_1	947	37.225	1.528
Seq2_SK_4	947	37.044	0.622
Seq3_SK_1	1020	37.084	1.516
Seq3_SK_4	1020	37.122	0.925
Hadsundvej-1	4800	42.381	1.236
Hadsundvej-2	6000	42.423	1.845
Hasserisvej-1	6000	42.432	0.438
Hasserisvej-2	5998	42.684	0.540
Hasserisvej-3	5998	42.599	0.650
Hjorringvej-2	5998	42.589	1.101
Ostre-3	5998	42.4242	0.264
Mean	3712.47	39.558	0.916
Std	1948.6	2.663	0.418

The proposal is a three-stage methodology – detection, tracking, and clustering – where the key part is the clustering task, which, in addition to detecting and following vehicles, gives their tracks meaning, in this case, unsupervised detection of inflows and outflows. To ensure the real-time feasibility of the methodology, runtimes are computed by stages. In Table 2, the detection and tracking execution time (ms) per frame for each video is reported together with the mean and standard deviation (Std). While the detection task takes between 37 and 42 ms, the tracking task is too fast, taking around 1 ms to track and return the starting and ending points of the vehicle's follow. Independent of the video nature (high-traffic or lonely intersections), the two stages require less than a second.

In Table 3, the number of clusters estimated by each strategy is reported. It can be noted that DBSCAN yields more spread values, while K-means with the Elbow method shows the least change in the number of clusters between 3 and 5, except for Quadrant-based, which, due to its implementation, can distribute points into 4 clusters. However, if in the frame there is a quadrant without points, the number of clusters counted is 3 instead of 4. Figure 3 depicts some samples of how these clusters are distributed to ensure a qualitative inspection. The samples shown, from left to right, are the first frame of CARLA1, one of the three videos synthesized using CARLA, Hasserisvej-1 from the AUU RainSnow Dataset, and Seq1_SK_1 from the Ko-PER dataset. The Seq1_SK_1 first frame (third column) shows how the Quadrant-based strategy works (first row): assigning a point to the cluster and splitting the image into a 2×2 grid. In this case, the clustering makes sense according to the intersection. However, in the CARLA1 and Hasserisvej-1 videos, this approach is unsuitable because of its lack of adaptation to the scene. Whereas clustering methods give the methodology the ability to adapt to the position of the camera and the shape of the intersection, for example, be indifferent to three and four-way intersections. In this sample, the K-means algorithm correctly clusters the entry and departure roads, although in Hasserisvej-1 (second column), the points corresponding to the right road (obscured by a large tree) are not differentiated by K-means alternatives, nor by Mean Shift. When the Silhouette score is maximized in K-means, the number of clusters increases due to the criterion to minimize the intra-cluster distance, as can be observed in the Seq1_SK_1 video (third column) when comparing the second and third rows. In the samples shown, the Mean Shift and DBSCAN methods retrieved considerably more clusters (and noise points in the case of DBSCAN). In the CARLA1 frame, clustered by DBSCAN, it can be seen that the three lanes at the left upper and right bottom corners are associated with different clusters, while points located more centrally around the intersection are unclustered and assigned as noise (points colored in olive green). These points are commonly associated with the location of the vehicles at the beginning of the recording, and it is interesting to note how the clustering method addresses them, as it is a way to filter outliers that do not follow the complete path in sequences . Only DBSCAN leaves out these points, while the others assign them to a cluster (increasing its size) or create a new one.

Table 3. Number of clusters obtained from the clustering process executed as part of the traffic flow detection method proposed.

Video	Quadrant-based	K-means (Elbow)	K-means (Silhouette)	Mean Shift	DBSCAN
CARLA1	4	3	3	5	8
CARLA2	4	4	10	6	9
CARLA3	3	3	3	6	7
Highway	4	3	3	5	5
Seq1_SK_1	4	4	7	7	8
Seq1_SK_4	4	4	6	7	5
Seq2_SK_1	4	4	5	6	9
Seq2_SK_4	4	4	5	6	8
Seq3_SK_1	4	4	6	6	9
Seq3_SK_4	4	4	5	6	9
Hadsundvej-1	4	4	10	5	11
Hadsundvej-2	4	4	10	5	16
Hasserisvej-1	3	4	4	5	7
Hasserisvej-2	3	5	4	4	9
Hasserisvej-3	3	5	3	5	8
Hjorringvej-2	4	4	3	4	15
Ostre-3	3	4	8	4	7

Table 4. Mean Squared Error (MSE) obtained from the clustering process executed as part of the traffic flow detection method proposed. For each video (row), the best value among the clustering methods is highlighted in bold. The lower the better.

Video	Quadrant-based	K-means (Elbow)	K-means (Silhouette)	Mean Shift	DBSCAN
CARLA1	9956.217	12350.548	12350.548	**6770.168**	114126.938
CARLA2	13000.487	8339.266	**627.041**	5462.168	23825.615
CARLA3	4521.686	4521.686	4521.686	**1206.038**	11981.324
Highway	10530.192	4979.555	4979.555	**2078.328**	45435.673
Seq1_SK_1	13436.413	13170.683	**3783.763**	3876.677	107545.304
Seq1_SK_4	11095.887	9032.155	3140.652	**2586.105**	50539.211
Seq2_SK_1	17326.015	10449.417	7249.272	**5180.770**	12826.112
Seq2_SK_4	24582.805	6326.696	2706.947	**2071.361**	36121.573
Seq3_SK_1	15509.273	15509.273	**5720.161**	8138.330	56139.482
Seq3_SK_4	18664.424	13070.108	9077.905	**6577.037**	45179.618
Hadsundvej-1	4510.452	2636.424	**576.047**	2526.784	9312.325
Hadsundvej-2	4634.410	2862.392	**530.749**	3178.662	16391.929
Hasserisvej-1	3477.890	756.543	756.543	**679.731**	1587.225
Hasserisvej-2	3188.378	**298.561**	872.436	872.436	3684.462
Hasserisvej-3	2477.752	**260.617**	1304.901	889.434	1331.422
Hjorringvej-2	4984.974	**2353.246**	4514.143	2499.075	6778.097
Ostre-3	5494.156	1278.395	**256.893**	1858.866	3620.832
Mean	9846.554	6364.445	3704.073	**3320.704**	32142.773
Std	6294.529	4955.040	3327.339	**2229.811**	33826.149

To quantitatively analyze clustering performance, the results obtained from the metrics for each clustering method across all videos are reported and discussed below. Table 4 presents the Mean Squared Error (MSE) achieved by each clustering method. The Quadrant-based strategy and DBSCAN algorithm yielded the highest MSE values, while K-means and Mean Shift achieved the lowest, suggesting poor clustering performance in terms of fit. Specifically, optimizing the Silhouette score to estimate the optimal number of clusters produced lower MSE values more frequently than the Elbow method. The difference in the K-means means across the two approaches—Elbow and Silhouette—arises from videos with different numbers of clusters. In these cases, a larger number of clusters (as shown in Table 3) reduced the MSE by decreasing the distance between the centroid and the points within the cluster; for this reason, the lowest value achieved (256.893 with Ostre-3) is with K-means (Silhouette). Finally, the Mean Shift algorithm achieved the lowest MSE values in 8 out of 17 videos within the dataset. It reported the lowest mean while requiring fewer clusters than K-means (using Silhouette) or DBSCAN. This can be interpreted as a more effective distribution of clusters around the starting and ending points of the tracks.

The Davies-Bouldin Index (DBI) is the other performance metric to minimize; it improves by increasing separation between clusters and decreasing variation within clusters. The outcomes across clustering proposals are reported in Table 5. In this case, the lowest value per video is more spread, with DBSCAN showing the highest values, followed

Figure 3. Samples of the first frame, where the initial and final points for each detected vehicle trajectory are overlaid on the frame. The columns, from left to right, represent the CARLA1, Hasserisvej-1, and Seq1_SK_1 videos, respectively. Each row corresponds to a different clustering method. Points that belong to the same cluster (as determined by the clustering method) are shown in the same color, while cluster centroids are marked in black.

by Quadrant-based, whereas K-means and Mean Shift report more similar values. It is worth noting that the Quadrant-based strategy has the CARLA3 masked as the lowest value (along with the two K-means options) with a DBI of 0.110, because it considers 3 clusters in this video. This outstanding result is a consequence of the lack of points in one quadrant of the image; it is due to the distribution of the points, not the approach itself. In this case, the K-means algorithm optimized with the Silhouette score obtained better (lower) results than the Elbow method. The Elbow method yields the same outcome only when the K value matches that from the Silhouette approach, as in the CARLA1 and CARLA3 videos. The DBI obtained by K-means (Silhouette) with a mean and standard deviation of 0.318 ± 0.110, while Mean Shift had retrieved 0.351 ± 0.148. Across the dataset, DBI K-means achieved the lowest values more frequently, occurring 11 times compared to 7 times for Mean Shift.

The Calinski-Harabasz index can be interpreted as indicating that well-defined clusters have a large variance between clusters and a small variance within clusters. According to the values obtained, reported in Table 6, the approach that maximizes the Calinski-Harabasz index is K-means (Silhoutte), which achieved the highest mean value of 2101.099 and performed best 8 times across all videos. This is followed by the Mean Shift method and the K-means method using the Elbow technique, which achieved the highest values 7 and 5 times, respectively. In contrast,

Table 5. Davies-Bouldin Index (DBI) obtained from the clustering process executed as part of the traffic flow detection method proposed. For each video (row), the best value among the clustering methods is highlighted in bold. The lower the better.

Video	Quadrant-based	K-means (Elbow)	K-means (Silhouette)	Mean Shift	DBSCAN
CARLA1	0.215	**0.128**	**0.128**	0.280	0.633
CARLA2	0.302	0.374	**0.296**	0.490	0.939
CARLA3	**0.110**	**0.110**	**0.110**	0.120	0.718
Highway	0.341	0.233	0.233	**0.162**	0.802
Seq1_SK_1	0.367	0.376	**0.328**	**0.328**	0.897
Seq1_SK_4	0.374	0.392	0.277	**0.199**	1.223
Seq2_SK_1	0.470	0.415	**0.377**	0.404	0.864
Seq2_SK_4	0.658	0.265	0.275	**0.193**	1.081
Seq3_SK_1	0.409	0.409	**0.286**	0.336	0.624
Seq3_SK_4	0.557	0.524	0.476	**0.414**	0.790
Hadsundvej-1	0.671	0.483	**0.454**	0.518	1.427
Hadsundvej-2	0.632	0.492	**0.477**	0.609	1.107
Hasserisvej-1	0.551	0.288	0.288	**0.206**	0.876
Hasserisvej-2	0.472	0.250	**0.228**	**0.228**	0.932
Hasserisvej-3	0.419	**0.242**	0.297	0.432	0.996
Hjorringvej-2	0.560	0.489	**0.478**	0.485	0.872
Ostre-3	0.802	0.425	**0.391**	0.569	0.905
Mean	0.465	0.347	**0.318**	0.351	0.923
Std	0.171	0.123	**0.110**	0.148	0.198

Table 6. Calinski-Harabasz obtained from the clustering process executed as part of the traffic flow detection method proposed. For each video (row), the best value among the clustering methods is highlighted in bold. The higher the better.

Video	Quadrant-based	K-means (Elbow)	K-means (Silhouette)	Mean Shift	DBSCAN
CARLA1	2865.495	**3462.808**	**3462.808**	3225.945	694.355
CARLA2	1531.665	2425.423	**10700.693**	2470.792	586.805
CARLA3	8510.273	8510.273	8510.273	**12710.396**	2324.966
Highway	314.664	1047.715	1047.715	**1235.552**	265.047
Seq1_SK_1	700.874	715.876	**1268.408**	**1268.408**	74.233
Seq1_SK_4	461.888	574.278	1001.929	**1016.524**	138.332
Seq2_SK_1	361.319	619.278	672.256	**782.817**	272.549
Seq2_SK_4	104.576	467.897	828.425	**869.816**	50.569
Seq3_SK_1	351.589	351.589	**582.242**	408.218	149.789
Seq3_SK_4	124.132	184.968	201.862	**233.204**	37.745
Hadsundvej-1	174.427	325.425	**513.254**	257.627	70.139
Hadsundvej-2	480.394	840.110	**1624.751**	676.688	110.147
Hasserisvej-1	236.903	**930.525**	**930.525**	779.604	446.893
Hasserisvej-2	392.462	**2596.522**	1142.908	1142.908	377.995
Hasserisvej-3	529.532	**2991.325**	1111.086	853.369	430.558
Hjorringvej-2	945.496	**2159.449**	1591.853	2101.830	502.843
Ostre-3	48.007	234.965	**527.699**	163.555	51.669
Mean	1066.688	1672.849	**2101.099**	1776.309	387.331
Std	1975.898	1992.803	2851.448	2847.044	**524.553**

both the quadrant-based strategy and DBSCAN show no outstanding values.

The Silhouette score outcome by clustering methods is shown in Table 7. The first fact to highlight is that K-means (Silhouette) retrieved the highest values, ranging from 0.632 to 0.908, indicating good performance with a strong structure. This is to be expected, as K-means with Silhouette is specifically designed to maximize this metric. The Silhouette score measures how compact clusters are, with how well separated they are. Therefore, when the number of clusters increases, as shown in Table 3, often clusters become smaller and more compact, which results in a lower intra-cluster distance that can boost the score. This could explain why K-means with the Elbow method obtained good performance (0.755±0.081), but slightly worse than the Silhouette version (0.786 ± 0.072), whose number of clusters is usually larger. K-means alternatives are followed by Mean Shift, which obtains a slightly lower but still good value (0.730 ± 0.080). However, the results from DBSCAN and Quadrant-based clustering indicate a weaker structure, with scores of 0.578 ± 0.117 and 0.613 ± 0.159, respectively.

Finally, in Table 8 is shown the runtime (ms) employed by each clustering method in the third stage of the methodology:

Table 7. Silhouette score obtained from the clustering process executed as part of the traffic flow detection method proposed. For each video (row), the best value among the clustering methods is highlighted in bold. The higher the better.

Video	Quadrant-based	K-means (Elbow)	K-means (Silhouette)	Mean Shift	DBSCAN
CARLA1	0.816	**0.878**	**0.878**	0.759	0.729
CARLA2	0.765	0.770	**0.863**	0.642	0.806
CARLA3	**0.908**	**0.908**	**0.908**	0.856	0.652
Highway	0.686	**0.868**	**0.868**	0.793	0.715
Seq1_SK_1	0.736	0.739	**0.785**	**0.785**	0.358
Seq1_SK_4	0.759	0.780	**0.809**	0.803	0.697
Seq2_SK_1	0.618	0.737	**0.787**	0.778	0.549
Seq2_SK_4	0.371	0.792	**0.805**	0.793	0.630
Seq3_SK_1	0.718	0.718	**0.794**	0.696	0.698
Seq3_SK_4	0.543	0.689	0.718	**0.723**	0.501
Hadsundvej-1	0.452	0.627	**0.632**	0.575	0.528
Hadsundvej-2	0.514	0.632	**0.681**	0.573	0.503
Hasserisvej-1	0.487	**0.815**	**0.815**	0.779	0.500
Hasserisvej-2	0.579	0.774	**0.806**	**0.806**	0.517
Hasserisvej-3	0.635	0.770	**0.799**	0.698	0.496
Hjorringvej-2	0.537	0.685	**0.699**	0.685	0.487
Ostre-3	0.293	0.652	**0.721**	0.663	0.456
Mean	0.613	0.755	**0.786**	0.730	0.578
Std	0.159	0.081	**0.072**	0.080	0.117

Table 8. Runtime (ms) obtained from the clustering process executed as part of the traffic flow detection method proposed. For each video (row), the best value among the clustering methods is highlighted in bold. The lower the better.

Video	Quadrant-based	K-means (Elbow)	K-means (Silhouette)	Mean Shift	DBSCAN
CARLA1	49.393	70.778	54.327	13.472	**7.249**
CARLA2	61.317	71.659	54.851	13.372	**3.196**
CARLA3	65.581	53.835	56.381	16.143	**9.766**
Highway	56.953	54.095	47.294	15.464	**2.452**
Seq1_SK_1	41.407	50.098	40.087	12.318	**8.176**
Seq1_SK_4	41.477	41.724	49.688	16.427	**2.437**
Seq2_SK_1	51.234	49.479	60.562	16.615	**7.564**
Seq2_SK_4	45.306	43.059	51.286	14.606	**2.209**
Seq3_SK_1	42.387	45.118	46.536	16.100	**3.860**
Seq3_SK_4	47.633	55.856	46.313	19.370	**2.616**
Hadsundvej-1	20.935	60.968	46.991	12.418	**2.579**
Hadsundvej-2	20.217	129.836	120.720	21.000	**6.966**
Hasserisvej-1	13.039	51.509	44.304	14.818	**6.771**
Hasserisvej-2	20.021	54.937	55.705	11.433	**3.306**
Hasserisvej-3	16.679	47.994	57.794	12.743	**3.524**
Hjorringvej-2	19.981	119.094	128.149	31.878	**5.177**
Ostre-3	15.864	57.820	55.174	16.298	**8.190**
Mean	37.025	62.227	59.774	16.146	**5.061**
Std	17.089	24.163	24.222	4.624	**2.468**

cluster starting and ending points retrieved by the first (detecting) and the second (tracking) steps. These results are considered a key metric due to the importance of solving in real-time scenarios. In terms of runtime, the fastest clustering approach is DBSCAN (5.061 $\pm$ 2.468), whose maximum runtime recorded is 9.766 ms, while the next-fastest method is Mean Shift (16.146 $\pm$ 4.624), with a minimum runtime of 11.433 ms. They are followed by Quadrant-based (37.025 $\pm$ 17.089), K-means (Silhouette) (59.774 $\pm$ 24.222), and K-means (Elbow) (62.227 $\pm$ 24.163). The main difference between the recorded runtimes lies in how each algorithm handles clustering. DBSCAN is a non-iterative process that calculates the *epsilon* parameter, which establishes the threshold for neighbor distance, and the *Min points*, to define clusters. However, both Mean Shift and K-means are iterative methods, with K-means also involving the estimation of the optimal number of clusters.

Due to the complexity of decision-making, we aim to compare clustering methods by considering performance metrics: MSE, DBI, Calinski-Harabasz index, Silhouette score, and clustering runtime as key factors to optimize. In this multi-objective context, where no method achieves all objectives, it is important to make trade-offs among them. For this reason, a Pareto front, Figure 4, is generated to

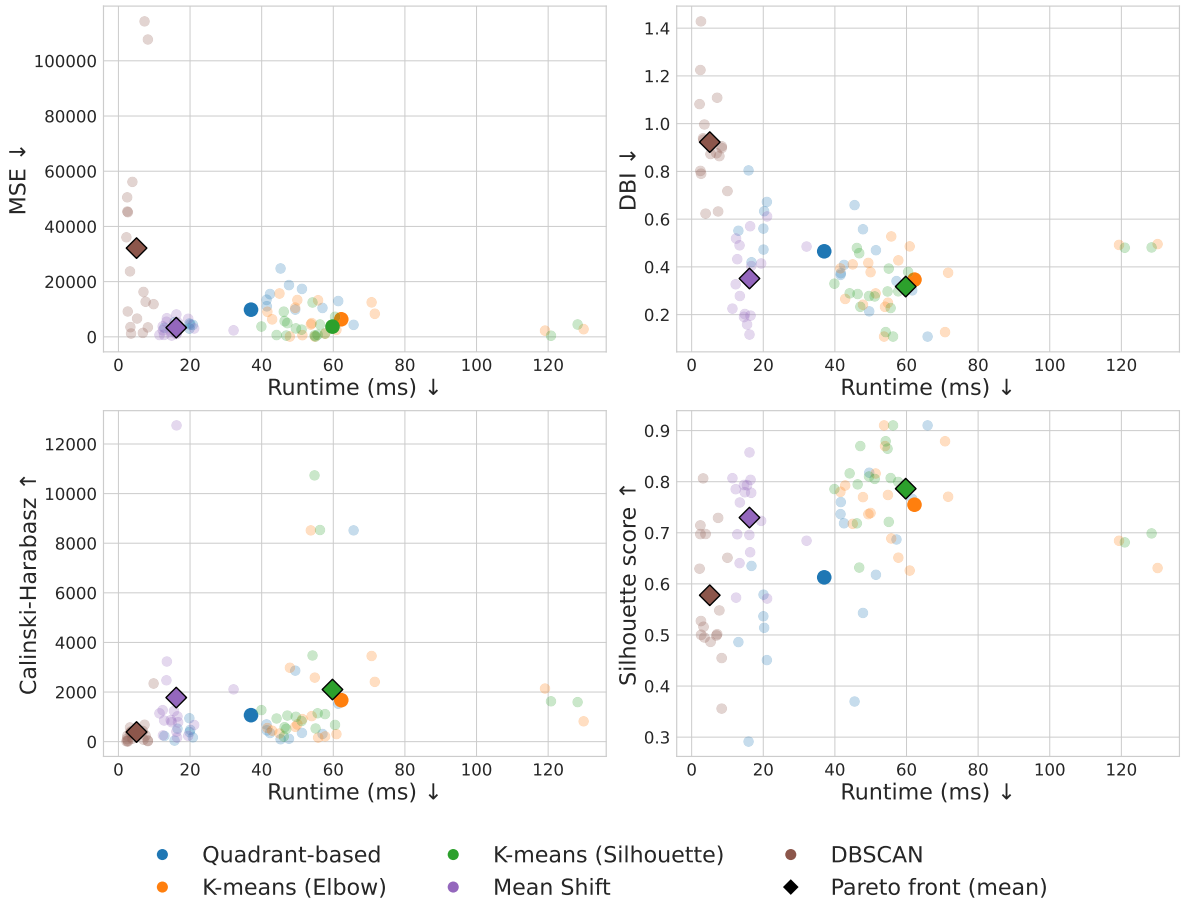

Figure 4. Trade-off between clustering runtime and each clustering measure –MSE, DBI, Calinski-Harabasz, and Silhouette score– among all clustering methods –Quadrant-based, K-means (with Elbow method and Silhouette score optimization), Mean Shift, and DBSCAN–. Colors represent each clustering method. Values attained by videos are shown as a scatter plot with light marker colors, while the mean across all videos is shown as a marker with a darker color. Means are evaluated as a Pareto front, and those in the Pareto front, i.e., the optimized trade-off between runtime and performance, are represented with a rhombus marker. The arrows represent whether the metric should be maximized (↑) or minimized (↓).

summarize the results, where each clustering performance metric (vertical axes) is compared against clustering runtime (horizontal axes). Each data point represented in a light color corresponds to a value from a video, while the dark color represents the mean. The color of the points indicates the clustering strategy used, and the rhombus marker shape identifies the means on the Pareto front. A mean value belongs to the Pareto front if it is non-dominated, meaning there is no other solution that can improve at least one objective without worsening at least one other objective. In this case, each plot is treated as an objective aimed at minimizing runtime, making DBSCAN the primary choice, as is reported at Table 8. However, due to its low performance metrics, the Mean Shift method is another solution, as both algorithms appear on the Pareto front in each plot. Following these two options is K-means with Silhouette optimization, which comes out three times out of four on the front as well. Although Mean Shift is not the fastest in terms of runtime, it offers a better balance between speed, with a mean under 17 ms, and high-performance metrics that result in well-defined and structured clusters, excelling in MSE, while also retrieving competitive outcomes in DBI, Calinski-Harabasz, and Silhouette scores.

The difference in performance between Mean Shift and DBSCAN lies in how the algorithms reach and expand clusters. Mean Shift is an iterative method that shifts data points to find local maxima of the kernel density estimation. This approach can be computationally intensive but generally produces high-quality clusters. In contrast, the DBSCAN algorithm checks only the points within a specified radius, making it computationally efficient. However, this may result in reduced clustering accuracy. Therefore, DBSCAN is often more suitable for large datasets that require extensive analysis. In the task addressed, cluster entry/exit points at intersections, the Mean Shift approach, although not the fastest, achieves high performance with low runtime.

We evaluate the robustness of the methodology and the clustering methods against tracking failures by including in the clustering process additional points as if they were starting/ending points in a legitimate, actually tracked trajectory, and checking how the clustering process is affected by the additional noise in the input data. To simulate failures in the tracking method, we draw these additional points from the set of all centers of bounding boxes for detected vehicles across all video frames in a scene. In this way, the probability distribution of additional, erroneous

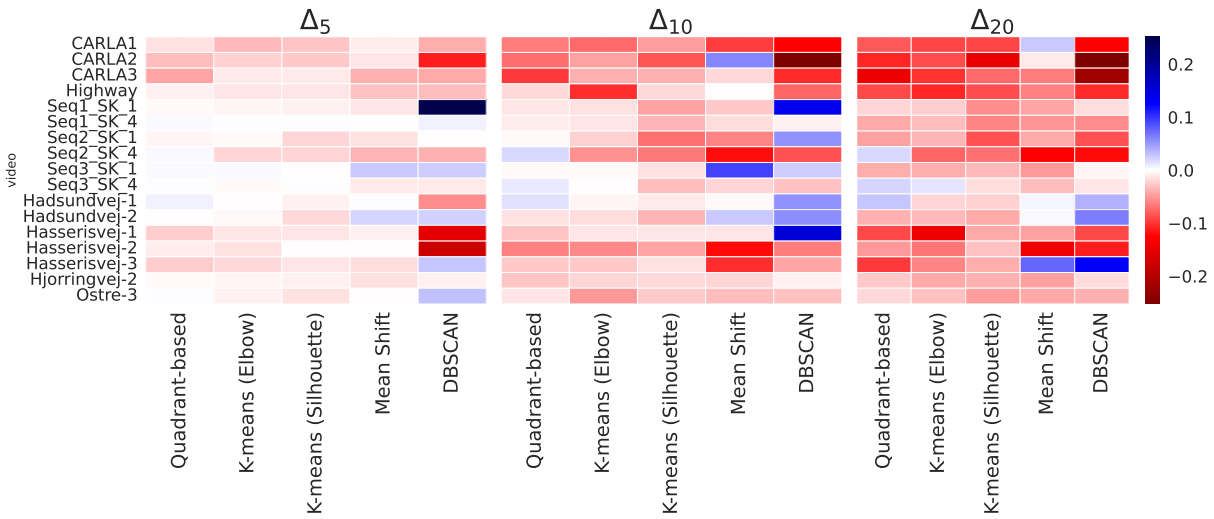

Figure 5. Heatmap of difference across percentage of noise added and non-noise baseline in Silhouette score. Since it is a function to maximize, negative values (indicated in red) denote a decrease when noise is added.

Figure 6. Samples of the first frame from Ostre-3 video, where the cluster centroids for each percentage of added noise are overlaid on the frame. From left to right and top to bottom, correspond to the legend: Quadran-based, K-means (Elbow), K-means (Silhoutte), Mean Shift, and DBSCAN.

points is not uniformly random, but mimics the observed patterns of traffic flow in the video sequence, simulating tracking failures that change the input data to the clustering methods. For each video sequence, we add increasingly higher amounts of simulated tracking failures, computed as percentages of all starting and ending points detected in the video sequence (i.e., before adding any noise). We test three scenarios, setting the amounts of simulated tracking failures as 5%, 10%, and 20% of the total starting and ending points in each video sequence. Silhouette score is chosen to analyze clustering method behavior under increasingly higher noise levels because it ranges [-1,1] and gaps between the non-noise baseline and the noise-added version are clearly represented.

Figure 5 depicts a heatmap illustrating how the Silhouette score changes when adding noise points to the input data for the clustering algorithm, relative to the baseline (i.e., without additional noise). If M_0 represents the Silhouette score without additional noise, and M_5, M_{10} and M_{20} represent the Silhouette score for 5%, 10%, and 20% additional noise, the heatmap illustrates the difference between Silhouette scores $\Delta_i = M_i - M_0, \;\; i \in \{5, 10, 20\}$ for each combination of video sequence and clustering method. In the Silhouette score, higher values indicate better performance. Therefore, a negative value, represented in red, indicates a decrease when noise points are included. As expected, when the input to the clustering method is corrupted with a noise point, the Silhouette score generally worsens. Specifically, for Δ_5, Δ_{10}, and Δ_{20}, the Silthouette score decreases by an average of 0.018, 0.024, and 0.043, respectively. These results show a slight overall worsening of performance. However, we can remark that clustering

methods are affected in different ways. K-means (with Silthouette or Elbow) attains overall higher gaps, whereas DBSCAN reported, in general, small gaps but with widely dispersed values, i.e., its performance fluctuates severely across videos. Finally, the Mean Shift algorithm reports the smallest gaps while exhibiting a stable behavior. It worsens as the added noise portion increases, but the largest difference is below 0.15, maintaining good performance against the impact of noise addition. To illustrate how it qualitatively affects the added noise, Figure 6 shows the cluster centroids for each scenario: no noise addition and with 5%, 10%, and 20% added. In K-means(Silthouette), a great centroid shift can be detected in each cluster and every strategy (0%, 5%, 10%, and 20%). Nevertheless, Mean Shift displaces only 20% of the centroids with respect to the baseline, with 5% and 10% of the centroids unaffected. As well, DBSCAN only has one cluster centroid rearranged, but it is quite moved, demonstrating its widespread values.

4 Conclusions

Traffic surveillance videos from road intersections capture traffic patterns that can be analyzed to extract valuable scene information. In this work, a three-stage method is proposed to automatically identify potential incoming and outgoing traffic flows. The process begins by detecting vehicle positions in each video frame with the YOLOv5x6 model. These frame-by-frame detections are then used to reconstruct vehicle trajectories through the Norfair tracking method. By applying unsupervised clustering to these trajectories, the resulting cluster centroids provide an automatic understanding of the traffic flows entering and exiting the intersection.

In order to provide a comprehensive evaluation of traffic patterns captured in those road intersection surveillance videos, we compare several different clustering algorithms – Quadrant-based, which is a spatial assignment of points spatially; two versions of K-means, applying the Elbow method and the optimization of the Silhouette score; Mean Shift, and the DBSCAN algorithm–. The inclusion of multiple clustering methods allows us to analyze the adaptability of each algorithm to the nature of the video data and identify which performs best in this context. Traffic flows in videos can vary in complexity, and each algorithm offers a unique approach to clustering, which helps us compare their effectiveness at handling the dynamic movement of vehicles. This multi-faceted approach ensures that the system is adaptable to a variety of real-world traffic scenarios.

To support the evaluation of the above-mentioned clustering performance, we incorporated the runtime by stages and several clustering performance metrics, in order to provide quantitative backing for assessing the quality of the clusters produced by the different algorithms, ensuring that the analysis is thorough and consistent. Additionally, a comprehensive analysis of the trade-off between clustering runtime and performance metrics within a multi-objective decision-making process has been conducted using a Pareto front. Finally, to test methodology robustness against tracker failures, a set of noise inputs corresponding to a percentage of noise added was included. To compare results, differences between the noisy and noise-free versions were extracted and reported as a heatmap. The evaluation demonstrates the effectiveness of the proposal, given the slight performance decrease.

This gives us a clearer understanding of which algorithm is best suited for extracting meaningful traffic flows from the video data. While DBSCAN runs faster, Mean Shift offers a better balance between computational cost, performance, and robustness in the presence of noise, making it the most suitable clustering method.

The proposed method has been tested with a set of videos, both from publicly available datasets and synthetically generated. Experimental results demonstrate that using better object detection models reduces the number of false negatives and false positives in the placement of cluster centroids at road ends. Since the method is robust to false starting/ending points in vehicle trajectories, it also works when vehicles follow unconventional trajectories (speeding, tailgating, repeatedly switching lanes) and in bad weather (rain and snow). The method can be useful when processing large batches of traffic videos from many different intersections, in order to provide awareness of the layout of the intersection, as well as to help in the detection of vehicles making anomalous entries or exits in each traffic video (i.e., far from any of the clusters, or in the fringes of an existing cluster). These findings highlight the potential of traffic forecasting, providing essential insights for intelligent transportation systems.

For future research on the system proposed in this article, the authors plan to explore two main research lines. The first involves experimenting with different tracking algorithms beyond the one used in the current work to determine whether these alternatives can improve performance. By testing various tracking methods, the authors aim to enhance the accuracy and reliability of vehicle trajectory detection, particularly in complex or crowded traffic scenes. This exploration of new tracking approaches may lead to better overall results and more robust handling of vehicle movements under diverse conditions.

The second research focus is on translating the coordinates of all real-world scenarios presented in this work to real-world GPS coordinates. This step would allow the system to map detected vehicle trajectories more accurately to real-world geographic locations, providing a more practical and scalable solution for real-world applications. By aligning the traffic data with GPS coordinates, the system can support advanced traffic analysis and integration with other geo-referenced systems, such as navigation or traffic management platforms as well as enabling data fusion from multiple cameras, in order to provide more data points to the clustering algorithms, and enabling the application of the algorithm to more complex traffic road layouts that cannot be adequately watched from the point of view of a single traffic camera.

To address the lack of semantic information provided by the methodology, since it just provides clusters of road localization, another further research line could be implementing a fourth stage post-processing where clusters will be semantically labelled. This proposal could be explored with supervised techniques, such as deep learning-based models or through a visual large model (VLM).

These three research directions will help extend the current system's applicability and improve its effectiveness in real-world traffic monitoring and analysis scenarios.

Acknowledgements

This work is partially supported by the Autonomous Government of Andalusia (Spain) under project PPRO-TIC163-G-2023 (TIC163-G-FEDER); also by the Ministry of Science and Innovation of Spain, grant number PID2022-136764OA-I00, project name Automated Detection of Non Lesional Focal Epilepsy by Probabilistic Diffusion Deep Neural Models. It includes funds from the European Regional Development Fund (ERDF). It is also partially supported by the Fundación Unicaja under project PUNI-003_2023, project name Intelligent System to Help the Clinical Diagnosis of Non-Obstructive Coronary Artery Disease in Coronary Angiography, the Instituto de Investigación Biomédica de Málaga y Plataforma en Nanomedicina-IBIMA Plataforma BIONAND under project ATECH-25-02, and the Instituto de Salud Carlos III, project code PI25/02129 (co-financed by the European Union). The authors thankfully acknowledge the computer resources, technical expertise and assistance provided by the SCBI (Supercomputing and Bioinformatics) center of the University of Málaga. They also gratefully acknowledge the support of NVIDIA Corporation with the donation of an RTX A6000 GPU with 48Gb. The authors also thankfully acknowledge the grant of the Universidad de Málaga and the Instituto de Investigación Biomédica de Málaga y Plataforma en Nanomedicina-IBIMA Plataforma BIONAND.

Data availability

The AAU RainSnow Traffic Surveillance Dataset (46) is an open-access dataset available at https://www.kaggle.com/datasets/aalborguniversity/aau-rainsnow
The Ko-PER Intersection dataset (47) is publicly available at https://www.uni-ulm.de/in/mrm/forschung/datensaetze.html
The 2014 CDNET dataset (48) is available at http://changedetection.net
The source code for this work is published at https://github.com/icai-uma/Unsupervised_detection_of_inflows_and_outflows_in_intersection_traffic_videos

Author Contributions

All authors listed have made a substantial, direct, and intellectual contribution to the work and approved it for publication.

Conflicts of Interest

The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] Datondji SRE, Dupuis Y, Subirats P et al. A survey of vision-based traffic monitoring of road intersections. *IEEE Transactions on Intelligent Transportation Systems* 2016; 17: 2681–2698.

[2] Jahongir Azimjonov AO. A real-time vehicle detection and a novel vehicle tracking systems for estimating and monitoring traffic flow on highways. *Advanced Engineering Informatics* 2021; 50.

[3] Ercan Avsar YO. Moving vehicle detection and tracking at roundabouts using deep learning with trajectory union. *Multimedia Tools and Applications* 2021; 17: 6653–6680.

[4] Carranza-García M, Galán-Sales FJ, Luna-Romera JM et al. Object detection using depth completion and camera-lidar fusion for autonomous driving. *Integrated Computer-Aided Engineering* 2022; 29(3): 241–258.

[5] Garcia-Aguilar I, García-González J, Luque-Baena RM et al. Optimized instance segmentation by super-resolution and maximal clique generation. *Integrated Computer-Aided Engineering* 2023; 30(3): 243–256.

[6] Chong YS and Tay YH. Modeling representation of videos for anomaly detection using deep learning: A review. *ArXiv* 2015; abs/1505.00523.

[7] Abbas Q, Ibrahim M and M J. Video scene analysis: an overview and challenges on deep learning algorithms. *Multimedia Tools and Applications* 2018; 77(16): 20415 – 20453. DOI:-10.1007/s11042-017-5438-7.

[8] Molina-Cabello MA, Luque-Baena RM, Lopez-Rubio E et al. Vehicle type detection by ensembles of convolutional neural networks operating on super resolved images. *Integrated Computer-Aided Engineering* 2018; 25(4): 321–333.

[9] Abbas A, Sheikh U, Al-Dhief F et al. A comprehensive review of vehicle detection using computer vision. *TELKOMNIKA (Telecommunication Computing Electronics and Control)* 2021; 19: 838–850. DOI:10.12928/TELKOMNIKA.v19i3.12880.

[10] Fernández JD, García-González J, Benítez-Rochel R et al. Anomalous trajectory detection for automated traffic video surveillance. In Ferrández Vicente JM, Álvarez-Sánchez JR, de la Paz López F et al. (eds.) *Bio-inspired Systems and Applications: from Robotics to Ambient Intelligence*. Cham: Springer International Publishing. ISBN 978-3-031-06527-9, pp. 173–182.

[11] Abbasi M, Shahraki A and Taherkordi A. Deep learning for network traffic monitoring and analysis (ntma): A survey. *Computer Communications* 2021; 170: 19–41.

[12] Jain NK, Saini R and Mittal P. A review on traffic monitoring system techniques. *Soft Computing: Theories and Applications* 2019; : 569–577.

[13] Novotny G, Liu Y, Morales-Alvarez W et al. Vehicle side-slip angle estimation under snowy conditions using machine learning. *Integrated Computer-Aided Engineering* 2024; 31(2): 117–137.

[14] Lopez-Fuentes L, van de Weijer J, González-Hidalgo M et al. Review on computer vision techniques in emergency situations. *Multimedia Tools and Applications* 2018; 77(13): 17069–17107. DOI:10.1007/s11042-017-5276-7.

[15] Vivacqua AS. Preparing a smart environment to decision-making in emergency traffic control management. In

Information Technology in Disaster Risk Reduction: Third IFIP TC 5 DCITDRR International Conference, ITDRR 2018, Held at the 24th IFIP World Computer Congress, WCC 2018, Poznan, Poland, September 20–21, 2018, Revised Selected Papers, volume 550. Springer Nature, p. 12.

[16] Pramanik A, Sarkar S and Maiti J. A real-time video surveillance system for traffic pre-events detection. *Accident Analysis & Prevention* 2021; 154: 106019.

[17] Mo X, Sun C, Zhang C et al. Research on expressway traffic event detection at night based on mask-spynet. *IEEE Access* 2022; 10: 69053–69062.

[18] Pineda-De-Luelmo S, Garcia-Espinosa FJ, Montemayor AS et al. Combining deep learning methods and rule-based systems for automatic parking space detection. *Integrated Computer-Aided Engineering* 2025; 32(1): 97–108.

[19] Krassnig PJ, Haselmann M, Kremnitzer M et al. Efficient surface defect detection in industrial screen printing with minimized labeling effort. *Integrated Computer-Aided Engineering* 2025; 32(1): 3–23.

[20] Zhou C, Fan L and Neri F. A spatio-temporal fusion deep learning network with application to lightning nowcasting. *Integrated Computer-Aided Engineering* 2024; 31(3): 233–247.

[21] Datondji SRE, Dupuis Y, Subirats P et al. A survey of vision-based traffic monitoring of road intersections. *IEEE Transactions on Intelligent Transportation Systems* 2016; 17: 2681–2698.

[22] Pan YA, Li F, Li A et al. Urban intersection traffic flow prediction: A physics-guided stepwise framework utilizing spatio-temporal graph neural network algorithms. *Multimodal Transportation* 2025; 4(2): 100207.

[23] Shen Z, Wang W, Shen Q et al. A novel learning method for multi-intersections aware traffic flow forecasting. *Neurocomputing* 2020; 398: 477–484.

[24] Jakubec M, Cingel M, Lieskovská E et al. Integrating neural networks for automated video analysis of traffic flow routing and composition at intersections. *Sustainability* 2025; 17(5): 2150.

[25] Tang J and Wang W. Vehicle trajectory extraction and integration from multi-direction video on urban intersection. *Displays* 2024; 85: 102834.

[26] Azimjonov J, Özmen A and Varan M. A vision-based real-time traffic flow monitoring system for road intersections. *Multimedia tools and applications* 2023; 82(16): 25155–25174.

[27] Song Jf, Wang Sy and Zhao Hl. Traffic flow detection at road intersections based on k-means and nurbs trajectory clustering. *Mathematical Problems in Engineering* 2020; 2020(1): 1383198.

[28] Osama Abdeljaber WA Adel Younis. Extraction of vehicle turning trajectories at signalized intersections using convolutional neural networks. *Arabian Journal for Science and Engineering* 2020; : 8011–8025.

[29] Tak S, Lee JD, Song J et al. Development of ai-based vehicle detection and tracking system for c-its application. *Journal of Advanced Transportation* 2021; 2021: 1–15. DOI: 10.1155/2021/4438861.

[30] Moradi A, Shahbahrami A and Akoushideh A. An unsupervised approach for traffic motion patterns extraction. *IET Image Processing* 2021; 15(2): 428–442.

[31] Wang C, Zourlidou S, Golze J et al. Trajectory analysis at intersections for traffic rule identification. *Geo-spatial Information Science* 2021; 24(1): 75–84.

[32] Wang W, Xie Y and Tang L. Hierarchical clustering algorithm for multi-camera vehicle trajectories based on spatio-temporal grouping under intelligent transportation and smart city. *Sensors* 2023; 23(15): 6909.

[33] Molina-Cabello MA, Luque-Baena RM, López-Rubio E et al. Vehicle type detection by convolutional neural networks. In *International Work-Conference on the Interplay Between Natural and Artificial Computation*. Springer, pp. 268–278.

[34] Molina-Cabello MA, Luque-Baena RM, López-Rubio E et al. A growing neural gas approach to classify vehicles in traffic environments. *International Journal of Computer Vision and Image Processing (IJCVIP)* 2017; 7(3): 1–12.

[35] Fernandez-Rodriguez JD, García-González J, Benítez-Rochel R et al. Automated detection of vehicles with anomalous trajectories in traffic surveillance videos. *Integrated Computer-Aided Engineering* 2023; 30(3): 293–309.

[36] Lin TY, Maire M, Belongie S et al. Microsoft coco: Common objects in context. In *European conference on computer vision*. Springer, pp. 740–755.

[37] Ultralytics. Reference repository for YoloV5. https://github.com/ultralytics/yolov5.

[38] Tryolabs. Reference repository for Norfair. https://github.com/tryolabs/norfair/.

[39] McQueen JB. Some methods of classification and analysis of multivariate observations. In *Proc. of 5th Berkeley Symposium on Math. Stat. and Prob.* pp. 281–297.

[40] Fukunaga K and Hostetler L. The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Transactions on information theory* 1975; 21(1): 32–40.

[41] Comaniciu D and Meer P. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on pattern analysis and machine intelligence* 2002; 24(5): 603–619.

[42] Ester M, Kriegel HP, Sander J et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96. pp. 226–231.

[43] Pedregosa F, Varoquaux G, Gramfort A et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research* 2011; 12: 2825–2830.

[44] Pedregosa F, Varoquaux G, Gramfort A et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research* 2011; 12: 2825–2830.

[45] García-González J, Molina-Cabello MA, Luque-Baena RM et al. Road pollution estimation from vehicle tracking in surveillance videos by deep convolutional neural networks. *Applied Soft Computing* 2021; 113: 107950.

[46] Bahnsen CH and Moeslund TB. Rain removal in traffic surveillance: Does it matter? *IEEE Transactions on Intelligent Transportation Systems* 2018; : 1–18DOI:10.1109/TITS.2018.2872502.

[47] Strigel E, Meissner D, Seeliger F et al. The ko-per intersection laserscanner and video dataset. In *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*. IEEE, pp. 1900–1901.

[48] Wang Y, Jodoin PM, Porikli F et al. Cdnet 2014: An expanded change detection benchmark dataset. In *Proceedings of the IEEE conference on computer vision and pattern recognition*

workshops. pp. 387–394.

[49] Dosovitskiy A, Ros G, Codevilla F et al. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*. pp. 1–16.

[50] Maulik U and Bandyopadhyay S. Performance evaluation of some clustering algorithms and validity indices. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 2002; 24(12): 1650 – 1654. DOI:10.1109/TPAMI.2002.1114856.

[51] Rousseeuw P. Rousseeuw, p.j.: Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. comput. appl. math. 20, 53-65. *Journal of Computational and Applied Mathematics* 1987; 20: 53–65. DOI:10.1016/0377-0427(87)90125-7.