CIRRELT

Centre interuniversitaire de recherche
sur les réseaux d'entreprise, la logistique et le transport

Interuniversity Research Centre
on Enterprise Networks, Logistics and Transportation

Applications of Multi-Level Pattern Learning to Traffic Scene Interpretation and Anomaly Detection

Mohamed Gomaa M. Mohamed
Nicolas Saunier

July 2017

CIRRELT-2017-48

Bureaux de Montréal :
Université de Montréal
Pavillon André-Aisenstadt
C.P. 6128, succursale Centre-ville
Montréal (Québec)
Canada H3C 3J7
Téléphone : 514 343-7575
Télécopie : 514 343-7121

Bureaux de Québec :
Université Laval
Pavillon Palasis-Prince
2325, de la Terrasse, bureau 2642
Québec (Québec)
Canada G1V 0A6
Téléphone : 418 656-2073
Télécopie : 418 656-2624

www.cirrelt.ca

Applications of Multi-Level Pattern Learning to Traffic Scene Interpretation and Anomaly Detection

Mohamed Gomaa M. Mohamed[1], Nicolas Saunier[2,*]

[1] Department of Public Works, Faculty of Engineering, Cairo University, Cairo, Egypt. 12613,

[2] Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT) and Department of Civil, Geological and Mining Engineering, Polytechnique Montréal, P.O. Box 6079, Station Centre-ville, Montréal, Canada H3C 3A7

Abstract. The availability of surveillance cameras and the development of video analysis tools enable extracting massive amounts of positional data of road users to further analyze their behaviour. This paper is motivated by the need for automatic and unsupervised methods to handle “big (video) data” for scene interpretation. At the beginning, an existing open source video analysis tool is used. A smoothing algorithm is developed for the extracted trajectories and its performance is evaluated quantitatively. The performance measure is improved by 86-95 % for all road users, up to 97 % for vehicles, and only up to 80-86 % for pedestrians. A multi-level motion pattern learning (MLMP) framework is proposed and validated for automated scene interpretation and anomalous behaviour detection. A comprehensive experimental analysis of three intersections demonstrates the framework abilities in several ways: it reduces the computation cost up to 90 % for motion pattern learning, connects incomplete trajectories with performance accuracy up to 97 %, removes outliers from trajectory dataset, and detects anomalous events such as abnormal left turn, improper turns, and excessive speed.

Keywords: Road user behaviour, video data, trajectory learning, motion patterns, anomaly detection.

Acknowledgements. The authors wish to acknowledge the financial support of the Natural Sciences and Engineering Research Council of Canada (NSERC) (Grant No. 402320-2011). The authors declare that there is no conflict of interest regarding the publication of this paper.

---

* Corresponding author: Nicolas.Saunier@cirrelt.ca

Dépôt légal – Bibliothèque et Archives nationales du Québec
Bibliothèque et Archives Canada, 2017

Introduction

Surveillance cameras have been installed over the last decades in many road elements providing a continuous monitoring of traffic activities. The development of video analysis tools enable extracting massive amounts of positional data of road users to further analyze their behaviour. Behaviour understanding is a fundamental procedure towards road safety analysis, in particular surrogate safety analysis (SSA) which relies on the observation of road user interactions without a collision [1]. The core concept of SSA is the collision course, i.e. a situation in which road users would collide if their movements remain unchanged. This depends on the ability to predict the road users' future positions: among various motion prediction methods [1], using the frequent road user movements on the studied site, or Motion Patterns (MPs), is considered to be the most realistic and robust method. Indeed, road users do not move randomly in a scene, they usually follow the MPs. However, MPs cannot be easily predefined in supervised learning methods especially for large datasets. This paper reports on the first phase of a research project that aims to develop a consistent and generic framework for automatic scene interpretation and SSA. In particular, this paper is motivated by the need for automatic and unsupervised methods to handle "big (video) data" for scene interpretation.

Unsupervised scene interpretation is a challenging problem for traffic scenes. It should provide an appropriate description of road users' behaviour with respect to traffic rules and the road environment, with or without prior knowledge of the scene. This is mainly performed by learning the MPs using machine learning techniques, in particular clustering algorithms. Most of the existing clustering procedures require setting the number of clusters in advance. In addition, the resulting clusters are not always easy to interpret according to our previous experiments. A custom clustering algorithm was developed in [2] that trades the number of clusters for a maximum similarity and remains interpretable using prototypes (longest trajectory or time series in each cluster) as cluster representatives. Nevertheless, existing trajectory learning methods have several shortcomings such as:

1. The construction of pairwise similarity matrix is computationally expensive in both time and required storage space for large datasets with non-metric similarity measures (e.g. Longest Common Sub-Sequence (LCSS) [3]). Computing the LCSS similarity between two trajectories requires constructing an internal matrix for the matching between all points in each trajectory (or up to a defined bound [3]). This internal matrix is computed $\frac{(N^2-N)}{2}$ time for a training dataset with N trajectories to construct the full pairwise similarity matrix.
2. Tracking dependency: the quality of MP learning depends on tracking performance. Good quality trajectories, with little noise or tracking errors, result in robust learning.

Because of the complex nature of a traffic scene and of the computation cost for large trajectory datasets, it is recommended to learn MPs in a hierarchical fashion. A Multi-Level Motion Patten Learning Framework (MLMP) was first proposed and introduced in our previous work [4]. It consists of generic algorithms that enable automated scene interpretation, behaviour understanding, and anomalous behaviour detection. Moreover, logical constraints were proposed to accelerate the processing time for MP learning. This paper aims to refine the MLMP and evaluate it in a larger experimental study for MP-based behaviour understanding. The following contributions are described in the present paper:

1. The proposed smoothing algorithm is evaluated comprehensively for vehicles in addition to pedestrians.
2. A new method of connecting fragmented trajectories is proposed and evaluated.
3. The effectiveness of the framework in speeding up MP learning and reducing required memory space is demonstrated on real data.
4. Anomalies are detected and explained.
5. The framework is validated on several case studies, including a case study with a low camera angle.

The reader is referred to [5] for more details. The remainder of this paper is organized as follows: the next section covers the review of previous work on scene interpretation, then the video analysis method and the MLMP framework are presented, followed by the experimental results of three real world case studies of intersections in Montreal, Canada, before finally concluding with the discussion of the results.

Previous Work

Scene interpretation applied to transportation seeks to understand traffic behaviour and analyze activities in the scene. One accepted framework for scene interpretation was introduced in [5] where the scene is modeled as a topological map consisting of nodes and edges. The nodes are points of interest (POIs) which refer to regions where some activity occurs, whereas the edges are the activity paths that represent how the objects move between the POIs. Therefore, the activity analysis can be summarized in two main learning tasks: learning the POIs and the activity paths. There are two main approaches for activity learning: the most popular learns trajectories while are more recent is topic modelling (see [6, 7, 8] for more details).

Point of Interest (POIs) Learning

The POIs correspond to the origins and destinations of trajectories, which correspond mainly to the entry/exit zones and sometimes stop zones (i.e. if objects are not tracked when stopping). There is relatively little work in the literature addressing the learning of POIs. Stauffer [9] proposed a method to learn the entry/exit zones that was called sources and sinks in his work. The zones are learnt using a hidden Markov model (HMM) where all sequences are of length two, there are only two states (entry and exit) and the state model is not shared across time. A HMM represents each trajectory with hidden states and a transition matrix. Because of noisy tracking data, the author proposed a method for stitching broken/incomplete tracks. Given that this method assumes that all tracks are broken, a transition matrix is created using all pairwise transition likelihoods. Then, any two corresponding tracks should be stitched. In this work, the detection of the entry and exit zones and the stitching of the trajectories are conducted and updated simultaneously using an iterative process. Although the author recommended a thorough quantitative analysis for the stitching procedure, no information is provided about the performance of the stitching. The dependency of zone detection and stitching could be an issue as well, i.e. the errors in zone detection may result in errors in the stitched trajectories and vice versa.

A popular and straightforward method used to model the POIs is based on Gaussian Mixture Models (GMM) and the Expectation Maximization (EM) algorithm [10, 5, 11, 12]. These algorithms are also mainly applied to the entry and exit datasets constructed using the trajectory endpoints. However, fragmented

trajectories caused by tracking failures and occlusions lead to false entry/exit zones (noisy clusters). These noisy clusters can be distinguished using a density criterion [5, 12]. POI leaning is applied to a simulated intersection dataset in [12]: the authors used the detected entry and exit zones to filter the trajectory dataset into two datasets of complete and incomplete trajectories. Complete trajectories go from an entry zone to an exit zone, while the rest are considered incomplete. Only the complete trajectories were used in the MP learning phase, which may represent an important loss of some typical motion information (e.g. left turning users stopped in the intersection waiting for a gap in the opposite through traffic). Recently, Nedrich & Davis [13] modeled the entry and exit zones using a standard mean-shift clustering algorithm. Although the authors assert that mean-shift clustering is able to localize cluster modes automatically without knowing the number of clusters, the algorithm depends on a bandwidth criterion that is as challenging to select as the number of clusters. They used the detected zones to identify static occlusion regions based on analyzing the distance distributions between the zones. From the available literature, it is evident that the GMM and the EM algorithms have been quite successful for learning interest points. It was tested mainly on cases of indoor scenes and limited work has been done in real cases of traffic scenes. In addition, the detected POIs have the potential of being used for further applications. Consequently, a consistent framework of the detection and applications of POI needs further investigations.

Behaviour Analysis based on Trajectory Learning

Trajectory learning, a popular approach for scene interpretation and behaviour analysis, aims to cluster a dataset of observed trajectories into the main subsets of similar trajectories or motion patterns. This approach has gained the researchers' interest because it fits into a typical surveillance and computer vision architecture as a higher level of the analysis where the trajectory of each object is detected and tracked at a lower level. This kind of analysis has three steps: pre-processing, clustering, and modelling (see [6] for more details). In some cases, the pre-processing step can be merged into the clustering step. The preprocessing step is performed to set up trajectories suitable for standard clustering techniques that operate only on series or vectors that have equal length. Typically, the trajectories do not share the same lengths, even if they move along similar paths, for many reasons such as the variation of individuals' behaviour, the differences in moving objects speed and the tracking performance. To ensure equal lengths, the simplest and most popular technique is to normalize the trajectories, by resampling (e.g. linear interpolation), padding, e.g. by repeating the last position (assuming zero velocity), and extrapolating the trajectories. Normalization is performed on all the extracted trajectories whether they are complete or incomplete trajectories. Most normalization methods either discard or distort the velocity information. Accordingly, it is recommended to avoid pre-processing the trajectories before leaning and to search for a method that can deal with the data as it is.

The clustering is the important step of trajectory learning. The key component of clustering is how to compute the similarity/distance between any pair of trajectories. The Euclidean distance is the popular, but it requires that both time series have the same length and it is sensitive to distortions (e.g. shifting along the time axis). Therefore, a pre-processing step is mandatory before using the Euclidean distance. The development of elastic distance and similarity measures, such as Dynamic Time Warping (DTW) and LCSS, overcome such drawbacks. Both DTW and LCSS are implemented using dynamic programming. DTW attempts to find the best alignment between two time series by minimizing the distance between them. Conversely, LCSS finds the length of the longest matching subsequence by comparing every point of the two time series using a given matching method. One limitation of the elastic measures is the high computation cost to construct a pairwise similarity matrix. Morris and Trivedi [14] evaluated different

similarity measures (Hu [15], PCA (Principle Component Analysis), DTW, LCSS, PF (Piciarelli and Foresti [16]), modified Hausdorff) and clustering methods for trajectories as a first step to understand road user behaviour. After tests on six different datasets, the authors concluded that LCSS was consistently the top performer. Nonetheless, the LCSS does not consider well the rate of change. To overcome this shortcoming, a new similarity measure based on LCSS, known as aligned LCSS (ALCSS), is proposed in [2].

The second component of the clustering is the selection of the clustering algorithm. Clustering is the task of grouping similar elements from a dataset in an unsupervised fashion. Among clustering algorithms, the hierarchical, partitioning, density based, and grid based algorithms are the best known [17, 18]. A survey of time series clustering is presented in [19] for further reading. Partitioning algorithms are the most popular technique for their simplicity (e.g. standard K-means and its soft variant of fuzzy C-means (FCM)). They are based on the following procedure: 1) selecting the initial K (or C) centroids, 2) assigning each element to the nearest centroid, and 3) updating the centroids. Steps 2 and 3 are repeated until the assignments no longer change. To compute the centroids, these partitioning algorithms operate only on equal length time-series where the mean of these time series can be computed. Another clustering algorithm used for trajectories learning is spectral clustering. A spectral clustering algorithm is presented in [20] that can be carried out easily using standard linear algebra libraries. This algorithm was tested on a number of challenging clustering situations, showing that spectral clustering performs better than traditional clustering algorithms. The advantage of spectral clustering methods is that it makes no assumptions on the distribution of data points. Instead, it relies on the eigen decomposition of a similarity matrix which approximates an optimal graph partition. Therefore, the similarity/distance matrix can be constructed using any distance function (e.g. LCSS), then any traditional clustering (e.g K-mean) can be applied to the top eigen vectors matrix derived from computed similarity matrix. Morris and Trivedi [12] have successfully learnt the vehicle trajectories using spectral clustering algorithm based on LCSS and FCM. Spectral clustering is fast and takes as only input the predetermined number of groups. In our experience, finding the number of clusters by trial and error proved to be a challenge, and the resulting clusters are not always easy to interpret [2].

In our recent work [4], the MLMP has been proposed with the following advantages:

1. It includes a new algorithm to reconstruct smooth road user trajectories based on feature trajectories.
2. It can filter trajectories, detect outliers, and speed up MP learning by exploiting the learnt entry and exit zones of the traffic scene.
3. The MPs are learnt in two stages using spatial (positions) and temporal (speed) information and then used for anomaly detection.

The MLMP capabilities for behaviour analysis and anomaly detection were evaluated on a single case.

Methodology

The methodology is summarized in the flow chart in Figure 1. The following sections provide a brief description of each component in the methodology and readers are referred to [4] for details and a complete description of the methodology.

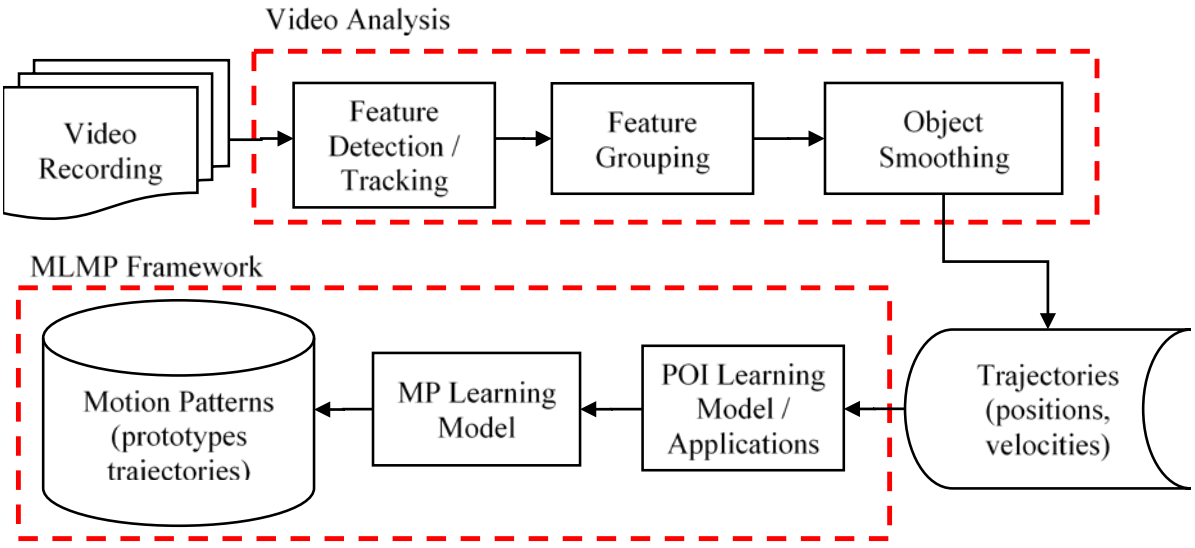


Figure 1: Methodology Flow Chart

Video Analysis

A video tracking tool from the open source "Traffic Intelligence" (TI) project [21] is used to detect and track moving road users: the result, road user trajectories, is the main input of the MLMP. TI relies on feature-based video tracking [22] where all distinct points (features) that move (more than a defined distance) are tracked from frame to frame The features are tracked through each frame using the well-known Kanade-Lucas-Tomasi (KLT) interest point tracker [23]. A sample of tracked features points is shown in Figure 2a).

The output of feature tracking algorithm is a large number of trajectories. Mainly, one or more features trajectory represent one road user. The second step of video analysis is to group feature trajectories into road user (object) hypotheses: features within a defined spatial proximity that have a similar motion are grouped as an object hypothesis. A road user is thus represented by a set of feature trajectories and deriving one overall trajectory, ideally corresponding to the centroid, is not easy. The current default solution implemented in TI is the mean of the feature positions at each frame: this average trajectory is noisy and only suitable for visualization purposes. Figure 2b) shows an example of grouped features into vehicle and pedestrian objects (road users' classification is performed based on aggregated speed values as will be discussed later). It is clear that the average trajectory is noisy especially when a road user enters and exits the scene and when it is partially occluded. That is why motion patterns were learnt previously from feature trajectories, which are smoother [24]. Although the features track a road user well and have little noise, they have other issues such as:

1. They are fragmented which affects the detection of the entry/exit zones.
2. The feature trajectories constitute a larger dataset, which affects the time necessary to learn the MPs.

It would therefore be beneficial to learn MPs from road user trajectories, providing that the noise in their trajectories could be reduced. A smoothing algorithm proposed in [4] is used to reconstruct a smooth road user trajectory. The algorithm relies on extracted features trajectories of each road user: the

resulting object trajectory has less noise and better reflects the road user dynamics. Later, road users are classified based on a simple speed criterion. The traditional speed classifier uses the maximum speed reached by the road users to distinguish between pedestrians and vehicles. As an alternative, we used 95th percentile instead of maximum function to avoid the effect of outliers. The threshold used is 15 km/h (see Figure 2b)). In that case, cyclist trajectories may be classified as pedestrians or vehicles.

a) Feature tracking algorithm b) Feature grouping algorithm

Figure 2: sample of road users extracted using TI tracking tool (For each bounding box, the letters C and P mean respectively car (motorized vehicle) and pedestrian).

A Multi-Level Motion Pattern Learning Framework

The proposed MLMP learns road user behavior through two main models based on the road user trajectories as follows:

The POI Model

The POI model relies on Gaussian Mixture Models (GMM) and the Expectation Maximization (EM) algorithm. As in [5], the POIs are learnt from trajectory endpoints based on GMMs and the EM algorithm and the results correspond to entry and exit zones as well as clusters of noisy points (e.g. caused by moving occlusions, stopping/starting vehicles and tracking errors) and sometimes clusters of static occlusions zones (e.g. caused by foreground static occlusions such as a lampposts, or trees). The static occluded zones have the same characteristics as entry/exit zones and are labelled manually at this point. On the other hand, the moving occlusion zones are represented by wider Gaussian distributions, with lower density than the entry and exit zones. Once the POIs are detected, they are used in different applications to deal with noisy trajectories (outliers) and to speed up the computational time of the MP learning phase. This is performed in an unsupervised multi-level procedure as follows:

1. *Filtering algorithm:* trajectories are filtered into complete and incomplete trajectories. Trajectories are complete if they go from an entry zone to an exit zone, which constitutes an activity path (AP).

2. *Connection algorithm:* some fragmented trajectories may be reconstructed into complete trajectories. The connection algorithm follows the following two steps:

    a. Finding candidate incomplete trajectories using the POIs: incomplete trajectories are identified automatically in the filtering algorithm. The dataset of incomplete trajectories contains three sets of trajectories: the trajectories starting in an entry zone (iT_1), the trajectories ending in an exit zone (iT_2), the trajectories that do not start or end in an entry

or exit zone (iT_3), and the occluded trajectories, defined as incomplete trajectories starting and/or ending in a static occlusion zone. The dataset is processed systematically, considering all incomplete trajectories in iT_1 with incomplete trajectories in iT_2 and iT_3. This simple procedure is useful to identify candidate trajectories automatically and to speed up the connection of incomplete trajectories.

b. Logical connection procedure: once the candidate trajectories are identified, the connection procedure based on spatial and temporal proximity is used to connect any pair of incomplete trajectories (T_1, T_2), where T_1 is the candidate first segment and T_2 is the candidate second segment. Spatial proximity is measured through the Euclidean distance between the last position (x_1,y_1) of T_1 and the first position (x_2,y_2) of T_2 which should be less than a defined distance Δ_1 as follows:

$$Distance = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \leq \Delta_1$$

The second constraint is the temporal proximity defined by an acceptable stopping duration Δ_2. It is measured by the difference between last instant F_1 of T_1 and first instant F_2 of T_2.

Duration = $F_2 - F_1 \leq \Delta_2$

Other proximity constraints can be added such as a similar motion constraints (two trajectories move in the same direction) and a pixel intensity constraint (last and first point have a similar pixel value). In case of multiple potential candidates, the trajectories with the minimum distance and minimum stopping duration are selected, with the temporal proximity having priority in the selection.

c. *Cleaning algorithm:* The dataset of complete trajectories is used for MP learning. Because outliers may remain in this dataset, a pre-processing procedure is recommended. The proposed approach detects outliers by analyzing the distribution of the travelled distances in each AP: it is expected that vehicles moving along a given AP will have similar travelled distances. The extreme distances therefore represent outliers. To accomplish this, the distributions are analyzed using boxplots and their traditional statistics: the median, the first (Q1) and third (Q3) quartile, the interquartile range (IQR=Q3-Q1) and the "whisker" limits typically defined as Q1-1.5 IQR and Q3+1.5 IQR. The usual application is to consider points outside of the whiskers as outliers. For this application, a distinction is made between mild and extreme outliers. The extreme outliers are beyond Q1-3 IQR and Q3+3 IQR which empirically correspond to grouping or smoothing errors and extreme unusual movements and are therefore removed from the AP. Mild outliers are beyond Q1-1.5 IQR and Q3+1.5 IQR, but not beyond the limits of extreme outliers. The mild outliers correspond mostly to lane changes and mild unusual movements that are kept. These outliers can be reviewed for a better understanding of the scene and activities.

MP Learning

Trajectories in each AP form the training dataset for MP learning using a two-stage trajectory clustering method based on spatial and temporal information. The MPs are learnt for each AP using spatial information at the first stage, and then these MPs are further clustered using temporal information in the

second stage. The hierarchical clustering is presented in Figure 3. Each MP is represented by its longest trajectory, known as a prototype. With the purpose of comparing the trajectories without pre-processing that would distort the data, a similarity measure should be able to handle variable length inputs. LCSS can deal with variable length vectors and is robust to noise and outliers as some points may not be matched [25]. Temporal dynamics, measured in particular by speed, are important motion characteristics that may vary within a spatial AP represented by the same spatial prototype trajectory learnt in the first stage. Therefore, the speed profile should be studied for each spatial MP generated by the first stage. To differentiate properly speed profiles, a similarity measure that considers the rate of change is needed. The Aligned LCSS (ALCSS) similarity measure proposed in [2] is used for this stage. A cluster of small size corresponds to a low probability of occurrence and is used to detect anomalous events. Anomalies are defined as unusual behaviours in terms of position and speed (e.g. excessive speed, violations of traffic law and unsafe movements, and tracking errors).

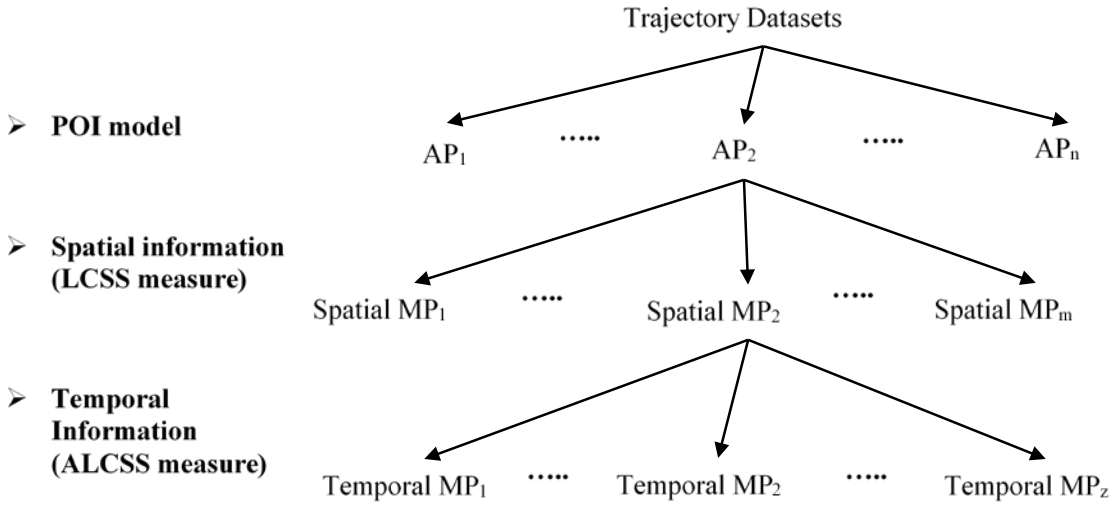


Figure 3: Two stage motion pattern hierarchical leaning

Experimental Results

This section presents the experimental results of the MLMP. It demonstrates the applicability of the proposed methodology for behaviour analysis in an unsupervised manner and highlights its application for anomalous detection. The framework algorithms are implemented in the open source Python language using several scientific libraries, in particular, scikit-learn (available at http://scikit-learn.org/stable/index.html) for GMM and EM, and most are or will be available in the TI project.

Data Description

The proposed methodology is evaluated using three different cases studies of video recordings. These case studies are intersections located in Montreal, Canada; 1) intersection of Guy Street and Boulevard Rene Levesque in downtown Montreal (Guy intersection), 2) intersection of St-Marc Street and Boulevard de Maisonneuve west in downtown Montreal (St Marc intersection) and 3) intersection of Atwater Street and Saint Jacques Street in south-west Montreal (Atwater intersection). Video data is captured using a consumer camera at a resolution of 1280 x 720 pixels (at 29.97 fps (frame per second)). Video data for

Guy and St Marc intersections were recorded from a high-rise building facing the intersection, while in the Atwater intersection the camera was put in the available three story building across the intersection. It is expected that the latter intersection suffers from poorer tracking and grouping caused by the low camera angle. The duration of the recorded video in each case is approximately 1 hour. The data is recorded on weekdays in August 2012 for the three sites and respectively at noon (12 p.m.), in the afternoon (3:00 p.m.), and the early evening (7 p.m.) for the Guy, Atwater, and St Marc intersections. Figure 4 shows the camera Field of View (FOV) of each intersection. It can be seen that the Guy and St Marc intersections are captured from an almost overhead view of the intersection, while the Atwater intersection has a low angle view that can lead to a high level of occlusions.

a) Guy intersection

b) St Marc intersection

c) Atwater intersection (red box is the studied area)

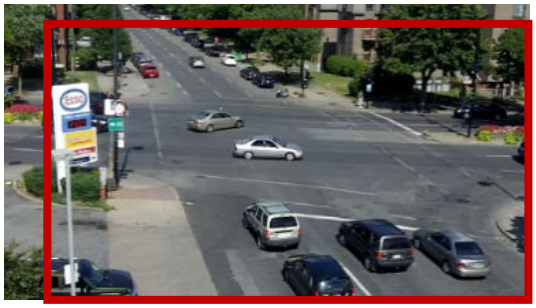


Figure 4: Camera views for each studied intersection

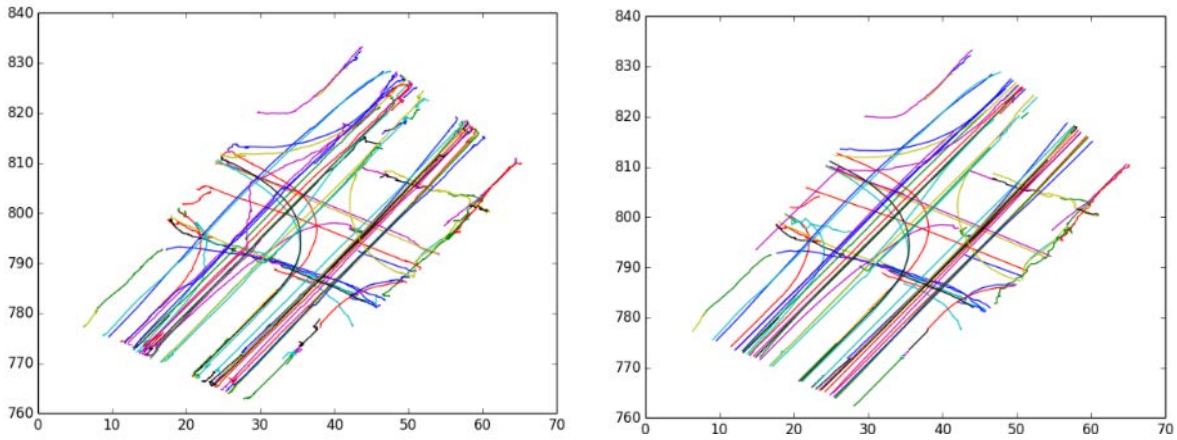
Smoothing Algorithm Results

We have implemented the smoothing algorithm with the Python programming language. In order to verify our algorithm, a quantitative criterion named Cumulative Squared Jerk (CSJ) is proposed in [4] and is used to measure the effectiveness of the smoothing algorithm. The CSJ can be calculated from the positions or the velocities using the following equation:

$$CSJ = \begin{cases} \sum_{t_f}^{t_l} (\dddot{x}(t)^2 + \dddot{y}(t)^2) \; for \; position \; input \\ \sum_{t_f}^{t_l} (\ddot{v}_x(t)^2 + \ddot{v}_y(t)^2) \; for \; velocities \; input \end{cases}$$

Where $\dddot{x}$ and $\dddot{y}$ are the third time derivatives of the x and y positions respectively, $\ddot{v}_x$ and $\ddot{v}_y$ are the second time derivative of the v_x and v_y velocities respectively and t_f and t_l are the trajectory first and last instants.

A sample of the first 100 trajectories is selected randomly from Guy intersection dataset and examined in Figure 5. This sample contains trajectories of different road users (pedestrians, cyclists, and motorized vehicles). Examining the performance of our algorithm visually shows its ability to remove noise effectively. This finding is confirmed quantitatively using the proposed criterion: the CSJ value for the original trajectories equals 11.394 while it equals 0.97 for smoothed trajectories. A smaller value of CSJ indicates less noise and smoother trajectories. The main limitation of this algorithm appears to be for trajectories having an over-grouping problem. Over grouping occurs when multiple (two or more) objects are tracked as one object.



a) original trajectories b) smoothed trajectories

Figure 5: Smoothed and original 100 trajectories

To further demonstrate the performance of our algorithm for the three studied intersections, CSJ is calculated for any trajectory using the position information of both the original average object trajectory (TR1) and the smoothed one (TR2), and using the velocity information of the average feature velocities provided by TI (SP3).The calculation is performed for the following datasets: the whole dataset, the vehicle trajectories only, and the pedestrian trajectories only.

Table 1 shows the results of CSJ calculation applied on all trajectories for each dataset at each site. At first glance, the mean CSJ and its standard deviation (s.d.) for the Atwater intersection are larger which reflects the level of noise compared to other intersections. This noise, as stated earlier, is due to the low angle camera that makes trajectories more prone to dynamic occlusions and over-grouping errors. Over all datasets, CSJ values for TR2 are much smaller than CSJ values for TR1, which confirms the effectiveness of the smoothing algorithm. Comparing the smoothness of TR2 and SP3, we found that SP3 was smoother

than TR2. Consequently, the positions are smoothed using the proposed algorithm while velocity datasets were chosen using the original TI computations since it was already computed and has less noise.

The results also show that the algorithm performs better for vehicle trajectories than for pedestrian trajectories. CSJ can be reduced by 86-95 % for all road users, up to 97 % for vehicles, and only up to 80-86 % for pedestrians. A possible reason of the lower performance for pedestrians is due to the periodic and cyclic characteristic of pedestrian motion, which affects the estimation of the relationship between object and features. In addition, the assumption of constant distance and angle is violated for pedestrians, which, unlike vehicles, are non-rigid.

POI Model

POI Detection Experiments

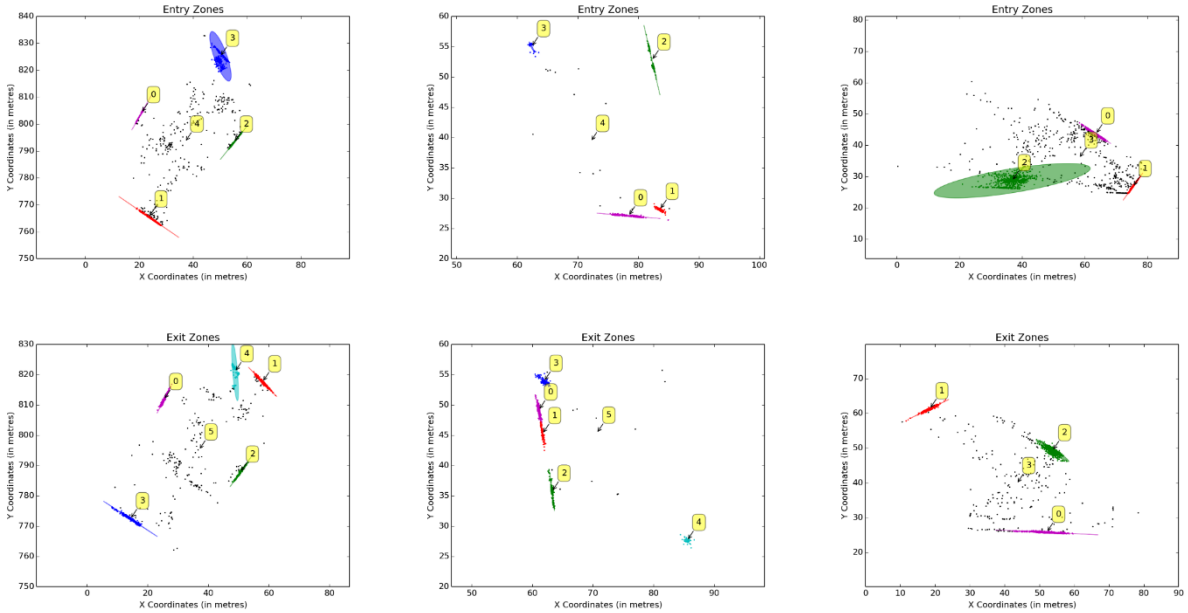
Only vehicle trajectories are used for further analysis. The parameter for the GMM learning is the number of components or expected zones in the scene, including the noise clusters. Although component numbers can be estimated automatically using the Bayesian Information Criterion (BIC), the result often suffers from over fitting (selecting more zones than necessary) [13, 26], which was observed in our experiments as well. Therefore, the number of components is chosen by trial and error for each scene. POIs are learnt and classified into entry zones, exit zones, and noise clusters based on a density criterion (see [4] for more details on the criterion).Sub-Figure 6 (i) represent the entry and exit zones overlaid over a camera image of each intersection. All detected POIs and their covariance ellipses are shown in Sub-Figure 6 (ii) in world coordinates of each intersection. Following are the experimental results of each case study:

1. In Guy intersection, the number of POIs is chosen as five and six for the entry and exit datasets respectively. We were able to detect all four-entry zones and all four exit zones of this four legged intersection. In addition, the static occluded zone (exit zone 4) under the pole in the top-right corner of the scene is detected as an extra exit zone, but not as an additional entry zone. This is due to the closeness of the occluded zone to the entry zone: it was therefore merged with the closest entry zone producing a relatively wider Gaussian distribution. In this dataset, tracking failures caused by moving occlusion are clustered as a large Gaussian noise cluster over the whole scene.

2. St Marc intersection has a different layout: it is a four-legged intersection with bidirectional movements in addition to a segregated bidirectional bike lane. The number of POIs is chosen as five and six for entry and exit datasets respectively by trial and error. Results confirm the ability of learning all of the expected entry and exit zones. We also detected the entry and exit zones for the bike lane. The only issue noted here was the split of one exit zone into two exit zones (zones 0, 1), which may be caused by the location of the exit at the limit of the FOV which makes the exit zone longer.

The Atwater intersection is also a four-legged intersection, where one road is unidirectional (northbound). The number of components is chosen as four for both entry and exit datasets. The results are three entry and three exit zones, plus a noise zone in each category. One of the entry zones has a wider Gaussian distribution. The reason is that this zone is located behind an area of trees and a median, which leads to many fragmented trajectories. Similar to the other intersections, the beginning and ends of noisy trajectories are each represented as one cluster with a large Gaussian distribution.

(i) Image Coordinates (the top and bottom rows represents respectively the entry and exit zones)



(ii) World Coordinates (the top and bottom rows represent respectively the entry and exit zones)

a) Guy intersection b) St Marc intersection c) Atwater intersection

Figure 6: Detected POI zones for studied intersections

POI Applications Experiments

The POI applications are performed with three algorithms; 1) a filtering algorithm, 2) a connection algorithm, and 3) a cleaning algorithm. The results of each algorithm are described in the following subsections for each case study.

Filtering Algorithm

Applying the filtering algorithm based on the detected POIs creates two different datasets:

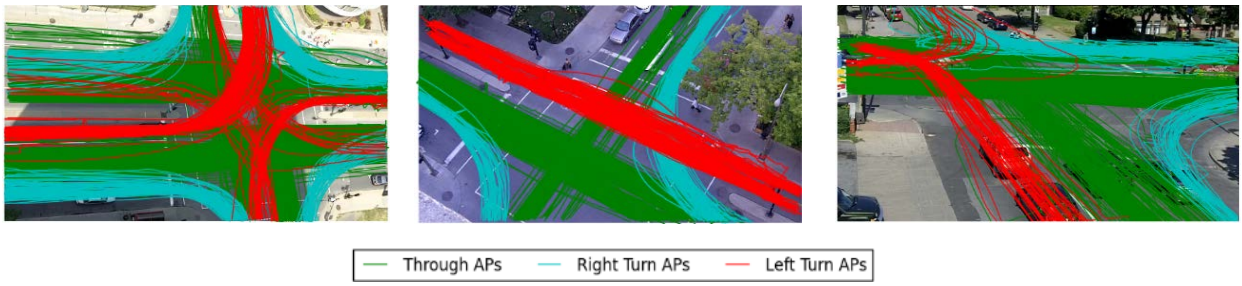
- A dataset of complete trajectories: The complete trajectories form the APs and describe the typical movements in an intersection (e.g. left turn, right turn, and through movements). This dataset will be used in the MP learning phase.
- A dataset of incomplete trajectories: there are three types of trajectories, trajectories starting in an entry zone (set iT1), trajectories ending in an exit zone (set iT2), and trajectories that do not start nor end in an entry or exit zone (set iT3). This dataset is the main input of the connection algorithm.

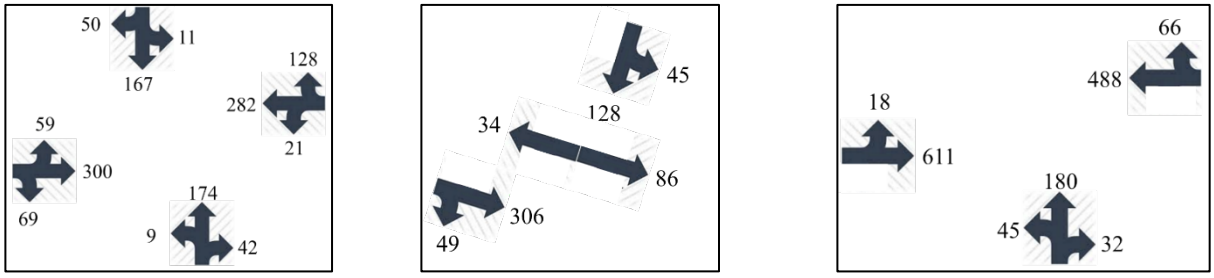
Table 1 summarizes the results of the filtering algorithm in each case study. The Guy intersection is the only intersection with a static occlusion zone, which defines an extra dataset, called the occluded dataset, of incomplete trajectories starting and/or ending in a static occlusion zone. For this case study, these trajectories will not be further studied as the occlusion zone is near the border of the FOV.

Table 1: Filtering algorithm results

Datasets	**Intersections**		
	Guy	**St Marc**	**Atwater**
Vehicle trajectories size	2538	941	2064
Complete dataset	1312	654	1442
Incomplete dataset			
• Starting in an entry zone (iT1)	423	54	125
• Ending in an exit zone (iT2)	601	229	392
• Do not start and end in an entry or exit zone (iT3)	88	4	105
Occluded dataset	114	0	0

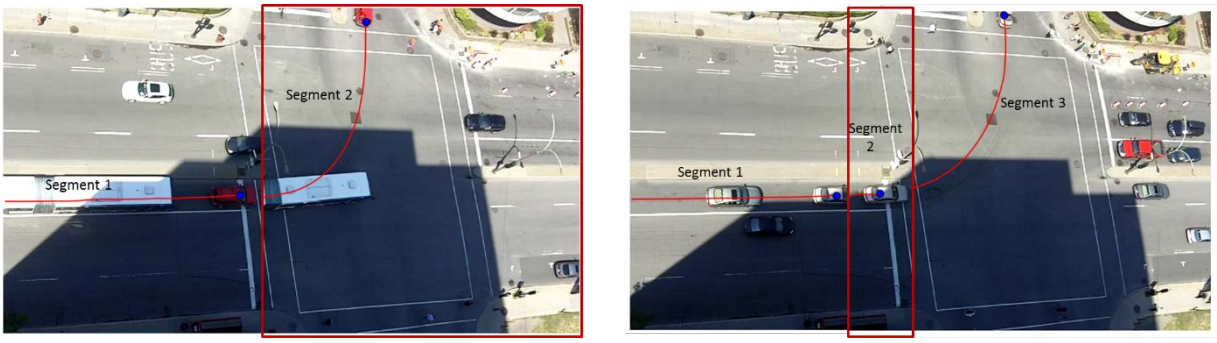
Figure 7**Erreur ! Source du renvoi introuvable.** shows the datasets of complete trajectories (corresponding to the APs) dataset and the traffic volumes of each movement for each case study. The traffic volume is estimated using the number of road users classified as vehicles for each movement per one hour (the studied period). In the St Marc intersection, vehicle trajectories include misclassified cyclists trajectories. More APs were created than expected given the road users types, for observed trajectories from entry to exit zones of different types of road users: these APs were reviewed and correspond to five cyclists moving from vehicle lanes to the cyclist lane or vice versa, as well as a motorcyclist trajectory detected as moving in the cyclist lane with an entry zone detected in the vehicle entry zone. The entry and exit zones could be used to reclassify cyclists that move in the cyclist lane and are classified as pedestrians. This application is out of our research scope as we are interested in vehicle trajectories only.





a) Guy intersection b) St Marc intersection c) Atwater intersection

Figure 7: POI applications: results of the filtering algorithm for the studied intersections (the top row shows APs and the bottom row shows traffic volume (vehicles per hour) for each AP)



a) Connected Trajectories (2 segments) b) Connected Trajectories (3 segments)

Figure 8: Examples of connected incomplete trajectories

Connection Algorithm

The second application of POI is to connect incomplete trajectories. The thresholds for spatial and temporal proximities are chosen as 1.5 meter and 2000 frames (66.7 sec) respectively. The large value of stopping time is to cover the maximum possible waiting time due to the traffic lights.

The experimental results for the Guy intersection show that 464 incomplete trajectories are connected and result in 232 complete trajectories. To test the performance of the connection algorithm, we reviewed the connected trajectories manually by watching the video. In this dataset, we found 226 complete trajectories to be correctly connected, with corresponds to a connection accuracy of 97.4 %. In addition, 118 incomplete trajectories are connected, but still do not end in an exit zone. Likewise, 170 incomplete trajectories are connected, but still do not start in an entry zone. Finally, 21 complete trajectories are merged from three segments (63 incomplete trajectories). Most of these trajectories correspond to a

specific motion behaviour in an intersection: entering the scene then stopping (segment 1), moving slowly motion while waiting for the signal to be green or following another moving vehicle (segment 2), and then moving until the exit from the scene (segment 3). Notably, the second segment might be split into more sub-trajectories and be mainly classified as a pedestrian because of its low-speed. Figure 8 illustrates two examples of connected trajectories. Since the ultimate goal is to use the learnt prototypes for future motion prediction, the trajectories consisting of three or more segments, representing interrupted movements through the intersection, are not considered. However, the trajectories consisting of two segments are used and added to the dataset of complete trajectories.

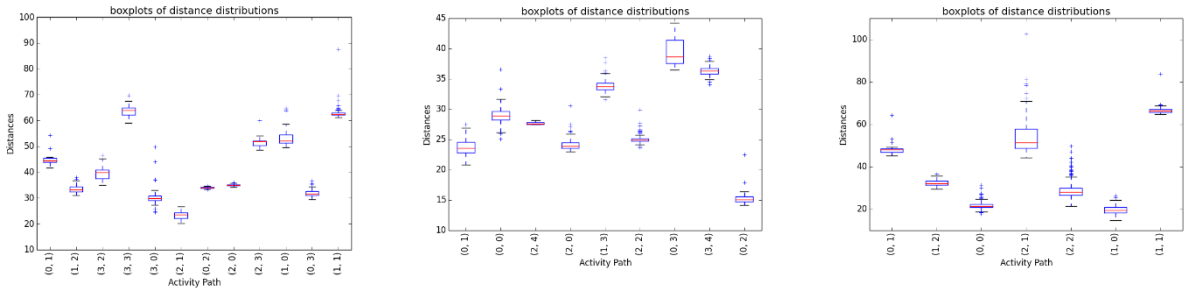
At the St Marc intersection, the connection algorithm succeeded in connecting 306 incomplete trajectories. The accuracy of connected trajectories is 98.7 % with only two trajectories falsely connected. Moreover, 60 incomplete trajectories are connected but still had missing entry or exit zones. For the Atwater intersection, the connection algorithm works less accurately. Only 82 incomplete trajectories are connected for which 70 % are correctly connected. This lower accuracy can be attributed to the low camera angle that affects the performance of the video tracking and grouping algorithms. In conclusion, the detection and tracking algorithms require further investigation for low angle FOVs. However, the use of POI for detecting and connecting incomplete trajectories performed well overall.

Cleaning Algorithm

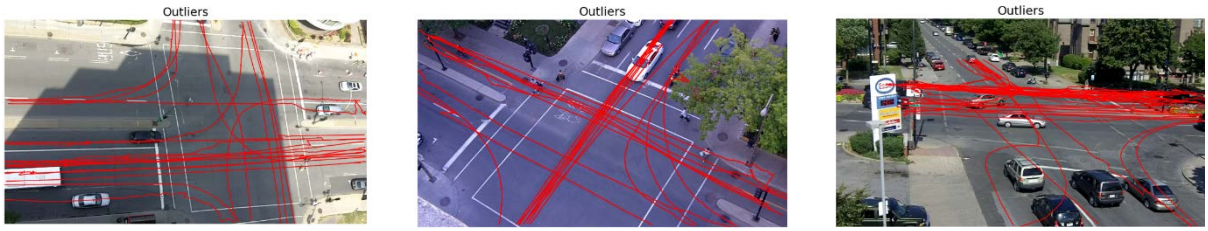
The third application of POIs is the cleaning algorithm that involves the detection of outliers from the dataset of complete trajectories. The outliers are detected based on the analysis of the distribution of travelled distances in each AP. Two types of outliers exist: extreme and mild outliers.

Figure 9 (i) and (iii) show the boxplots of the travelled distance distribution for both mild and extreme cases respectively. The only difference between the two figures is the whisper limits. Figure 9 (ii) and (iv) represent the detected trajectories considered as mild and extreme outliers respectively for each case study. The results for the Guy intersection show that the dataset contains 23 mild outliers and 13 extreme outliers. Noticeably as shown in Figure 9 (a), extreme outliers are very noisy trajectories or represent an extreme unusual movement. Figure 10 shows some examples of mild and extreme outliers. The difference between mild and extreme outliers can be seen in the unusual left turn movement examples. Although each outlier corresponds to an unusual left turn movement, the mild outliers are smoother than extreme ones.

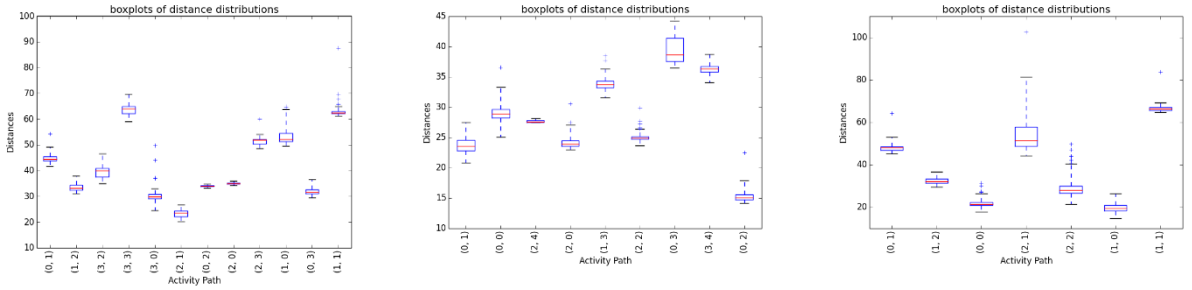
When testing the cleaning algorithm in the St Marc intersection, 20 mild outliers and 14 extreme outliers are detected. As shown in Figure 9 (b), it is noticed that all extreme outliers are connected trajectories with significant noise at the connection location. Mild outliers are a mix of lane change trajectories and connected trajectories. For the Atwater intersection, the dataset contains 37 mild outlier trajectories and 17 extreme outliers, all of which are due to grouping and smoothing errors. Figure 9 (c) shows the detected outliers for both mild and extreme outliers, where the main difference between the types of outliers is found to be the noise levels. Therefore, it is suggested to keep the mild outliers for the motion pattern learning, and to remove the extreme outliers from the complete datasets.



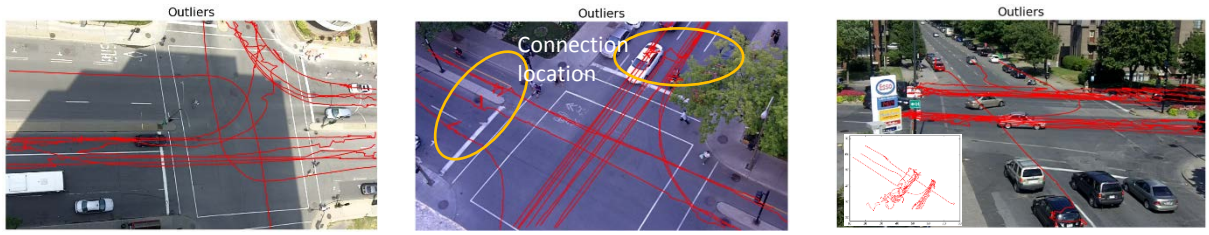
(i)Boxplots of distance distribution per AP (Mild outliers)



(ii)Detected mild outliers



(iii)Boxplots of distance distribution per AP (Extreme outliers)



(iv)Detected extreme outliers

a) Guy intersection b) St Marc intersection c) Atwater intersection

Figure 9: Cleaning trajectories using boxplots based on the distribution of travelled distances in each AP

abnormal overtaking | Mild abnormal left turn | Extreme abnormal left turn | over-grouping with smoothing errors

a) mild outliers b) extreme outliers

Figure 10: Examples of detected outliers based on the travelled distance distributions

Two-Stages MP Learning Experimental Results

Reducing the computational cost

The main challenge of MP learning comes from the need to compute all pairwise similarities between trajectories. The construction of a pairwise similarity matrix is computationally inefficient in both computational time and storage space: the proposed framework addresses precisely this issue. To avoid unnecessary computations of trajectory similarities, two procedures were proposed and their performance is tested as follows:

1. Unlike other clustering algorithms, it is not necessary to compute the similarities between all elements: it is only necessary for each element to compute its similarity to all existing prototypes at the current stage of the algorithm. For example, when a trajectory dataset contains N trajectories, a complete similarity matrix requires traditionally $\frac{(N^2-N)}{2}$ computations. If the number of prototypes is m, the maximum number of necessary computations is obtained from the following equation (computed in the worst case if the first m elements are identified as prototypes). Otherwise, the number of necessary computations $N_{\max computation}$ will decrease according to when prototypes are identified.

$$N_{\max computation} = (N-m) * m + \frac{m(m-1)}{2} = \frac{m(2N-m-1)}{2}$$

This procedure is tested in a sample dataset containing 167 trajectories and grouped into 13 clusters (prototypes). This dataset represents the through movement from north to south in the Guy intersection. The number of computed similarities for the traditional procedure equals 13861, while it is only at most 2080 in the proposed clustering method. The reduction of the number of computed similarities translates into savings of processing time. For this example, the gain is around 85 % of the overall computation time.

2. The complete trajectory dataset is divided into different sub-datasets corresponding to APs. Learning the MPs for each AP separately reduces significantly the computational cost. To test the effectiveness of our algorithm in speeding up the computation for MP learning, a sample of 200 complete trajectories taken from the Guy intersection dataset is used: their complete similarity matrix is constructed based on the traditional method without dividing the dataset first into sub-

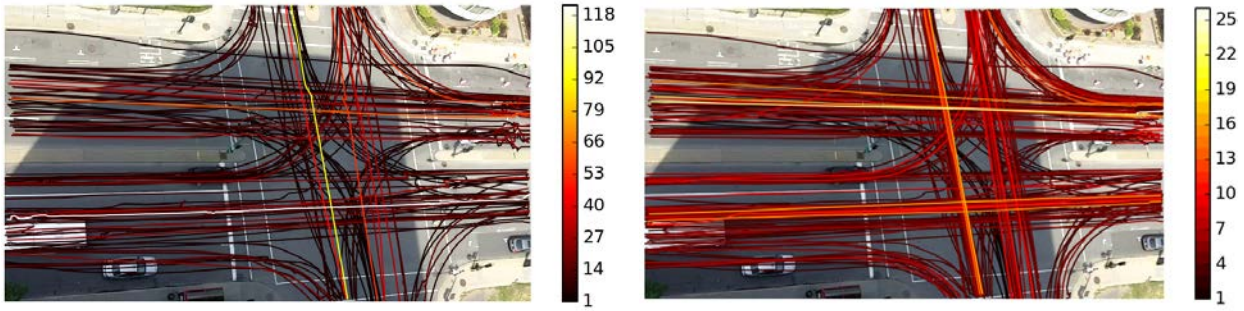
datasets (APs). The run time using the traditional method is 1600 s while the proposed method that divides the dataset into 12 APs takes only 140 s: the gain is more than 90 %. In addition, our procedure is able to reduce the required storage space. For example, we have 1531 trajectories in Guy intersection divided into 12 APs. For each AP, the similarity matrix is computed and saved in a separate file; the total required space of all 12 files is 8.72 MB. However, the required space based on a traditional procedure is 6.6 times larger (57.54 MB). Dealing with small files is easier in importing and exporting data than dealing with a large file. This procedure does not affect the final cluster results since the proposed method only avoids computing similarities that are not needed for clustering. Despite the simplicity of this procedure, it allows to analyze large datasets more efficiently.

MP Learning Results

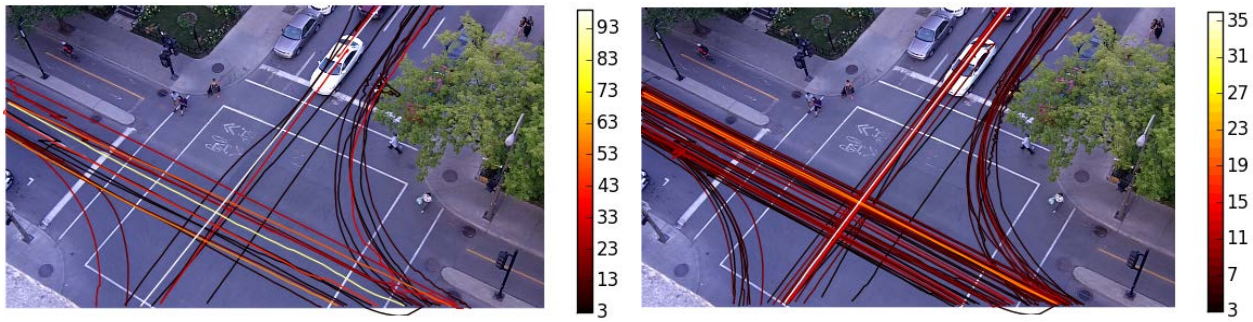
The MP learning algorithm uses the set of complete trajectories after removing the outliers. Table 2 details the size of each dataset and the number of learnt spatial and temporal prototypes for each case study. The parameters for the learning phase are the matching threshold ε and the minimum cluster similarity. These are chosen by trial and error respectively as 1.0 m and 0.75 for spatial information and 1.0 m/s and 0.75 for temporal information. For the ALCSS used for temporal information, δ equals two frames. The learnt prototypes for the two learning stages are presented in Figure 11 for the studied intersections. Although such unsupervised learning is difficult to evaluate, the results of MP learning based on spatial information suggest an acceptable division of the trajectories (as seen in the left column of Figure 11). As expected, results from the second stage based on temporal information provide more prototype trajectories (as seen in the right column of Figure 11). Different types of speed profiles (i.e. associated to temporal prototypes) can be obtained for a given spatial MP as shown for sample of typical movements in Figure 12.

Table 2: Size of datasets and number of learnt prototypes

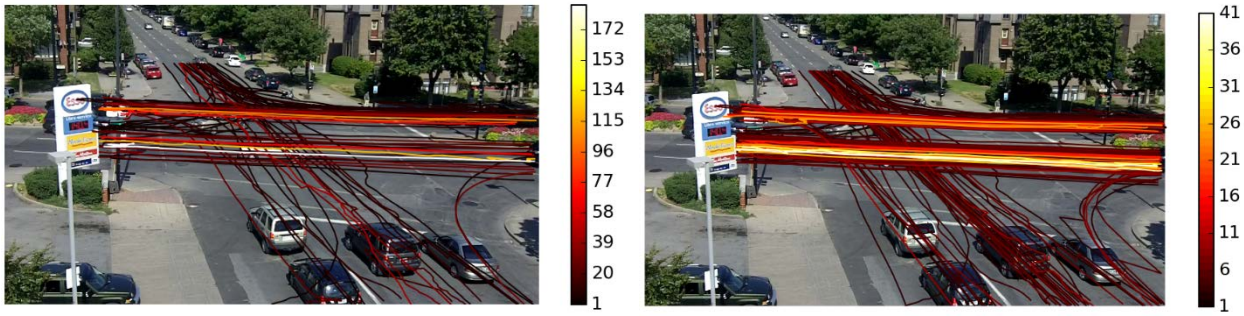
Dataset	**Dataset Size**	**Number of spatial prototypes**	**Number of temporal prototypes**
Guy Intersection	1531	133	260
St Marc intersection	566	28	86
Atwater intersection	1483	71	178



(i) Guy intersection



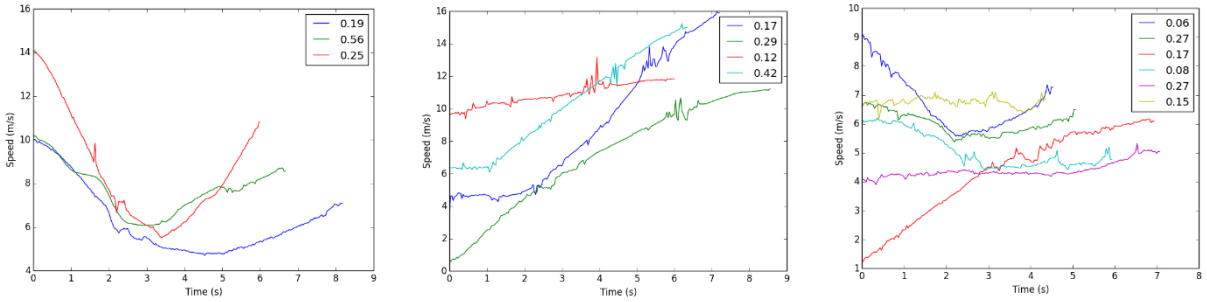
(ii) St Marc intersection



(iii) Atwater intersection

a) First-stage (position-based) prototypes b) Second-stage (speed-based) prototypes

Figure 11: Prototypes representing MPs (color-coded as a sequential black-red-yellow-white to represent the number of trajectories associated to each MP prototype)



Left Turn Movement | Through Movement | Right Turn Movement

Figure 12: Examples of various speed profiles and their percentage in the cluster for a given spatial MP.

Anomaly Detection

Anomaly detection aims to discover anomalous (unusual) trajectories with a low probability of occurrence, while the cleaning algorithm aims to detect the outliers before learning the MPs. These outliers mostly have longer cumulative distances within each AP as previously presented. Including these outliers would lead to two main issues:

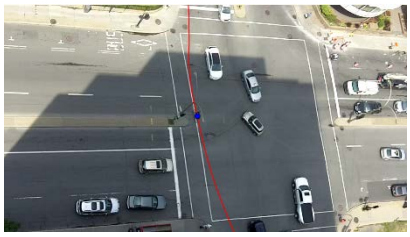
1. The outlier is selected as a prototype and no trajectory is assigned to it, so that it would then be considered as an anomaly in the end. This would lead to the unnecessary computation of similarities between all trajectories and this prototype.
2. In some rare cases, some trajectories are sufficiently similar to the outlier prototype such that this outlier would be considered as a MP prototype and would influence the accuracy of motion prediction later.

Therefore, the objective of the cleaning algorithm is to avoid these situations and remove the outliers before the MP learning. The anomalies have by definition a similar travelled distance as the other MPs in the same AP, but they are sufficiently dissimilar based on the LCSS. For example, a normal movement that rarely happens may be detected by the anomaly detection step while its travelled distance is similar to other trajectories in its AP. Moreover, in the cleaning algorithm, only extreme outliers are removed and mild outliers are generally included in MP learning. Thus, some of the mild outliers can be further detected as anomalies.

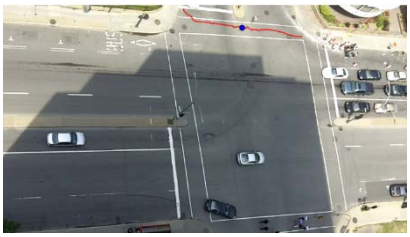
The parameter for anomaly detection is the minimum cluster size that was selected as the largest of 3 elements or 10 % of the dataset size. The dataset is the AP and the spatial MP when learning the spatial and temporal MPs respectively. It should be clear that it is difficult to detect anomalies (if they exist at all) in small datasets. When detected, the anomalies are reviewed manually. In our dataset, there are many sources of anomalies such as road user misclassification, tracking issues, normal but rare movements, and safety problems. Safety problems are violations of traffic law and unsafe movements and examples include:

- a lane change happens by crossing a white solid line which could indicate a side-swipe interaction with or without a collision,
- improper turns: the driver may only turn left or right from the left- and right-most lanes unless signs or marking on the intersection indicate otherwise. Such turns are detected as anomalies,
- excessive speed.

Samples of spatial and temporal anomalies are presented in Figure 13 for Guy intersection.

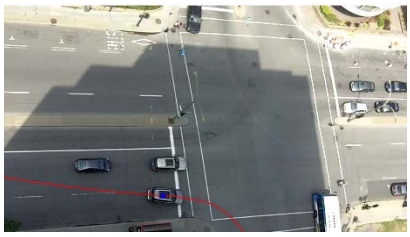


(a) Misclassification (cyclist)

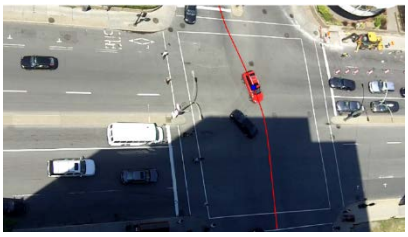


(b) Misclassification (pedestrian jogging)

(c) Abnormal Left Turn

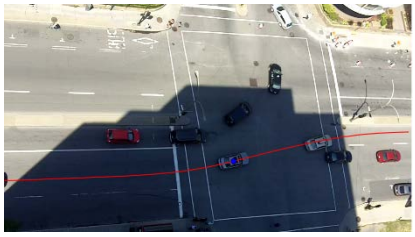


(d) Abnormal Right Turn

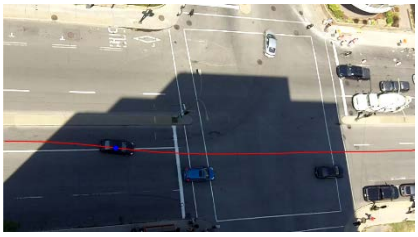


(e) Abnormal Overtaking

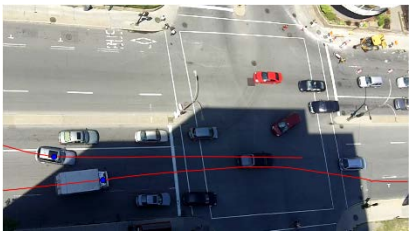
(f) Rare normal movement*



(g) Lane Change

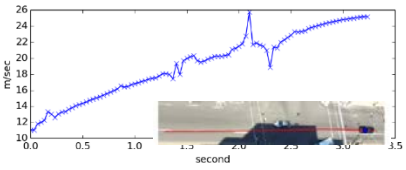


(h) Lane change over solid lane markings

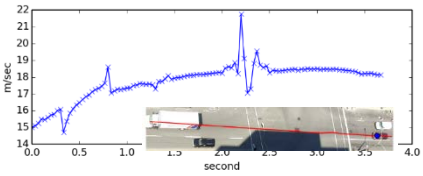


(i) Side-swipe interaction

(j) Connection error



(k) Excessive speed (85km/hr)



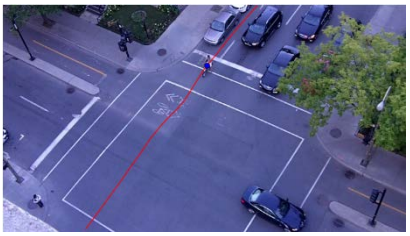
(l) Excessive speed (65km/hr)

* Normal movement that rarely happens (bus lane change to its exclusive lane)

Figure 13: Samples of detected spatial and temporal anomalies (Guy Intersection)

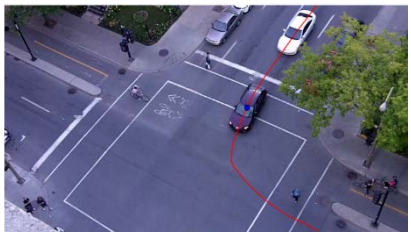
Figure 14 shows samples of detected anomalies in St Marc intersection. Different sources of anomalies are detected such as misclassification, grouping errors and connection errors. In this dataset, significant anomalies are caused by connection errors can be found. This was noted already in the cleaning algorithm. A possible explanation for these errors is the camera FOV where the camera is focused to only cover the

central area of the intersection. In the case of connection errors, they consist of two segments: the first segment represents trajectory where only part of the vehicle appeared, while in the second segment the entire vehicle appeared. Thus, the vehicle centroid in the first segment is falsely located, and conversely, it is correctly located in the second segment. When connecting these two segments, the centroids shift at the connection locations (see Figure 14 f)).



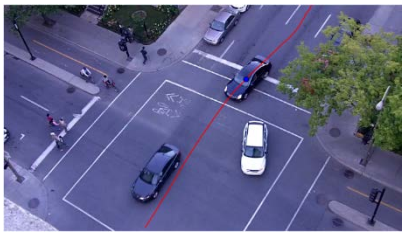
a) Misclassification (Motorcyclist)

b) Over-grouping error

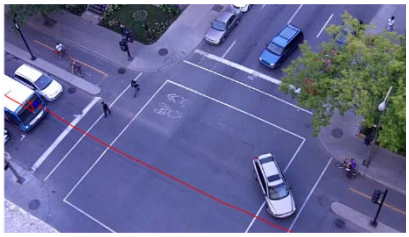


c) Abnormal Left turn

d) Abnormal Right Turn



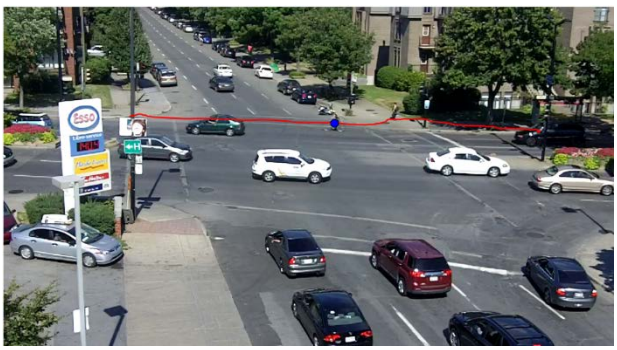
e) Lane Change and overtaking



f) Connection error

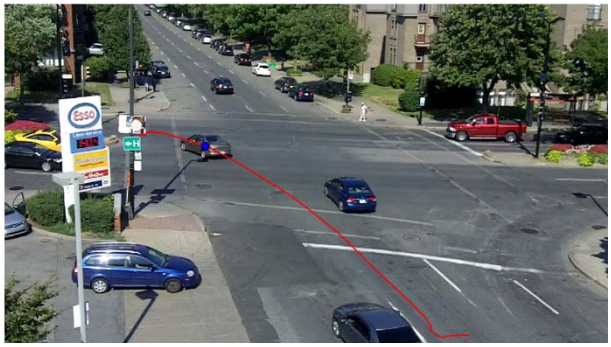
Figure 14: Samples of detected spatial anomalies at St Marc intersection

For the Atwater intersection, there is a large number of anomalies. Most of them are trajectories having an over-grouping problem and some are vehicles projected in the sidewalk. This occurs due to the low angle FOV. By reviewing the detected anomalies manually, we can filter anomalies as summarized in Figure 15.



a) Misclassification (cyclist)

b) Over-grouping error



c) Abnormal Left turn

d) Right turn with overtaking a pedestrian

Figure 15: Samples of detected spatial anomalies at Atwater intersection

Conclusion

This paper discussed a refined and validated MLMP framework for behaviour analysis and anomaly detection. The main input of the proposed framework is the trajectories automatically extracted from video data. A smoothing algorithm is proposed to reconstruct smoother road user trajectories, by estimating the relationship between the road user features and their group. The performance of the algorithm is evaluated on three real case studies visually and quantitatively. The algorithm performs well and can reduce the noise, as measured by the CSJ indicator, up to 97 % for vehicles.

Henceforth, the MLMP is investigated for automated scene interpretation and anomalous behaviour detection. One of the advantages of this framework is to use actual trajectories as prototypes without any pre- or post-processing. The framework is implemented for three real cases that demonstrate the following advantages of the proposed framework:

- It reduces computation up to 90 % for motion pattern learning,
- It reduces the required memory space up to 85 %,
- It connects incomplete trajectories with accuracy between 70 and 97 %,
- It removes the outliers before MP learning,
- It detects anomalous events such as abnormal left turn, improper turns, and excessive speed.

The framework is tested in a traffic environment where it can be used to interpret any scene for surveillance purposes. Moreover, the improvements of computation cost in time and space pave the way to the use of the method in online systems. The limitations of the proposed framework are as follows:

- The selected parameters for MP learning method are the result of an iterative manual process of trial and error. They influence the number of prototypes and need to be carefully selected.
- Anomaly detection is dependent on tracking and grouping robustness, which is highly affected by camera FOV. Anomalies are manually validated and interpreted.

Future work will apply the framework to other types of road users, in particular vulnerable ones. The framework can be used for automatic counting of road users for each movement. In addition, the resulting prototypes can be used to predict realistic future motion based on spatial and temporal information, as explored in [4]. The second phase of this research project will investigate their use for surrogate safety analysis in a forthcoming paper.

Acknowledgement

The authors wish to acknowledge the financial support of the Natural Sciences and Research Council of Canada (NSERC) (Grant No. 402320-2011). The authors declare that there is no conflict of interest regarding the publication of this paper.

References

[1] M. Mohamed and N. Saunier, "Motion Prediction Methods for Surrogate Safety Analysis," *Transportation Research Record: Journal of the Transportation Research Board,* vol. 2386, no. -1, pp. 168-178, December 2013.

[2] N. Saunier and M. G. Mohamed, "Clustering Surrogate Safety Indicators to Understand Collision Processes," in *Transportation Research Board, 93rd meeting*, Washington,USA, 2014.

[3] M. Vlachos, G. Kollios and D. Gunopulos, "Discovering similar multidimensional trajectories," in *Data Engineering, Proceedings. 18th International Conference on IEEE*, 2002.

[4] M. G. Mohamed and N. Saunier, "Behaviour Analysis Using A Multi-Level Motion Pattern Learning Framework," *Transportation Research Record: Journal of the Transportation Research Board,* vol. 2528 (In press), pp. 116-127, 2015.

[5] M. G. Mohamed, "Automatic Behavior Analysis and Understanding of Collision Processes Using Video Sensors," 2015.

[6] D. Makris and T. Ellis, "Learning semantic scene models from observing activity in visual surveillance," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on,* vol. 35, no. 3, pp. 397-408, June 2005.

[7] B. Morris and M. Trivedi, "A Suvey of Vision-Based Trajectory Learning and Analyis for Surveillance," in *IEEE Transactions on Circuits and Systems for Video Technology*, 2008.

[8] B. T. Morris and M. M. Trivedi, "Understanding vehicular traffic behavior from video: a survey of unsupervised approaches," *Journal of Electronic Imaging,* vol. 22, no. 4, pp. 41113-41113, 2013.

[9] W. Hu, X. Xiao, D. Xie and T. Tan, "Traffic Accident Prediction Using 3-D Model-Based Vehicle Tracking," in *IEEE Transactions on Vehicular Technology*, 2004.

[10] C. Stauffer, "Estimating Tracking Sources and Sinks," in *Computer Vision and Pattern Recognition Workshop, 2003. CVPRW '03. Conference on*, 2003.

[11] S. McKenna and H. Nait-Charif, "Learning spatial context from tracking using penalised likelihoods," in *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, 2004.

[12] X. Wang, K. Tieu and E. Grimson, "Learning Semantic Scene Models by Trajectory Analysis," in *In ECCV (3) (2006*, 2006.

[13] B. Morris and M. Trivedi, "Trajectory Learning for Activity Understanding: Unsupervised, Multilevel, and Long-Term Adaptive Approach," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 33, no. 11, pp. 2287-2301, 2011.

[14] M. Nedrich and J. Davis, "Detecting behavioral zones in local and global camera views," *Machine Vision and Applications,* vol. 24, no. 3, pp. 579-605, 2013.

[15] B. Morris and M. Trivedi, "Learning Trajectory Patterns by Clustering: Experimental Studies and Comparative Evaluation," in *Computer Vision and Pattern Recognition*, 2009.

[16] W. Hu, D. Xie, Z. Fu, W. Zeng and S. Maybank, "Semantic-based surveillance video retrieval," *Image Processing, IEEE Transactions on,* vol. 16, no. 4, pp. 1168-1181, 2007.

[17] C. Piciarelli and G. L. Foresti, "On-line trajectory clustering for anomalous events detection," *Pattern Recognit. Lett,* pp. 1835-1842, 2006.

[18] P. Berkhin, "Survey Of Clustering Data Mining Techniques," in *Grouping multidimensional data*, 2006.

[19] R. Xu and D. Wunsch, "Survey of clustering algorithms," *Neural Networks, IEEE Transactions on,* vol. 16, no. 3, pp. 645-678, #may# 2005.

[20] T. W. Liao, "Clustering of time series data- a survey," *Pattern Recognition,* vol. 38, no. 11, pp. 1857-1874, 2005.

[21] A. Y. Ng, M. I. Jordan and Y. Weiss, "On Spectral Clustering: Analysis and an algorithm," in *Advances in neural information processing systems 2*, 2002.

[22] N. Saunier, "Traffic Intelligence software," 2015. [Online]. Available: https://bitbucket.org/Nicolas/trafficintelligence.

[23] N. Saunier and T. Sayed, "A feature-based tracking algorithm for vehicles in intersections," in *Computer and Robot Vision, 2006. The 3rd Canadian Conference on*, 2006.

[24] C. Tomasi and T. Kanade, Detection and tracking of point features, School of Computer Science, Carnegie Mellon Univ. Pittsburgh, 1991.

[25] N. Saunier, T. Sayed and C. Lim, "Probabilistic Collision Prediction for Vision-Based Automated Road Safety Analysis," in *The 10th International IEEE Conference on Intelligent Transportation Systems*, Seattle, 2007.

[26] M. Vlachos, G. Kollios and D. Gunopulos, "Discovering similar multidimensional trajectories," in *Proceedings of the 18th IEEE International Conference on Data Engineering*, 2002.

[27] N. Cruz-Ramírez, H.-G. Acosta-Mesa, R.-E. Barrientos-Martínez and L.-A. Nava-Fernández, "How good are the Bayesian information criterion and the minimum description length principle for model selection? A Bayesian network analysis," in *MICAI 2006: Advances in Artificial Intelligence*, Springer, 2006, pp. 494-504.