Tracking All Road Users at Multimodal Urban Traffic Intersections

Jean-Philippe Jodoin, Guillaume-Alexandre Bilodeau and Nicolas Saunier

***Abstract*—Because of the large variability of road user appearance in an urban setting, it is very challenging to track all of them with the purpose of obtaining precise and reliable trajectories. However, obtaining the trajectories of the various road users is very useful for many transportation applications. It is in particular essential for any task that requires higher level behaviour interpretation, including new safety diagnosis methods that rely on the observation of road user interactions without a collision and therefore do not require to wait for collisions to happen. In this work, we propose a tracking method that has been specifically designed to track the various road users that may be encountered in an urban environment. Since road users have very diverse shapes and appearances, our proposed method starts from background subtraction to extract the potential *a priori* unknown road users. Each of these road users is then tracked using a collection of keypoints inside the detected foreground regions, which allows the interpolation of object locations even during object merges or occlusions. A finite state machine handles fragmentation, splitting and merging of the road users to correct and improve the resulting object trajectories. The proposed tracker was tested on several urban intersection videos and is shown to outperform an existing reference tracker used in transportation research.**

***Index Terms*—Computer Vision, Multiple Object Tracking, Road Users, Transportation Engineering, Road safety**

I. INTRODUCTION

AMONG the various data collection technologies for road transportation, video tracking allows acquiring more easily, at a lower cost and more accurately, larger amounts of data than what could be done previously by hand. This leads to advances that can only be achieved by mining large amounts of observational data. There is a particular interest in road safety so that diagnosis methods may become more proactive [1]. Instead of waiting for accidents to occur, these proactive methods rely on the observation of road user interactions such as near misses and their safety evaluation using indicators ("surrogate measures of safety") such as time to collision that can be computed from road user trajectories [2]. Observing road user interactions over a few hours or days is thought to be sufficient for road safety diagnosis that would otherwise take years to perform based on historical accident records.

Although a large body of research has addressed the tracking problem, tracking road users at busy urban locations, in particular intersection, brings its own challenges. Indeed, recent tracking methods (e.g. [3], [4]) that compute trajectories by stitching target detections are not applicable because of the large variety of road users. That is, the shape and appearance of vehicles and people are very diverse which makes it very difficult to design a universal road user detector. Furthermore, the appearance of some road users can be sometime unpredictable (e.g. modified vehicles, cyclists, wheelchairs). As a result, only motion information can be exploited to detect the road users, either using optical flow or background subtraction. In both cases, the detected objects are often fragmented or merged.

With the aim of designing a tracking method to collect trajectory data for road safety analysis at busy urban locations, this paper describes a fully automatic multiple object tracker, coined Urban Tracker, that is adapted to track various *a priori* unknown road users. It should be noted that the tracker does not have to operate online in real time. Starting from background subtraction [5], [6], [7], [8], we propose a method that is based on tracking the resulting foreground blobs of pixels. Each blob is modelled by a collection of keypoints. Data association is performed from frame to frame, and a finite state machine (FSM) corrects the associations by handling blob merging, splitting and fragmenting. For increased precision in the location of the road users, keypoint locations are used to interpolate the position of occluded or split objects. Experiments show that the proposed method outperforms a reference tracker, "Traffic Intelligence", used in previous road safety studies [9], [2] for large road users like vehicles and small deformable road users like pedestrians. This paper extends a previous version of this work presented at a conference [10] with more detailed explanations and more experiments, in particular using cross-validation on the five video scenes to evaluated the trackers' robustness, which has never been carried out in video tracking to the best of our knowledge.

The paper is structured as follows. Section II presents the background and related works, section III motivates our approach and describes our method, section IV presents the experimental results, and section V concludes the paper.

II. BACKGROUND AND RELATED WORKS

Multiple object tracking (MOT) is a very active topic. Most recent works have focused on the data association problem, which is one of the most fundamental problem in MOT [11],

J.-P. Jodoin, G.-A. Bilodeau are with the Computer and Software Engineering Department of Polytechnique Montréal, Montréal, QC, H3C 3A7, Canada e-mail: jpjodoin@gmail.com, gabilodeau@polymtl.ca

N. Saunier is with the Civil, Geological and Mining Engineering Department, Polytechnique Montréal, Montréal, QC, H3C 3A7, Canada e-mail: nicolas.saunier@polymtl.ca

Manuscript received XXX; revised XXX.

This work was supported by scholarships from Fonds de Recherche Quebec - Nature et Technologies and Natural Sciences and Engineering Research Council of Canada (NSERC), and partially supported by the NSERC discovery grants No. 311869-2010 and 402320-2011.

[3], [4], [12]. In these works, it is assumed that the inputs of the tracker are object detections. Furthermore, it is assumed that some detections may be missing and some may be false. The problem is then to link the detections in such a way as to ignore false detections and still be able to correctly associate detections over time even if some are missing for a few frames. In its most basic form, data association is performed on a frame-by-frame basis. Greedy or even the well-known Hungarian algorithm can be used to find the optimal associations given some metrics on the appearance, position or speed of the targets [12]. These methods can be applied online. More sophisticated methods use min-cost network flow algorithms to find associations over a time window [3], [4] and are usually applied in offline mode. These methods ignore the possibility that objects may be fragmented. In the case of our application, this assumption does not hold because the appearance of the road users may be unpredictable (e.g. modified cars, cyclists, pedestrians). As such, a universal classifier-based object detector cannot be used to detect all road users. It is therefore necessary to use another type of detection method that may have the disadvantage of producing fragmented detections.

In this context, two alternative multiple object class detection methods are typically used in tracking. The first method relies on optical flow and groups features that move at similar speed and direction in the video [13], [9], [14], [15], [16]. The second method relies on background subtraction [17], [18], [19], [20], which produces detections that are based on temporal change in the image. In the first case, textureless objects might not be properly detected or demarcated, while in the second case, temporary stopping objects might create false detections.

Optical flow has been used in several papers for tracking objects of multiple classes. Luo and Bhandarkar [21] used optical and elastic matching to match object regions with the help of a Kalman filter to predict object positions. Aslani and Mahdavi-Nasab [22] relied only on optical flow for both object detection and tracking. Shin *et al.* [23] extract the objects based on optical flow, and then for every objects, feature points are extracted and matched. They include in their method an occlusion handling mechanism that predicts hidden feature points based on their position with respect to the centre of the object. However, they do not consider object fragmentation. Beymer *et al.* [14] used a corner detector and a Kalman filter to track objects that have been detected based on motion on entry/exit areas. Coifman *et al.* [13] used the Kanade-Lucas-Tomasi (KLT) tracker [24] to find good feature points and to track them. The feature tracks are then grouped based on common motion constraints by computing the difference between the minimal and the maximal distance between two tracks. Saunier *et al.* [9] adapted the work of Coifman *et al.* [13] to track all types of road users in outdoor urban intersections by detecting continuously new features and adding them to current feature groups. The challenge is to find the right parameters to segment objects moving at similar velocities, while at the same time not over-segmenting smaller non-rigid objects such as pedestrians. Recently, Wang *et al.* [16] perform real-time tracking and counting applying the KLT tracker only on the foreground. While these methods give reasonable results, they cannot segment nearby objects that move at the same speed. Furthermore, the quality of the localization of the target depends on the positions of the detected KLT feature points. If all the points are located for example at the top part of the target, its position will be biased toward the top part of the video frame, which will increase the positional error if projecting the coordinates to a ground plane because of perspective.

Background subtraction is also the basis of several multiple object class trackers. Stauffer and Grimson [25] proposed a tracking method that first extracts objects using the Gaussian Mixture Model background subtraction algorithm, and then a multiple hypothesis tracking method and a Kalman filter are used to find correspondences between blobs using position and size. Fuentes and Velastin [26] proposed a system that performs simple data association via the overlap of foreground blobs between two frames. In addition to matching blobs based on overlap, Torabi *et al.* [17] validate the matches by comparing the histograms of the blobs and by validating the data association over short time windows using a graph-based approach. Batista *et al.* [27] match objects from frame to frame using the auto-correlation of 8×8 pixel blocks obtained from foreground blobs. The small blocks allow handling partial occlusions. Song and Nevatia [28] match blobs using inferred 3D models and a Kalman filter. Jun *et al.* [19] used background subtraction to estimate the vehicle size. They used a watershed segmentation technique to over-segment the vehicle. The over-segmented patches are then merged using the common motion information of tracked feature points. This allows to segment vehicles correctly even in the presence of partial occlusion. Kim *et al.* [20] combines background subtraction and feature tracking approach with a multi-level clustering algorithm based on the Expectation-Maximization (EM) algorithm to handle the various object sizes in the scene. The resulting algorithm tracks various road users such as pedestrians, vehicles and cyclists online and the results can then be manually corrected in a graphical interface. Finally, Mendes *et al.* [15] also combine background subtraction and optical flow to deal with objects of varying size: the approach is evaluated in part on data made available for the first version of this work [10].

The major issue with the background subtraction approach is fragmentation that can be quite frequent even in the most recent background subtraction methods [5]. We propose to alleviate this problem by combining foreground blobs and keypoint tracking. The feature points allow greater robustness to occlusions while the foreground blobs allow more precise localization, as the foreground blob regions are used as referentials to represent the positions of the points. During occlusion, the position of the points in the foreground blob referential can be used to estimate the exact position of the centre of the target. Fragmentation is handled using information like common motion and proximity, similarly to the method used in [17].

In contrast to the work in [20] that uses EM to cluster feature trajectories in objects, our method relies on background subtraction for the same task. Features inside a common region are grouped instead of features with common motion. This allows us to discriminate between vehicles having common trajectory and speed, and to obtain better localization. Moreover, this feature grouping procedure allows us to achieve better tracking results on non-rigid object than previous feature-based methods relying on common motion criteria like [9]. Similarly to [29], a FSM and inter-frame blob associations are used to handle the occlusion and segmentation problems of the background subtraction, and we add a hypothesis state to avoid false object detections.

III. Method

A. Motivation and overview

In the context of a high probability of object fragmentation and merging, the proposed method is based both on blob and point tracking. Blobs allow us to better locate the position of objects, to track objects with only few keypoints and to track *a priori* unknown objects. Points allow us to better discriminate between objects and to handle occlusion, splitting, merging and fragmentation more robustly. The overview of our tracking method is presented in Algo. 1. At first, foreground blobs are extracted using the ViBe background subtraction method [6] (line 3). These foreground blobs are then modelled using their size and position, as well as FREAK keypoints [30] (line 4). After that, "low-level" tracking is performed to match blobs across two consecutive frames (line 5). These blob matches correspond to what are called short tracklets (s-tracklets). To handle merging, splitting and fragmentation, s-tracklets are analyzed and then assigned to object tracks (line 6), after which track properties are updated (line 7 to 14). This part is called high-level tracking and is based on the notion of track and track state. A track corresponds to the trajectory of a physical object in the real world. One or many blobs at each frame may represent a physical object. The track state is used for s-tracklet assignment and represents the track life cycle in the scene (entering, exiting, visible, etc.). The FSM is essential to determine when and how an s-tracket can be added to another track. This step uses the s-tracklet information to resolve the ambiguity between the assignments of s-tracklets to tracks. At the end of the algorithm (line 16), the final tracks are outputted. They may differ from the ones computed at each frame because they are adjusted when merging and splitting are discovered.

B. Foreground blob extraction

Foreground blobs are extracted using the ViBe background subtraction method [6]. This method was selected because it is fast and it is among the top performers on the original *changedetection.net* benchmark [5]. To decrease noise, we apply a Gaussian blur with a 5x5 kernel before the background subtraction and smaller holes in the foreground are filled with morphological operations after the background subtraction. Furthermore, we have slightly modified the ViBe method to better handle intermittent object motion that occurs often in urban scene since vehicles will park on the side of the street or stop at red lights (see [10] for the details). Foreground objects that are not moving are removed.

Algorithm 1 Overview of Urban Tracker

```
 1: procedure URBAN TRACKER
 2:     for each frame do
 3:         Extract foreground blobs (section III-B)
 4:         Compute model for each blob (section III-C)
 5:         Match blobs with those from previous frame (s-tracklets construction) (section III-D)
 6:         Assign s-tracklets to object tracks
 7:         for each s-tracklet do
 8:             if s-tracklet is assigned to object track then
 9:                 Update track model
10:                 Update track state
11:             else
12:                 Create new track in hypothesis state
13:             end if
14:         end for
15:     end for
16:     Return final object tracks
17: end procedure
```

As usual, once the foreground pixels are detected, connected components are computed [18], [31] to generate the blobs and small blobs are filtered out.

C. Blob and track model

The blob model is composed of the blob size, the blob position and a collection of keypoints located inside it. The keypoints are used for data association, and the size and position of the blobs are used for computing the object tracks. The keypoints are extracted using BRISK [32] with 3 octaves for some scale invariance and a detection threshold of 10 to obtain a relatively large number of points. The keypoints are then described using FREAK [30] for speed, rotation and scale invariance, and good distinctiveness. We use 3 octaves and a scaling pattern of 22 pixels. Scale invariance is particularly useful for long occlusions where the size of the object before and after is quite different as it happens often in perspective projections.

A track is modelled as a concatenation of blob models at successive instants (frames). If keypoints inside a track are not matched with keypoints from recent blob models, they are deemed not useful and deleted.

D. Building s-tracklets

Associating blobs into s-tracklets can be considered as low-level tracking since it deals with the temporal association between blobs in two consecutive frames. That is, at this point, we are not concerned with relating the new blobs with previous tracks. We are just building short tracklets between two consecutive frames that will be later connected to the currently tracked objects. In this work, we define a short tracklet (or s-tracklet) as a track segment between two blobs in consecutive frames that are matched based on appearance or overlap. The blob at frame $t-1$ is the source blob of the s-tracklet and the blob at frame t is the destination blob. We are interested in the following blob tracking events: blobs entering the scene, blobs exiting the scene, a blob associated to a single previous blob, blobs disappearing, blobs merging and blobs splitting. These events are represented inside a FSM.

The first step is to construct the s-tracklets by calculating the correspondence between the descriptors of the keypoints located inside the set of blobs B_{t-1} in the previous frame and the set of blobs B_t in the current frame using the Hamming distance. The ratio and the symmetry tests defined in [33] are run for each pair of keypoints to filter bad matches. The ratio test consists in computing the distance ratio between the best match and the second best match in order to filter out less distinctive keypoint matches. The symmetry test consists in verifying if the matching is mutual for the two keypoints in the pair. To avoid bad matches, s-tracklets with fewer than four matched keypoints in their blobs are removed.

In some cases, blobs will not have enough keypoints matches to build the s-tracklets. This can happen for small objects with sudden appearance changes like pedestrians, or on the border of the image where FREAK cannot be applied because of its spatial extent. In these cases, s-tracklets are built based on blob overlap. The area of overlap between blobs is calculated at the pixel level (not with bounding boxes). If a blob overlaps with more than one blob, the association is done with the blob that has the largest area of overlap.

A blob at time t that is not matched to any blob in the previous frame is considered as a potential new track and is labelled as an unassociated blob.

E. Track construction

Track construction consists in associating the s-tracklets with the current object tracks. Therefore, this part requires to interpret the relationships between the s-tracklets and tracks in order to link them. For example, two s-tracklets with the same source blob but having different destination blobs in the next frame represent a case of blob splitting. In some of these cases, the current tracks have to be modified based on the new s-tracklet information (e.g. when a split is discovered). Also, some s-tracklets are the result of fragmentation and have to be removed. The tracks are updated based on s-tracklet information using the FSM presented in Fig. 1. The FSM is applied to a track. Transitions to change the state of a track are triggered by the s-tracklets interpretation.

An s-tracklet may be linked to a track if the model of its destination blob is similar to the model of blobs stored in the track. S-tracklet keypoints descriptors are added to the model when they are linked together.

1) Track states:
A track stays in the *normal* state as long as an s-tracklet with unique (not shared) source and destination blobs can be added to it at each frame (transition a_6), i.e. when an object is tracked without ambiguity. While in the *normal* state, a track can move to the *lost* state (transition a_5) if it cannot be linked to an s-tracklet in a given frame. It will go back to the *normal* state the next time it can be linked to an s-tracklet (transition a_{11}). However, if the track remains in the *lost* state for more than a given number of frames, this track is saved and deleted from active tracks (transition a_{12}).

When an unassociated blob B_t^u is detected in a frame, two actions are possible:

1) If B_t^u can be linked to a track in the lost state, this track moves back to *normal* state as stated previously (transition a_{11});

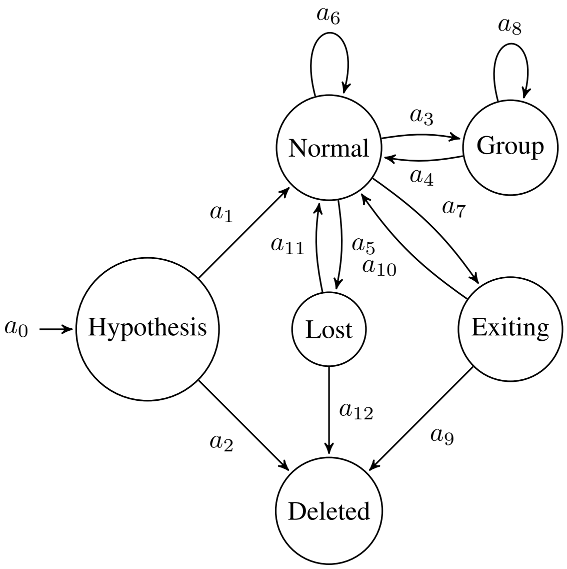


Fig. 1. Tracking FSM

2) if B_t^u cannot be linked to any active tracks, a new track is created in the *hypothesis* state (transition a_0). If no s-tracklet can be added to this track for more than 3 frames, it will be deleted without being saved (transition a_2). This allows us to remove noisy and unstable tracks. If s-tracklets can be added to this new track for 3 consecutive frames, the state of the track will be changed to *normal* (transition a_1). Note that tracks in *hypothesis* state are merged to spatially close tracks to reduce object fragmentation (see section III-E4).

Note that each time an s-tracklet is added to a track, the model of the destination blob of the s-tracklet is saved in the track to have a history of the position and appearance of the object during the complete track.

When s-tracklets share source or destination blobs, we are in the case of merging, splitting, or fragmentation. More analyses are required to link them to tracks. These situations may occur when

1) S-tracklets share a destination blob. This corresponds to merging. In that case, all the tracks that can be linked to the source blobs of these s-tracklets enter the *group* state (transition a_3);
2) S-tracklets share a source blob. This corresponds to splitting. In most cases, this happens to tracks already in the *group* state and the question is to identify which tracks have split from the group. Some tracks will stay in the *group* state (transition a_8), while others will go back to the *normal* state (transition a_4) (see section III-E2). If the split occurs for a track that was not in *group* state, it may either be a case of fragmentation or a case of under-segmentation (two objects entering the scene together for instance). In that case, a new track will have to be created in the *normal* state (see section III-E3).

There is also a state for tracks that end when objects exit the scene. This is necessary to handle objects that may overlap as one leaves and the other enters simultaneously. If a track in *normal* state reaches the border of the scene, it is changed to the *exiting* state (transition a_7). If it cannot be linked to an s-tracklet in the next frame, it will go in the *deleted* state (transition a_9). If the track changes direction and is no longer on the border, we verify that the model is still the same (three keypoint matches are required for the model to be considered the same). This is caused again by the lack of FREAK keypoints around the border: s-tracklets are constructed solely based on overlap, without appearance information. At the moment that appearance information starts to be available again for building s-tracklets, we must therefore check if we are still tracking the same object, or if the object exited and we are now tracking a new one. If it is the same object, the track state changes back to *normal* (transition a_{10}). If it is a different object, the track is deleted from active tracks and a new track is created. This is detailed in section III-E5.

In the following, we explain how the more complex state transitions are handled.

2) Splitting of tracks in group state:

If a track in *group* state is detected as splitting from the group, it will go back to *normal* state (transition a_3). During the time a track is in *group* state, its position and its bounding box can be interpolated by using the FREAK keypoints even if it is occluded. The FREAK keypoints are matched between the destination blob of the group s-tracklet and the FREAK keypoints that have been added to the track in the past. This process is illustrated on Fig. 2.

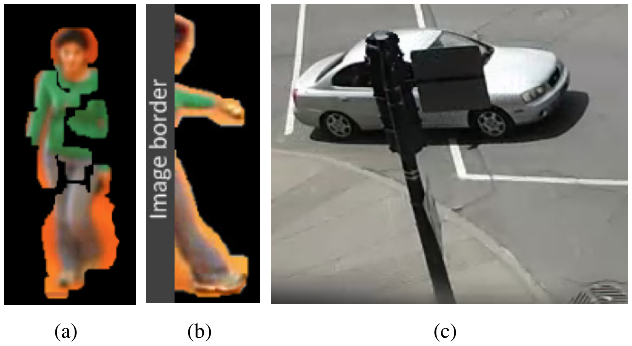
Fig. 2a) represents a previous observation in a track with keypoints and known bounding box. In Fig. 2b) a group blob with the same keypoints is shown. For all of the keypoints in a), we calculate their position relative to the centroid of the bounding box. These relative positions are added to the current positions of the matched keypoints in b) in order to get several estimates of the centroid position. In order to reduce noise, we select the median value on the x and y axis to get the final estimation of the box centroid. The dimension of the bounding box is also the median size of previous observations (before the object became part of the group of objects in the group blob). The final bounding box of each object can be seen on Fig. 2c).

3) Splitting of a track in normal state:

A track in *normal* state may split if it contained multiple objects that we could not detect individually when this track was first constructed. This may happen because multiple objects were occluding each other as they entered the scene. In this case, new tracks will be created in *normal* state and the model of each destination blob of the s-tracklets is used to retrieve a more exact track of the objects before the split. The same interpolation process described in the previous section is used in reverse to interpolate the object positions to obtain the tracks. Because of such track modifications using new evidence, the final object tracks may be different from the tracks obtained at each frame.

4) Fragmentation handling:

S-tracklets may share a source blob for reasons other than actual track splitting because of the following fragmentation scenarios: 1) errors in the background subtraction (Fig. 3a), 2) concave objects entering the scene (Fig. 3b), or 3) static objects occluding part of the scene (Fig. 3c).



(a) (b) (c)

Fig. 3. Fragmentation examples. a) A pedestrian is split in three parts, b) as a pedestrian walks into the scene, his body is split in three parts because his torso is not visible, so the arm, leg and head seem disconnected, and c) an object is split in half because of the pole in the foreground

To resolve cases 1) and 2), we use the *hypothesis* state as a temporary buffer before creating permanent tracks (transition a_1). This gives time to the background subtraction to stabilize and since tracks in the *hypothesis* state are automatically merged with spatially close tracks, the track of a single object is less likely to be split into several parts. Situation 3) is solved using spatial information and by analyzing the s-tracklets. Before splitting a track, we verify that the s-tracklets causing the split diverge enough spatially by dilating their destination blobs' bounding boxes by a factor (10 % in our case). If there is an overlap between the destination blobs of the s-tracklets, the split is delayed since the splitting tracklets are still too close to conclude that they correspond to two separate objects. The use of the bounding boxes and of a dilation factor makes this test more robust, which is crucial in urban scenes where there is considerable variation in object size.

5) Exiting objects:

Exiting objects are problematic because their blob may merge with entering objects (as seen in Fig. 4). The absence of FREAK keypoints near the border of the image complicates this issue further because the s-tracklets are more error prone in this case. Since this situation is quite common in urban scenes, an *exiting* state is introduced. As soon as a track reaches the border of the image, its state changes to *exiting*. From that point on, three situations may occur: 1) the object leaves the scene, 2) the object does not leave the scene and starts going back toward the middle of the scene, 3) the object leaves the scene and its blob is merged with an entering object. Case 1 requires no particular processing. The critical issue is to distinguish case 2 from case 3. If the object leaves the border and goes toward the middle, it will still be possible to match it using FREAK keypoints. Tracks in this situation need to have at least three matching keypoints to validate that they are still associated to the same physical object (case 2), or else case 3 is assumed and a new track is created. In case 3, the history of the new track is recovered from the previous track model (the one that exited the scene). In order to estimate the frame at which the group blob started to represent more the new object and less the exiting one, the group blob in the *exiting*

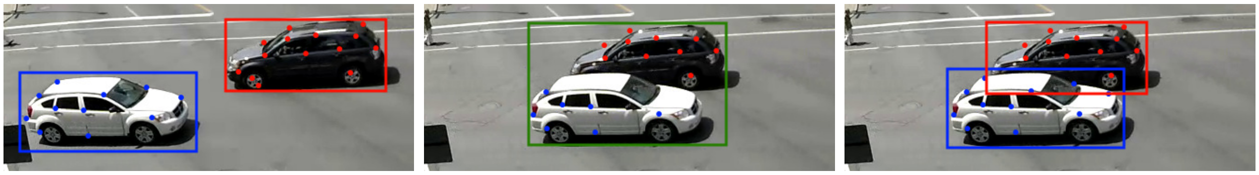

(a) (b) (c)

Fig. 2. Blob estimation process: a) Past bounding boxes with points b) Group blob with current points c) Estimated bounding boxes

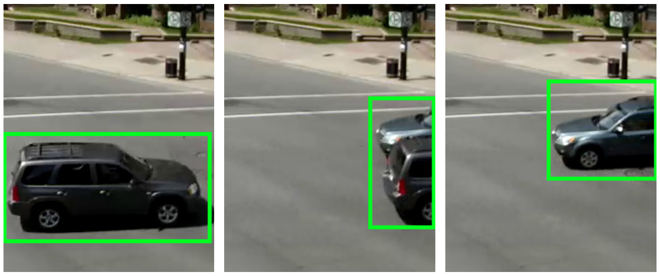

(a) (b) (c)

Fig. 4. Identity change problem when an object exits and another enters at the same time. These are cropped image of the right side of a scene: a) the dark SUV leaves the scene; b) the green car is part of the same group blob as the dark SUV; c) the green car takes erroneously the identity of the dark SUV

state is analyzed: the frame that marks the entry point is the one when the object blob was the smallest since it indicates the instant when the blob size started to be modified by the entering object. This allows us to separate the entering object from the exiting object.

IV. Experimental validation

To validate our proposed tracker (called Urban Tracker), it is compared to an open source reference tracker used in transportation applications called Traffic Intelligence. It is based on the KLT tracker to obtain point tracks and on common motion constraints to group point tracks into object tracks [9][1]. Using five video sequences of road intersections in cities, the tracking performance of Urban Tracker (UT) and Traffic Intelligence (TI) are compared globally and for different road users. The parameter sensitivity of both trackers is also investigated using cross-validation.

A. Evaluation Methodology

Five videos of urban road intersections were used for the validation of the algorithm[2]. Each video was captured at a different intersection with a different viewpoint. For all the videos, each object was annotated as soon as it started moving until it left the scene. An object leaving the scene and reentering later is considered as a different object. All objects were annotated (id, centre position, and bounding box corners) even if they were only partially visible. Also, a mask is applied on the Sherbrooke, Roundabout and Rene-Levesque (Rene-L. in the tables) video to specify a region of interest where tracking is performed. A frame is shown for each video sequence in Fig. 5 and the specifications of each annotated sequence are listed in Table I. The pedestrians in the Rene-Levesque video were not annotated because they are too small.

TABLE I
Specifications of the annotated videos: #frames is the length of each video in frames, #Car, #Cyc. and #Ped. are respectively the number of motorized vehicles, cyclists and pedestrians in the annotated video, and #Simul. is the maximum number of simultaneous objects in the scene

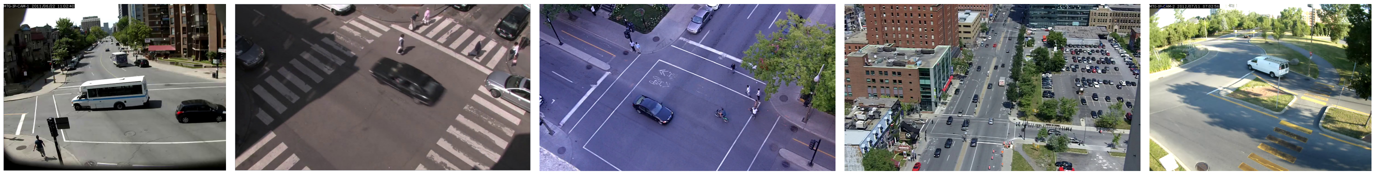
Video	Resolution	#frames	FPS	#Cars	#Cyc.	#Ped.	#Simul.
Sherbrooke	800x600	1001	30	15	0	5	7
Rouen	1024x576	600	25	4	1	11	8
St-Marc	1280x720	1000	29	7	2	19	14
Rene-L.	1280x720	1000	29	29	2	0	20
Roundabout	800x600	4000	15	54	0	0	4

The CLEAR MOT metrics are used to evaluate the quality of the tracking by comparing the tracker output with the manually created ground truth [34]. It consists of the multiple object tracking accuracy (MOTA) and the multiple object tracking precision (MOTP). MOTA is the overall ratio of the number of correct detections of each object over the number of frames in which each object appears (in the ground truth): it is defined as $1 - \sum_t (m_t + fp_t + mme_t) / \sum_t g_t$, where m_t, fp_t, mme_t and g_t are respectively the number of misses, over detections (false positives), mismatches and ground truth objects for frame t. MOTP measures the average precision (distance) of the instantaneous matches: it is defined as $\sum_{i,t} d_t^i / \sum_t c_t$, where d_t^i is the distance between the pair i of associated ground truth and tracker object at frame t and c_t the number of associated pairs in frame t. In addition, the MT (mostly tracked) and ML (mostly lost) metrics are used to report also the performance in terms of road users, with the same criteria as in [35]. MT and ML are the proportions of ground truth objects that are detected respectively for at least 80 % and for less than 20 % of their existence.

These metrics depend on matching the tracker output to the ground truth, which can be done in various ways depending on the type of tracker output. For a tracker yielding a bounding box such as UT, the distance can be the overlap of the bounding boxes of the tracker object and ground truth, while for a tracker yielding a centroid position such as TI, the distance is simply the Euclidean distance between the centroid of the tracker object and the center of the ground truth bounding box. The second definition is used in this study since TI is not designed to generate object bounding boxes. In addition, the maximum distance for a tracker object to match a ground truth

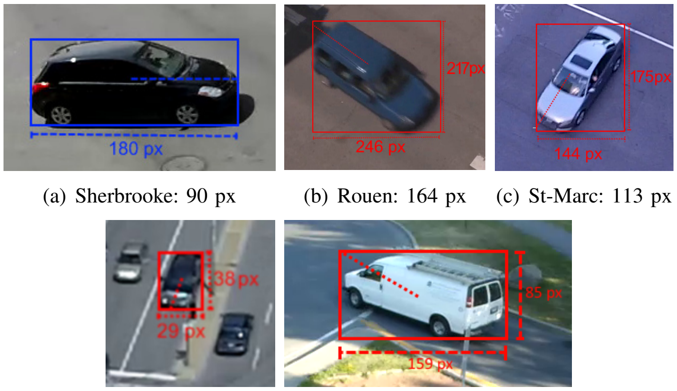
[1] Available at https://bitbucket.org/Nicolas/trafficintelligence (revision 43ae3a1 August 7th, 2015)

[2] The evaluation videos with ground truth and the tracker source code are available at http://www.jpjodoin.com/urbantracker/



(a) Sherbrooke (b) Rouen (c) St-Marc (d) Rene-L. (e) Roundabout

Fig. 5. Sample frame from each video



(a) Sherbrooke: 90 px (b) Rouen: 164 px (c) St-Marc: 113 px

(d) Rene-L.: 24 px (e) Roundabout: 80 px

Fig. 6. Association thresholds on the reference objects used for each video

object, called the association threshold, must be defined. It was chosen for each video as the half-diagonal of the bounding box of a typical element in the middle of the frame. The reference objects used and the association thresholds are shown in Fig. 6. The influence of the association threshold on the results is discussed in section IV-D.

For each test video, the parameters were optimized for UT and the authors of TI provided us with appropriate parameters for their algorithm. For UT, the parameters are the number of frames to look for a lost object (N_r), the fragmentation factor (D_b) and the minimum blob size (T_m). For TI, the parameters that were adjusted were the connection distance (d_{con}), the segmentation distance (d_{seg}) and the minimum distance between features (m_{df}). A homography was used when available for TI. The parameters used for each video are presented in Table II.

B. Experimental Results

The results are presented for each video as a whole and for each road user type separately in Table III. From this table, we can observe that UT's MOTA is higher than TI on each tested video and for each road user type (a MOTA around 0.8 and above has been associated empirically to god enoug performance for transportation applications [36]). This means that UT generates fewer id changes, false positives and false negatives overall. Looking at the variation of MOTA per road user type, it can be observed that it varies less for UT than TI. This can be explained by the use of background subtraction that allows UT to detect objects of various types and sizes without rigid motion constraints. The observed variation is caused by complex interactions between objects and the resulting occlusions, but not by the object type or size. For the Sherbrooke video, the difference between the MOTA for the cars and the pedestrians is caused by two pedestrians moving as a group for the whole scene, which does not permit the algorithm to segment them correctly. If both pedestrians are considered as a single group object in ground truth annotation, the MOTA for pedestrians increase from 0.705 to 0.895, which is comparable to the score for cars in this video. For the Rouen and St-Marc videos, the interactions between pedestrians are more complex than between cars, which leads to some id changes and misses. Based on the MOTP, UT is more precise in most cases. The cases of lower performance for UT can be attributed to deformations of the background subtraction due to shadows. This is especially true for Roundabout where the shadows are very large and some large objects near the camera are poorly segmented.

TABLE III
CLEAR MOT METRICS FOR THE VIDEOS: LARGER MOTA VALUES AND SMALLER MOTP VALUES REFLECT BETTER PERFORMANCE. ALL RESULTS ARE SHOWN SEPARATELY BY ROAD USER TYPE AND ALTOGETHER. BOLDFACE INDICATES THE BEST RESULTS BETWEEN UT AND TI

		UT		TI [9]	
Video	Type	MOTA	MOTP	MOTA	MOTP
Sherbrooke	Cars	**0.887**	10.59 px	0.825	**7.42 px**
	Ped.	**0.705**	**6.61 px**	0.014	11.98 px
	All	**0.787**	8.64 px	0.384	**7.54 px**
Rouen	Cars	**0.896**	**9.73 px**	0.185	66.69 px
	Ped.	**0.830**	**13.77 px**	0.647	20.04 px
	Cyc.	**0.927**	14.13 px	0.869	**13.11 px**
	All	**0.844**	**13.19 px**	0.588	24.20 px
St-Marc	Cars	**0.889**	**10.90 px**	-0.178	38.99 px
	Ped.	**0.730**	**5.05 px**	0.693	10.44 px
	Cyc.	**0.989**	**6.39 px**	0.895	7.46 px
	All	**0.764**	**5.99 px**	0.602	14.58 px
Rene-L.	Cars	**0.796**	**3.04 px**	0.547	5.23 px
	Cyc.	**0.232**	**2.20 px**	0.232	3.14 px
	All	**0.723**	**2.98 px**	0.503	5.10 px
Roundabout	All (Cars)	**0.718**	12.99 px	0.605	**8.57 px**

The MOTA of TI reveals that there is a road user type with a much lower score than the others for each video. The algorithm has difficulty handling objects of various sizes using one set of parameters: in order to maximize its MOTA, the parameters should be set for the most common object in the scene. For Sherbrooke, the parameters were tuned for cars because it is the type with the highest number of observations. As a result, almost no pedestrians were tracked. For Rouen and St-Marc, the parameters were tuned for pedestrians to maximize the score. Pedestrians and cyclists have similar sizes in those videos: MOTA is therefore better for them than for cars. In this case, cars are over-segmented and their detections as measured by MOTP have very low precision.

TABLE II
PARAMETERS USED FOR UT AND TI FOR EACH VIDEO (HOM. INDICATES THE USE OF A HOMOGRAPHY)

	UT		TI [9]	
Video	Parameters	Hom.	Parameters	Hom.
Sherbrooke	N_r = 1160, D_b = 0.1, T_m = 300	no	d_{con} = 4 m, d_{seg} = 1.7 m, m_{df}=5	yes
Rouen	N_r = 150, D_b = 0.7, T_m = 380	no	d_{con} = 25 px, d_{seg} = 25 px, m_{df}=5	no
St-Marc	N_r = 1160, D_b = 0.1, T_m = 300	no	d_{con} = 10 px, d_{seg} = 40 px, m_{df}=5	no
Rene-L.	N_r = 900, D_b = 0.2, T_m = 50	no	d_{con} = 10 px, d_{seg} = 40 px, m_{df}=2	no
Roundabout	N_r = 200, D_b = 0.2, T_m = 300	no	d_{con} = 3.75 m, d_{seg} = 1.5 m, m_{df}=5	yes

Returning to our method, UT, the MOTA for cyclists is very high for Rouen and St-Marc, and much lower for Rene-Levesque. In the first two videos, there are few interactions between cyclists and other road users. In the Rene-Levesque video, one of the two cyclists is far from the camera which makes it very small. It is too small to be detected by the background subtraction and it is filtered as noise. Since this cyclist stays for a very long time in the video, this causes a large number of false negatives. This explains the very low MOTA for cyclists in this scene. The MOTA of TI for cyclists is similarly small for the same reason in that scene. In the case of TI, the object is too small for the KLT tracker to find enough points.

From the Table IV, it can be seen that UT has overall higher MT and lower ML for all videos which is consistent with the comparison based on the CLEAR MOT metrics.

TABLE IV
MOSTLY TRACK (MT) AND MOSTLY LOST (ML) METRICS FOR THE VIDEOS: LARGER MT VALUES AND SMALLER ML VALUES REFLECT BETTER PERFORMANCE. BOLDFACE INDICATES THE BEST RESULTS BETWEEN UT AND TI

	UT		TI [9]	
Video	MT	ML	MT	ML
Sherbrooke	**0.850**	**0.050**	0.600	0.250
Rouen	**0.750**	**0.000**	0.375	0.375
St-Marc	**0.643**	**0.071**	0.607	0.179
Rene-L.	**0.759**	**0.138**	0.414	0.207
Roundabout	0.509	**0.000**	**0.636**	0.200

C. Parameters sensitivity

In order to evaluate the sensitivity of each algorithm to their parameters, and to avoid bias caused by our choice of parameters, a five-fold cross-validation has been applied where one fold is used for training and four folds are used for testing. To do that, the parameters found for each video sequence are used (see Table II) to evaluate each tracker on the four other test video sequences. The cross-validation results were then calculated by averaging MOTA and MOTP: UT has better MOTA (0.716 against 0.187) and MOTP (9.27 px against 17.19 px) than TI. The intermediate results obtained for each fold and used for calculating the final cross-validation results are presented in Table V.

The cross-validation results obtained by UT are close to the results obtained in Table III, which demonstrates that UT generalizes well and that it is not too sensitive to the choice of the parameters. On the other hand, the average MOTA for TI is low and far from the results obtained using parameters adjusted to each video sequence individually. Although it is has been shown that TI is less sensitive to parameters when using homographies [9], TI needs different grouping and segmentation parameters to handle correctly objects of different sizes: without classification and different sets of parameters for different road users, its accuracy is therefore significantly affected in scenes with objects of different sizes, leading to over-grouping and under-grouping of objects. These grouping problems will create a high number of false positives which affects negatively the MOTA and the precision of the position of the centroid.

D. CLEAR MOT association threshold

The choice of the association threshold of CLEAR MOT may have a large impact on the performance measures MOTA and MOTP and may bias the reported results: if too low, valid object tracks will be considered as false positives; if too high, false alarms may be associated to missed objects. In order to show that the choice of association threshold did not alter the comparison between UT and TI, the MOTA and MOTP were plotted as a function of the association threshold in Fig. 7 for the Sherbooke and Rene-Levesque videos (the plots for the others are similar). It shows that UT is superior to TI over the whole range of association threshold, except for the MOTA of Sherbrooke, where TI outperforms UT for small association thresholds. UT's MOTA is slightly lower in the Sherbrooke video when the association threshold is under 9 px, which is its average precision (see its MOTP "plateau"). We can also observe that the MOTA and MOTP curves get to a plateau much faster for UT. This is because UT generates very few false positives and the overall precision is higher.

E. Application to Traffic Data Collection

Finally, an example of the application of UT to traffic data collection is shown in Fig. 8. Four screen-lines are drawn and all objects crossing at least two lines are counted, with the first considered as the origin and the last as the destination in the intersection (U-turns are not counted). Average instantaneous speeds are also represented spatially in the intersection after filtering out unrealistic speeds above 70 km/h.

V. CONCLUSION

We have presented Urban Tracker (UT), a new tracking algorithm based on modern binary descriptor and background subtraction technique. Along with a FSM, the use of both methods allows to track road users of various types and sizes. It also includes a bounding box estimation method that handles partial occlusion. Experimental results show that

TABLE V

RESULTS OF 5-FOLD CROSS-VALIDATION. THE NUMBER ON THE LEFT OF THE ARROW IS THE RESULT AND THE NUMBER ON THE RIGHT REPRESENTS THE DIFFERENCE BETWEEN THE MOTA AND THE MOTP FOR THE OPTIMIZED AND THE SCENE SPECIFIC PARAMETERS. THE MOTA DIFFERENCE IS A RELATIVE DIFFERENCE IN PERCENTAGE POINTS. FOR THE MOTA COLUMN, A ⇓ IS A LOWER TRACKING ACCURACY WHILE FOR THE MOTP COLUMN, A ⇑ IS A LOWER TRACKING PRECISION. RESULTS WERE NOT REPORTED FOR TI WITH SHERBROOKE AND ROUNDABOUT PARAMETERS SINCE THEY TAKE INTO ACCOUNT A HOMOGRAPHY AND CANNOT BE APPLIED TO THE OTHER VIDEOS

Fold 1: Sherbrooke parameters				
	UT		TI [9]	
Video	MOTA	MOTP	MOTA	MOTP
Rouen	0.770 ⇓ **7.38%**	14.01 px ⇑ **0.82 px**	NA	NA
St-Marc	0.764 ≈ **0.00%**	5.99 px ≈ **0.00 px**	NA	NA
Rene-L.	0.717 ⇓ **0.58%**	2.80 px ⇓ **0.17 px**	NA	NA
Roundabout	0.507 ⇓ **21.14%**	14.07 px ⇑ **1.07 px**	NA	NA

Fold 2: Rouen parameters				
	UT		TI [9]	
Video	MOTA	MOTP	MOTA	MOTP
Sherbrooke	0.787 ≈ **0.00%**	8.68 px ⇑ **0.04 px**	0.379 ⇓ 0.47%	13.80 px ⇑ 6.26 px
St-Marc	0.701 ⇓ **6.28%**	7.03 px ⇑ **1.04 px**	0.422 ⇓ 18.0%	20.93 px ⇑ 6.34 px
Rene-L.	0.669 ⇓ **5.37%**	3.05 px ⇑ **0.07 px**	0.305 ⇓ 19.7%	5.77 px ⇑ 0.67 px
Roundabout	0.664 ⇓ **5.40%**	13.62 px ⇑ **0.62 px**	0.561 ⇓ 4.32%	10.44px ⇑ 1.88px

Fold 3: St-Marc parameters				
	UT		TI [9]	
Video	MOTA	MOTP	MOTA	MOTP
Sherbrooke	0.787 ≈ **0.00%**	8.64 px ≈ **0.00 px**	-0.860 ⇓ 124.41%	19.90 px ⇑ 12.37 px
Rouen	0.770 ⇓ **7.38%**	14.01 px ⇑ **0.82 px**	0.443 ⇓ 14.58%	30.64 px ⇑ 6.44 px
Rene-L.	0.717 ⇓ **0.58%**	2.80 px ⇓ **0.17 px**	0.206 ⇓ 29.73%	5.95 px ⇑ 0.84 px
Roundabout	0.507 ⇓ **21.14%**	14.07 px ⇑ **1.07 px**	-0.003 ⇓ 60.79%	17.16 px ⇑ 8.59 px

Fold 4: Rene-L. parameters				
	UT		TI [9]	
Video	MOTA	MOTP	MOTA	MOTP
Sherbrooke	0.750 ⇓ **3.66%**	8.90 px ⇑ **0.26 px**	-0.503 ⇓ 88.70%	17.29 px ⇑ 9.75 px
Rouen	0.765 ⇓ 7.92%	15.84 px ⇑ **2.64 px**	0.517 ⇓ **7.12%**	34.17 px ⇑ 9.97 px
St-Marc	0.679 ⇓ **8.55%**	7.73 px ⇑ **1.74 px**	0.335 ⇓ 26.66%	16.78 px ⇑ 2.19 px
Roundabout	0.633 ⇓ **8.56%**	12.26 px ⇓ **0.74 px**	0.167 ⇓ 43.73%	13.40 px ⇑ 4.83 px

Fold 5: Roundabout parameters				
	UT		TI [9]	
Video	MOTA	MOTP	MOTA	MOTP
Sherbrooke	0.791 ⇑ **0.43%**	8.65 px ⇑ **0.02 px**	NA	NA
Rouen	0.729 ⇓ **11.57%**	14.02 px ⇑ **0.83 px**	NA	NA
St-Marc	0.742 ⇓ **2.24%**	6.25 px ⇑ **0.26 px**	NA	NA
Rene-L.	0.619 ⇓ **10.34%**	2.89 px ⇓ **0.09 px**	NA	NA

UT performs better than a current state-of-the-art road user tracker on five real urban traffic scenes that contain several types of road users. The main contribution is a new tracker designed specifically for urban tracking that requires no prior knowledge (camera calibration) while needing few intuitive parameter adjustments. Indeed, cross-validation results show that the new tracker generalizes very well to different scenes with the same parameters. Results show balanced performance for the tracking of pedestrians, cyclists and motorized vehicles.

ACKNOWLEDGEMENT

The authors wish to thank Marilyne Brosseau, Jean-Simon Bourdeau, Jean-Michel Auberlet and Mohamed Gomaa Mohamed for collecting the video data.

REFERENCES

[1] A. Tarko, G. A. Davis, N. Saunier, T. Sayed, and S. Washington, “Surrogate measures of safety,” ANB20(3) Subcommittee on Surrogate Measures of Safety, White Paper, Apr. 2009.

[2] N. Saunier, T. Sayed, and K. Ismail, “Large scale automated analysis of vehicle interactions and collisions,” *Transportation Research Record*, vol. 2147, pp. 42–50, 2010.

[3] A. A. Butt and R. T. Collins, “Multi-target tracking by lagrangian relaxation to min-cost network flow,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2013.

[4] J. Liu, P. Carr, R. T. Collins, and Y. Liu, “Tracking sports players with context-conditioned motion models,” in *Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition*, ser. CVPR ’13. Washington, DC, USA: IEEE Computer Society, 2013, pp. 1830–1837.

[5] N. Goyette, P.-M. Jodoin, F. Porikli, J. Konrad, and P. Ishwar, “A novel video dataset for change detection benchmarking,” *Image Processing, IEEE Transactions on*, vol. 23, no. 11, pp. 4663–4679, Nov 2014.

[6] O. Barnich and M. Van Droogenbroeck, “ViBe: A universal background subtraction algorithm for video sequences,” *IEEE Transactions on Image Processing*, vol. 20, no. 6, pp. 1709–1724, June 2011.

[7] X. Zhao, Y. Satoh, H. Takauji, S. Kaneko, K. Iwata, and R. Ozaki, “Object detection based on a robust and accurate statistical multi-point-pair model,” *Pattern Recogn.*, vol. 44, no. 6, pp. 1296–1311, Jun. 2011.

[8] P. Peng, Y. Tian, Y. Wang, J. Li, and T. Huang, “Robust multiple cameras pedestrian detection with multi-view bayesian network,” *Pattern Recogn.*, vol. 48, no. 5, pp. 1760–1772, May 2015.

[9] N. Saunier and T. Sayed, “A feature-based tracking algorithm for vehicles in intersections,” in *The 3rd Canadian Conference on Computer and Robot Vision, 2006*, 2006, pp. 59–59.

[10] J.-P. Jodoin, G.-A. Bilodeau, and N. Saunier, “Urban tracker: Multiple object tracking in urban mixed traffic,” in *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2014.

[11] X. Yan, X. Wu, I. A. Kakadiaris, and S. K. Shah, “To track or to detect? an ensemble framework for optimal selection,” in *Computer Vision ECCV 2012*, no. 7576. Springer Berlin Heidelberg, Jan. 2012, pp. 594–607.

[12] C. Huang, B. Wu, and R. Nevatia, “Robust object tracking by hierarchical association of detection responses,” in *Proceedings of the 10th European Conference on Computer Vision*, ser. ECCV ’08. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 788–801.

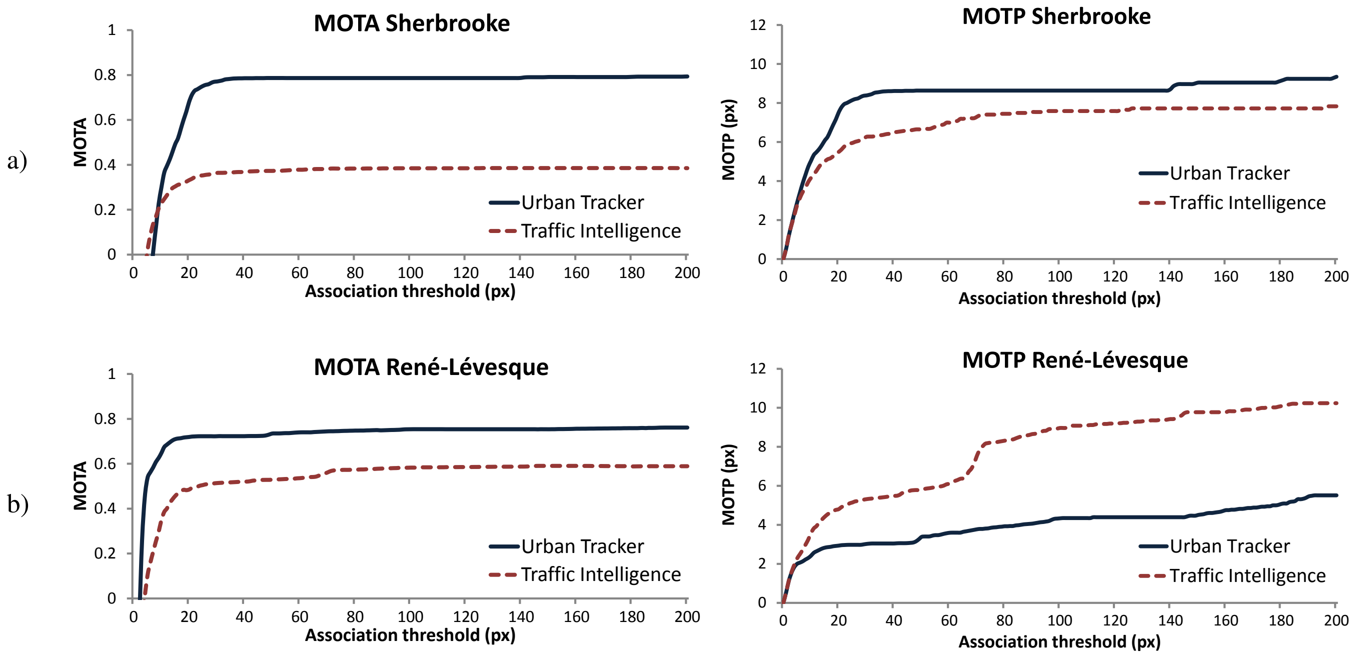
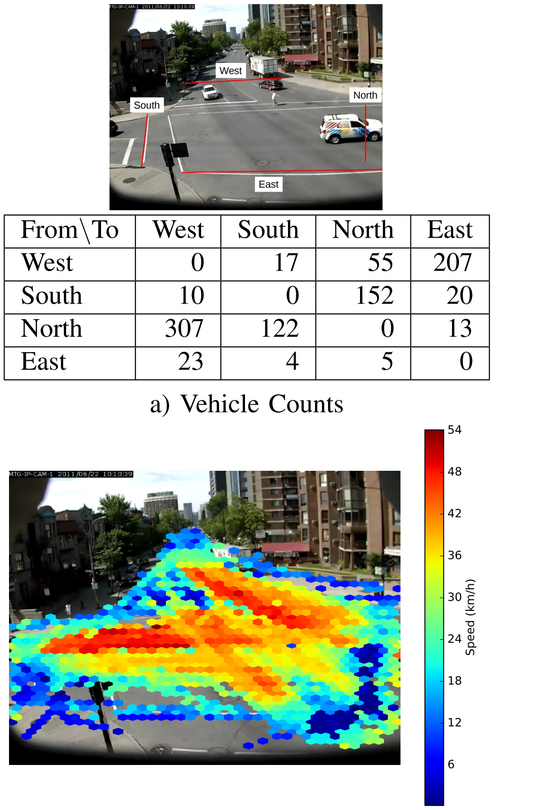


Fig. 7. Impact of the association threshold of CLEAR MOT on the MOTA and MOTP for each video and tracking algorithm. For the left column, a higher curve represents a better MOTA while for the right column, a lower curve represents a better MOTP



From\To	West	South	North	East
West	0	17	55	207
South	10	0	152	20
North	307	122	0	13
East	23	4	5	0

a) Vehicle Counts

b) Spatial distribution of average instantaneous speeds

Fig. 8. Sample traffic data for Sherbrooke (11AM to 12PM):

[13] B. Coifman, D. Beymer, P. McLauchlan, and J. Malik, “A real-time computer vision system for vehicle tracking and traffic surveillance,” *Transportation Research Part C: Emerging Technologies*, vol. 6, pp. 271–288, 1998.

[14] D. Beymer, P. McLauchlan, B. Coifman, and J. Malik, “A real-time computer vision system for measuring traffic parameters,” in *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*, 1997, pp. 495–501.

[15] J. C. Mendes, A. G. C. Bianchi, and A. R. P. Júnior, “Vehicle tracking and origin-destination counting system for urban environment.” in *Proceedings of the International Conference on Computer Vision Theory and Applications (VISAPP 2015)*, 2015.

[16] W. Wang, T. Gee, J. Price, and H. Qi, “Real time multi-vehicle tracking and counting at intersections from a fisheye camera,” in *Proceedings of the 2015 IEEE Winter Conference on Applications of Computer Vision*, ser. WACV ’15. Washington, DC, USA: IEEE Computer Society, 2015, pp. 17–24.

[17] A. Torabi and G. A. Bilodeau, “A multiple hypothesis tracking method with fragmentation handling,” in *Computer and Robot Vision, 2009. CRV ’09. Canadian Conference on*, 2009, pp. 8–15.

[18] F. Chang, C.-J. Chen, and C.-J. Lu, “A linear-time component-labeling algorithm using contour tracing technique,” *Computer Vision and Image Understanding*, vol. 93, no. 2, pp. 206–220, 2004.

[19] G. Jun, J. K. Aggarwal, and M. Gokmen, “Tracking and segmentation of highway vehicles in cluttered and crowded scenes,” in *Proceedings of the 2008 IEEE Workshop on Applications of Computer Vision*, ser. WACV ’08. Washington, DC, USA: IEEE Computer Society, 2008, p. 1–6.

[20] Z. Kim, “Real time object tracking based on dynamic feature grouping with background subtraction,” in *IEEE Conference on Computer Vision and Pattern Recognition, 2008. CVPR 2008*, 2008, pp. 1–8.

[21] X. Luo and S. Bhandarkar, “Tracking of multiple objects using optical flow based multiscale elastic matching,” in *Dynamical Vision*, ser. Lecture Notes in Computer Science, R. Vidal, A. Heyden, and Y. Ma, Eds. Springer Berlin Heidelberg, 2007, vol. 4358, pp. 203–217.

[22] S. Aslani and H. Mahdavi-Nasab, “Optical flow based moving object detection and tracking for traffic surveillance,” *International Journal of Electrical, Robotics, Electronics and Communications Engineering*, vol. 7, no. 9, pp. 773–777, 2013.

[23] J. Shin, S. Kim, S. Kang, S.-W. Lee, J. Paik, B. Abidi, and M. Abidi, “Optical flow-based real-time object tracking using non-prior training active feature model,” *Real-Time Imaging*, vol. 11, no. 3, pp. 204–218, Jun. 2005.

[24] J. Shi and C. Tomasi, “Good features to track,” in *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR ’94., 1994 IEEE Computer Society Conference on*, 1994, pp. 593–600.

[25] C. Stauffer and W. E. L. Grimson, “Learning patterns of activity

using real-time tracking," *IEEE Transactions on Pattern Analalysis and Machine Intelligence*, vol. 22, no. 8, pp. 747–757, Aug. 2000.

[26] L. M. Fuentes and S. A. Velastin, "People tracking in surveillance applications," *Image and Vision Computing*, vol. 24, no. 11, pp. 1165–1171, 2006.

[27] J. Batista, P. Peixoto, C. Fernandes, and M. Ribeiro, "A dual-stage robust vehicle detection and tracking for real-time traffic monitoring," in *Intelligent Transportation Systems Conference, 2006. ITSC '06. IEEE*, 2006, pp. 528–535.

[28] X. Song and R. Nevatia, "A model-based vehicle segmentation method for tracking," in *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, vol. 2, 2005, pp. 1124–1131 Vol. 2.

[29] I. O. Sebe, S. You, and U. Neumann, "Globally optimum multiple object tracking," in *Proc. SPIE 5810, Acquisition, Tracking, and Pointing XIX*, vol. 5810, 2005, pp. 82–93.

[30] R. Ortiz, "Freak: Fast retina keypoint," in *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, ser. CVPR '12. Washington, DC, USA: IEEE Computer Society, 2012, pp. 510–517.

[31] C. C. Liñán, "cvBlob," http://cvblob.googlecode.com. [Online]. Available: http://cvblob.googlecode.com

[32] S. Leutenegger, M. Chli, and R. Siegwart, "BRISK: binary robust invariant scalable keypoints," in *2011 IEEE International Conference on Computer Vision (ICCV)*, 2011, pp. 2548–2555.

[33] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vision*, vol. 60, no. 2, pp. 91–110, Nov. 2004.

[34] K. Bernardin and R. Stiefelhagen, "Evaluating multiple object tracking performance: the CLEAR MOT metrics," *J. Image Video Process.*, vol. 2008, p. 1:1–1:10, jan 2008.

[35] L. Leal-Taixé, A. Milan, I. Reid, S. Roth, and K. Schindler, "MOTChallenge 2015: Towards a benchmark for multi-target tracking," Tech. Rep., Apr. 2015, arXiv: 1504.01942.

[36] P. Morse, P. St-Aubin, L. F. Miranda-Moreno, and N. Saunier, "Transferability study of video tracking optimization for traffic data collection and analysis," in *Transportation Research Board Annual Meeting Compendium of Papers*, 2016.

Jean-Philippe Jodoin received the B.Sc.A. degree in computer engineering from Université de Sherbrooke in 2011 and the M.Sc.degree in computer engineering from Polytechnique Montréal in 2013. His research interests encompass object tracking, object detection and video surveillance. He is especially interested in the applications of computer vision to building smarter cities and safer transportation.

Guillaume-Alexandre Bilodeau (M'10) received the B.Sc.A. degree in computer engineering and the Ph.D. degree in electrical engineering from Université Laval, Canada, in 1997 and 2004, respectively. In 2004, he was appointed Assistant Professor at Polytechnique Montréal, Canada, where he is now Full professor since 2014. His research interests encompass image and video processing, video surveillance, object tracking, segmentation, and medical applications of computer vision. Dr. Bilodeau is a member of the REPARTI research network.

Nicolas Saunier received an engineer degree in telecommunication and a Doctorate (Ph.D.) in computer engineering from Telecom ParisTech in 2001 and 2005 respectively. In 2009, he was appointed Assistant Professor at Polytechnique Montréal, Canada, where he is now an Associate Professor since 2014. His interests include intelligent transportation systems (ITS), road safety, and information technology for transportation (data collection, storage, processing, using machine learning techniques, and visualization). He is a member of the CIRRELT interuniversity research centre and co-heads its ITS lab.