Motion-constrained Road User Tracking for Real-time Traffic Analysis

Nyan Bo Bo[1,2][a], Peter Veelaert[1,2][b] and Wilfried Philips[1,2]
[1]*TELIN-IPI, Ghent University, Sint-Pietersnieuwstraat 41, B-9000 Gent, Belgium*
[2]*imec, Kapeldreef 75, B-3001 Leuven, Belgium*

Keywords: Real-time Tracking, Automatic Traffic Analysis, Edge Processing, Privacy Preservation, Turning Movement Count.

Abstract: Reliability of numerous smart traffic applications are highly dependent on the accuracy of underlying road user tracker. Demand on scalability and privacy preservation pushes vision-based smart traffic applications to sense and process images on edge devices and transmit only concise information to decision/fusion nodes. One of the requirements for deploying a vision algorithm on edge devices is its ability to process captured images in real time. To meet these needs, we propose a real-time road user tracker which outperforms state-of-the-art trackers. Our approach utilizes double thresholding on detector responses to suppress initialization of false positive trajectories while assuring corresponding detector responses required for updating trajectories are not wrongly discarded. Furthermore, our proposed Bayes filter reduces fragmentation and merging of trajectories which highly effect the performance of subsequent smart traffic applications. The performance of our tracker is evaluated on the real life traffic data in turning movement counting (TMC) application and it achieves a high precision of 96% and recall of 95% while state-of-the-art tracker in comparison achieves 92% on precision and 87% on recall.

1 INTRODUCTION

Many computer vision-based smart traffic applications such as automatic turning movement counting (TMC), speed estimation, unusual trajectory detection, *etc.* require tracking of multiple road users simultaneously. The reliability of these applications is highly dependent on the performance, *i.e.*, accuracy, precision and speed, of the underlying visual tracker.

Beside the performance, the privacy preservation of road users also plays an important role in deploying camera-based applications in public spaces. Fortunately, the justified fears of camera invading privacy can be reduced by technological means. The major worry is that the video captured by the cameras can be abused. This problem can be avoided by processing the video inside the camera. Video that is never sent from the camera cannot be abused.

In technical terms, this requires smart cameras(Rinner and Wolf, 2008): cameras with on-board processing and communication capabilities. However, smart cameras usually possess lower computational power than desktop computers. Therefore, the computational complexity of a computer vision algorithm must be kept low for real-time deployment on a smart camera. Due to the limited field of view of cameras, multiple smart cameras are required to cover a wide area. Unlike centralized systems, this scale up can be achieved without computational and communication bottleneck since the computation load is distributed among many smart cameras and only concise information is transmitted rather than video streams.

To these ends, we propose a real-time road user tracker which is suitable to deploy on edge devices for privacy conservation. Unlike a conventional tracking-by-detection scheme in which detector responses are thresholded with a single threshold, we propose a double thresholding approach which is the first contribution of this paper. A higher threshold is used for an initialization of new tracks without producing a high number of false positives while a lower one is used to decide if a detector response is reliable enough to update the trajectory. The second contribution is the Bayes filter cascade with a constrained matching which significantly reduces mismatching of trajectories to detector responses. As a third contribution, we evaluated the performance of our tracker on video stream of real-life traffic in the city of Antwerp, Belgium.

[a] https://orcid.org/0000-0002-6904-4672
[b] https://orcid.org/0000-0003-4746-9087

Bo, N., Veelaert, P. and Philips, W.
Motion-constrained Road User Tracking for Real-time Traffic Analysis.
DOI: 10.5220/0010788700003124
In *Proceedings of the 17th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2022) - Volume 5: VISAPP*, pages 785-792
ISBN: 978-989-758-555-5; ISSN: 2184-4321

To investigate the exact accuracy of the statistics, we conducted a one time experiment in which we compared our automatic traffic statistics in a street to the correct value (determined by a human observer). We achieved a precision of 96% and recall of 95%, outperforming the state-of-the-art method in literature whose precision and recall is 92% and 87% respectively.

2 RELATED WORK

The work of (Bochinski et al., 2017) experimentally shows that their simple tracker based on the intersection-over-union (IOU) of detector responses at sufficiently high frame rates outperforms the state-of-the-art tracker at only a fraction of the computational cost. However, their method assumes that the detector produces a detection per frame for every object being tracked allowing only few missed detections. This assumption is often invalid when an object is occluded for a few frames. Although the computational cost of their tracker is very low, its requirement for high frame rate videos to ensure a large overlap between detections in consecutive frames poses a high computational load on CNN-based object detection.

The shortcomings of the tracker of (Bochinski et al., 2017) are addressed in the Simple Online and Real-time Tracking (SORT) of (Bewley et al., 2016) while keeping a low computational cost. The SORT tracker deploys Kalman filtering not only to filter noise in trajectories but also to handle missing detections. Similar to the work of (Bochinski et al., 2017), the assignment of detections to existing trajectories are based on the intersection-over-union (IOU) distance between each detection and all the predicted bounding boxes of the Kalman filter. If no matched detection is found, *i.e.*, when the detector failed to detect the object because it was occluded or corrupted by image noise, the Kalman filter prediction becomes the estimated state of the object. When there is a matched detection, the estimated state is corrected by incorporating information from the matched detection. The work of (Tran et al., 2021) utilizes SORT tracker in their turning movement counting system which is designed to be deployed on edge devices.

Since the detection-to-trajectory assignment of the SORT tracker is solely based on the motion model of the Kalman filter and the IOU distance, the SORT tracker experiences more identity switches between tracked objects than the state-of-the-art trackers although it outperforms in terms of Multiple Object Tracking Accuracy (MOTA). To tackle the identity switching problem of SORT tracker, (Wojke et al., 2017) extend the detection-to-trajectory assignment method of SORT by integrating appearance information. They experimentally show that their extended method, *i.e.* extending SORT tracker with a deep association metric (DeepSORT), reduces the number of identify switches by 45% while maintaining overall competitive performance at high frame rates. However, identity switching between road users with similar appearance still occur when they are close by.

Some CNN-based trackers (Xu and Niu, 2021; Gloudemans and Work, 2021) perform detection and association across frames jointly by utilizing feedback information from object tracking. Since the previous object location and appearance information from the tracker is used as region proposal/prior in detection and association to narrow down the search space, this approach is faster than detect–associate–track approach. (Gloudemans and Work, 2021) follow this approach to generate trajectories for TMC application. Since object detection is never performed on a full frame, they claim that their method is approximately 50% faster than state-of-the-art methods in comparison. However, evaluation result indicates that their accuracy is lower than the DeepSort-based method (Lu et al., 2021).

The aforementioned trackers assume a very general tracking scenario where the cameras are not calibrated. However, trajectories on the ground plane are often required in smart traffic applications for trajectory clustering, abnormal behavior detection, analyzing the interaction between road users and so on. The projection of the road user's position from an image coordinates to the ground coordinates (GPS coordinates) can be found by determining the transformation (i.e., a homography) between the image plane and the ground plane. Since an image position can be mapped onto a ground position, Bayesian state estimation can be applied to the ground plane instead of the image plane.

Furthermore, a road user moving with constant velocity can result in non-constant velocity movement in the image plane. In addition to this, acceleration/deceleration of the road user can cause even more complex movement on the image plane. Therefore, our earlier work (Nyan et al., 2020) utilizes image to ground plane projection and tracks road users on the ground plane using Bayesian state estimation. However in this earlier work, only size and position difference between the prediction of the Bayesian filter and the detector responses are considered in the cost function formulation for track–detection association. Incorporating appearance information in cost function as in the work of (Wojke et al., 2017) could result in performance improvement.

3 THE PROPOSED METHOD

Our real-time traffic analysis system is designed to run entirely on a smart camera, *i.e.*, a hardware platform with an onboard camera, processors (CPU+GPU) and a communication module. The video frame grabber reads an image from the image sensor and feeds it to the YOLO object detector (Redmon et al., 2016) where locations of road users are identified. Our proposed constrained Bayes filter cascade tracks the YOLO detector responses to generate trajectories as input for the subsequent trajectory analysis block. Statistics of road users such as counts, speed, acceleration/deceleration, etc. can be extracted and transmitted to higher level smart traffic applications which may fuse this information with data streams from other data source/modality for joint traffic analysis/prediction.

3.1 Constrained Bayes Filter Cascade

The constrained Bayes filter block consists of three modules as depicted in Figure 1. Detector responses from the YOLO detector [1] are matched with existing trajectories in the matching cascade module taking into account the feedback constraints from both image and ground plane Bayes filter modules. Matched detector responses are then used to update the corresponding trajectories while unmatched ones are initialized as new trajectories. The detailed description on cascade module will be given later in Subsection 3.2 after some prerequisites have been discussed.

In detector-based tracking methods (Bochinski et al., 2017; Bewley et al., 2016; Wojke et al., 2017; Nyan et al., 2020), a threshold T is usually applied to the score of detector responses to suppress false positive detections. If T is set low, more false positive responses will be forwarded to the matching cascade. Since the matching cascade will not find any matching trajectory for these false positive responses, they are initialized as new trajectories, resulting in false positive trajectories. On the other hand, when setting T high to reduce the false positive rate, some true positives are sometimes rejected causing missed detections of a true road user. Multiple missed detections of the same road user often lead to untimely termination of the trajectory, *i.e.*, incomplete trajectory. If the road user is then redetected, a new trajectory is initialized creating multiple trajectories of a single road user, *i.e.*, fragmented trajectories.

Since no detector is perfect, even if T is optimally set, the problem of false positive, incomplete or segmented trajectories still exists. This issue can be mitigated by using two thresholds: a higher threshold for initialization of new trajectories and a lower threshold for updating the existing trajectories. First, a threshold T_l is applied to the score of all detector responses. Only responses with a score higher than T_l are considered further in the matching cascade. Then a second threshold $T_h > T_l$ is applied to the unmatched responses produced by the matching cascade module. An unmatched responses is initialized as a new trajectory only if its detection score is higher than T_h.

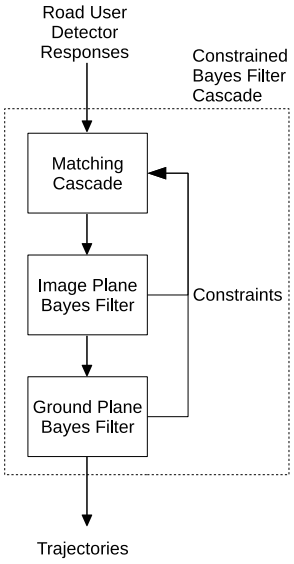

Figure 1: Modules of the proposed Bayes filter cascade.

To keep track of position, appearance and size of the road user in the image, the image plane Bayes filter is deployed. It estimates the eight dimensional state space of a road user denoted as $\mathbf{r} = [u, v, \gamma, h, \dot{u}, \dot{v}, \dot{\gamma}, \dot{h}]^T$ based on the corresponding detector responses $\mathbf{d} = [u, v, \gamma, h]^T$ selected by the matching cascade module. The center position of a road user either being tracked or detected is defined by u, v while aspect ratio γ and height h determine its size. The rate of change of position and size of the road user is defined by $\dot{u}, \dot{v}$ and $\dot{\gamma}, \dot{h}$ respectively. Similar to the related state-of-the-art trackers (Bochinski et al., 2017; Bewley et al., 2016; Wojke et al., 2017; Nyan et al., 2020), a standard Kalman implementation of a Bayes filter with a constant velocity motion model and a linear observation model is employed for its low computational complexity.

In addition to the image plane filter, the ground plane Bayes filter not only suppresses image plane–ground plane projection noise but also naturally models the motion of road users on the ground plane. It estimates the four dimensional state space of a road

[1] Only a subset of YOLO output classes, i.e., car, truck, bus and train (tram) are used in this work.

user denoted as $\acute{\mathbf{r}} = [x, y, \dot{x}, \dot{y}]^T$ where x, y represent the ground plane position while $\dot{x}$ and $\dot{y}$ denote velocity components. Given a homography matrix $H_{3\times3}$, an image plane position of a road user can be projected onto the ground plane:

$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = H_{3\times3} \begin{bmatrix} u \\ v + \frac{h}{2} \\ 1 \end{bmatrix} \tag{1}$$

The center position $[u, v]^T$ of a matched detector responses $\mathbf{d}$ is projected onto the ground plane as $\acute{\mathbf{d}} = [x, y]^T$ to update the ground plane state of the corresponding road user.

For each trajectory, we keep track of the number of consecutive frames in which the trajectory did not have any matched detection, *i.e.*, no observation. This count is denoted as c_i and it increases for each frame that does not have a matched detection for the trajectory. When c_i exceeds a threshold T_c, *i.e.*, when there is no observation for T_c consecutive frames, the corresponding road user is considered to have left the camera's field of view or its appearance in the camera view has become smaller than what the object detector is able to detect. In this situation, the trajectory is terminated. If the corresponding matched detector response is found again before c exceeds T_c, it is reset to zero.

3.2 Matching Cascade

The purpose of the matching cascade module is to correctly match the object detector responses to their corresponding trajectories so that both the image and ground plane Bayes filters can update their states. The matching is usually done by computing a cost matrix, that contains the matching costs between the detector responses and the predictions of the Bayes filter. A combinatorial optimization algorithm such as the Hungarian algorithm (Kuhn and Yaw, 1955) is then applied to find the matched pairs with minimum cost.

In this work, we adopt the cost matrix of (Wojke et al., 2017) which integrates motion and appearance information of the objects being tracked. To incorporate the motion information, the Mahalanobis distance between a predicted position of a trajectory i and the j-th detected position $\mathbf{d}_j$ is computed as:

$$\Delta_m(i, j) = (\mathbf{d}_j - \ddot{\mathbf{r}}_i)^T \ddot{\Sigma}_i^{-1} (\mathbf{d}_j - \ddot{\mathbf{r}}_i), \tag{2}$$

where $\ddot{\mathbf{r}}_i = \mathbf{I}\hat{\mathbf{r}}_i$ and $\ddot{\Sigma}_i = \mathbf{I}\hat{\Sigma}_i\mathbf{I}^T$ are the projection of predicted state $\hat{\mathbf{r}}_i$ and the corresponding covariance matrix $\hat{\Sigma}_i$ of the trajectory i. The matrix $\mathbf{I}$ is a 4×8 matrix:

$$\mathbf{I} = \begin{bmatrix} 1,0,0,0,0,0,0,0 \\ 0,1,0,0,0,0,0,0 \\ 0,0,1,0,0,0,0,0 \\ 0,0,0,1,0,0,0,0 \end{bmatrix}. \tag{3}$$

Furthermore, an appearance descriptor $\mathbf{a}_j$ with $||\mathbf{a}_j|| = 1$ is computed for each bounding box detection $\mathbf{d}_j$. A pre-trained residual network proposed by (Wojke and Bewley, 2018) on large-scale re-identification dataset by (Kanaci et al., 2018) is used to compute $\mathbf{a}_j$. For each trajectory i, a gallery $\mathcal{A}_i = \{\mathbf{a}_{i,k}\}_{k=1}^{K}$ of K latest appearance descriptors is constructed. When a new trajectory is initialized, there is a single appearance descriptor in $\mathcal{A}_i$. A new appearance descriptor is added to $\mathcal{A}_i$ only if there is a matched detector response. The oldest appearance descriptor in $\mathcal{A}_i$ is removed if number of appearance description in it exceeds K. Given $\mathcal{A}_i$, the appearance dissimilarity between the trajectory i and the detection j can be computed as:

$$\Delta_a(i, j) = \min\{1 - \mathbf{a}_j^T \mathbf{a}_\mathbf{i} | \mathbf{a}_\mathbf{i} \in \mathcal{A}_i\}. \tag{4}$$

Since $\mathbf{a}_j^T \mathbf{a}_\mathbf{i}$ equals the cosine of the angle between $\mathbf{a}_j$ and $\mathbf{a}_i$, $\Delta_a(i, j)$ is small when an appearance descriptor $\mathbf{a}_i$ along trajectory i is very similar to the detector response j.

Both the Mahalanobis distance Δ_m and appearance dissimilarity Δ_a are complementary as they are covering different aspects of the trajectory-to-detection matching task. The Mahalanobis distance Δ_m measures the location proximity as well as the similarity in size of a detection to a trajectory based on motion, and is particularly useful for short-term occlusion. However, motion becomes less reliable when the object is occluded for a longer period of time or when a detector fails due to noise. If this is the case, identity switching often occurs. The appearance distance Δ_a is particularly useful to mitigate this switching problem. Both metrics are combined using a weighted sum to compute the cost matrix for the assignment problem:

$$\Delta(i, j) = \lambda\Delta_m(i, j) + (1 - \lambda)\Delta_a(i, j), \tag{5}$$

where the weight λ can be experimentally tuned to achieve the optimal performance.

Furthermore, additional constraints can be incorporated in the cost computation to exclude unlikely associations. When the appearance dissimilarity is very high, it is unlikely to be a true match. Therefore, we apply a threshold T_a to define a binary variable $g_a(i, j)$ as follows:

$$g_a(i, j) = \mathbb{1}[\Delta_a(i, j) < T_a]. \tag{6}$$

As long as $\Delta_a(i, j)$ is smaller than T_a, $g_a(i, j) = 1$ and $g_a(i, j) = 0$ otherwise. This constitutes the constraint feedback from the image plane Bayes filter.

For a road user in motion, given its past known ground plane position, *i.e.*, $[x,y]^T$, and velocity at $t-1$, the prediction of its position at the current time t is usually close to its true position. Therefore, the corresponding detector response should not be far from the predicted position. Based on this assumption, we formulate a constraint (or gating function) on the angle and magnitude between the current, the predicted and the detected positions. A threshold T_θ is applied to the angle θ which is depicted in Figure 2 to create another binary variable g_θ:

$$g_\theta(i,j) = \mathbb{1}[\theta(i,j) < T_\theta]. \tag{7}$$

In addition to the constraints on the predicted and detected direction, the distance between the predicted and detected position is thresholded with T_d to further constrain the matching problem. Here, a binary variable on magnitude g_d is defined as:

$$g_d(i,j) = \mathbb{1}[\delta(i,j) < T_d], \tag{8}$$

where

$$\delta(i,j) = || \begin{bmatrix} 1,0,0,0 \\ 0,1,0,0 \end{bmatrix} \hat{\mathbf{r}}_i - \acute{\mathbf{d}}_j ||$$ (9)

as shown in Figure 2.

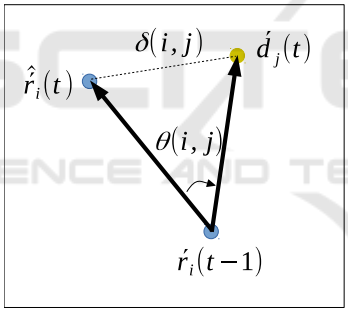

Figure 2: An illustration of $\delta(i,j)$ and $\theta(i,j)$ between a predicted state the ground plane Bayes filter and a detector response.

A prediction-detection pair is qualified to be included in the matching cascade only if all defined binary variables g_a, g_θ and g_d are equal to one. Therefore, we define the aggregation of these three binary variables g as:

$$g(i,j) = g_a(i,j)\ g_\theta(i,j)\ g_d(i,j). \tag{10}$$

Together, these variables act as gating functions on the position and appearance. The gated cost matrix Δ_g can then be computed using Equation 5 and 10 as follows:

$$\Delta_g = \begin{bmatrix} \Delta_g(1,1) & \Delta_g(1,2) & \dots & \Delta_g(1,J) \\ \Delta_g(2,1) & \Delta_g(2,2) & \dots & \Delta_g(2,J) \\ \vdots & \vdots & \ddots & \vdots \\ \Delta_g(I,1) & \Delta_g(I,2) & \dots & \Delta_g(I,J) \end{bmatrix} \tag{11}$$

where

$$\Delta_g(i,j) = \begin{cases} \Delta(i,j) & \text{If } g(i,j) \text{ is 1} \\ \kappa & \text{otherwise} \end{cases}. \tag{12}$$

The constant κ is set to be a very large number, *i.e.*, a very large cost.

Given the gated cost matrix Δ_g between the I trajectories and the J detector responses as well as the frame counts $C = \{c_1, c_2, \dots, c_I\}$ and $c_i \in \mathbb{Z}$, since the last valid update with a matched response, the cascade matching algorithm adopted from (Wojke et al., 2017) and listed in Algorithm 1 is applied. The matrix Δ_g is computed using Equation 11 and 12. This matching cascade does not treat all trajectories equally. It first considers only trajectories for which $c_i = 0$, *i.e.*, trajectories for which there is a matched detection in the previous frame. Hungarian minimum cost matching (Kuhn and Yaw, 1955) is applied to these trajectories and all the detector responses that are still available. The matched trajectories and detections are removed from the matching pool and the remaining candidates are considered in the next iteration. At the next iteration, all trajectories with $c_i = 1$, *i.e.*, that did not have a match in the previous frame and thus have higher uncertainty, are considered in Hungarian matching. The algorithm iterates until the maximum value in C is reached.

Algorithm 1: Recursive matching cascade algorithm.

procedure MINCOSTMATCHING(Δ_g, C)
  Matches: $M \leftarrow \phi$
  Unmatched detection: $U \leftarrow \{1,2,\dots,J\}$
  for $c \in [0,1,\dots,max(C)]$ **do**
    $L \leftarrow \{i | i \in \{1,2,\dots,I\} \text{ and } c_i = c\}$
    $\acute{M} \leftarrow Hungarian(\Delta, L)$
    $M \leftarrow M \cup \acute{M}$
    $U \leftarrow U \setminus \{j | (*,j) \in M\}$
  return M, U

The matching cascade algorithm returns matched pairs and unmatched detector responses. The matched detector responses are used to update the state of their corresponding trajectories in the Bayes filter while the unmatched detector responses with score more than T_h are initialized as new trajectories.

4 EXPERIMENTS

To assess the performance of our tracker in smart traffic application, we evaluate its performance in the aspect of Turning Movement Count (TMC) application. TMC at an intersection provides counts for road users leaving a street x (source) and entering a street y

(destination) during a predefined time period. This provides essential information on how traffic from one street is flowing into other streets at the intersection. Manually obtaining TMC is highly labor intensive since counts for N^2, where N is the number of streets connecting to the intersection, source-destination pairs have to be performed simultaneously. Fortunately, trajectories produced by visual tracker can be used to automatically obtain TMC. First, three lines at the entrance to each street at the interest are defined as shown in Figure 3. Then the TMC is simply obtained by increasing the corresponding count of $x \rightarrow y$ when a trajectory intersects with line x first and intersects later with line y.

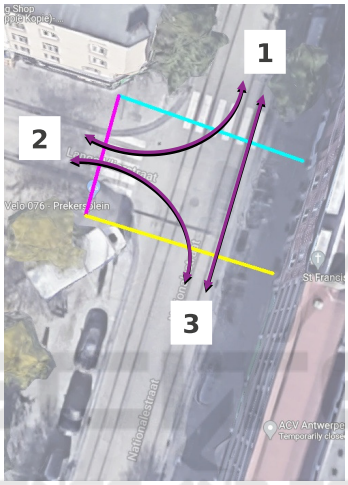

Figure 3: Lines defined for automatic turning movement counting. Arrows show the possible vehicle movements excluding u-turns. *Map Data: ©2021 Google, ©2021 Aerodata International Surveys.*

4.1 Dataset and Performance Metrics

For the quantitative evaluation of our tracker on turning movement counting application, four hours long video was captured between 7AM and 11PM from *Camera 11.1* of Antwerp's smart zone (asz,). The video is captured at 24 fps with the resolution of 960×720 pixels using the *AXIS Q6000-E MKII* camera located at the GPS coordinates 51°12′50.1″N 4° 23′ 53.5″E. The camera's field of view covers the intersection of a busy road *Nationalestraat* and a smaller street *Lange Vlierstraat*.

For the quantitative performance assessment, we manually annotated source and destination street of each vehicle in the video. Due to labor intensive nature of the manual annotation for TMC, only video segments between 7:00AM and 7:30AM, and 10:30AM and 11:00AM are annotated, resulting in annotations for 377 motorized road users. For qualitative evaluation on longer video, we captured a video from *Camera 11.1* for about three days covering both weekend and weekdays, *i.e.*, 14th to 17th of March 2021: from Sunday morning to Wednesday morning. Furthermore, five GPS coordinates as well as corresponding image coordinates of road markings visible in the camera's filed of view is obtained. Using these five GPS-image pairs, the image plane to ground plane homography matrix $H_{3\times3}$ is computed by using least-squares method.

Two performance metrics, precision and recall, are used for the quantitative performance assessment. Precision measures the ratio of correctly counted trajectories to the total count produced by the automatic TMC: which may also contains wrongly counted trajectories. It can be computed as:

$$precision = \frac{TP}{TP+FP}, \tag{13}$$

where TP is a number of true positive count and FP is a number of false positive count. Stray trajectories caused by false positive detector responses and identity switches (a trajectory formed by two or more road users) are two sources of false positive count. Recall, also known as sensitivity, measures the ratio of correctly counted trajectories to the total number of trajectories in the ground truth N_{GT}:

$$recall = \frac{TP}{N_{GT}}. \tag{14}$$

4.2 Quantitative Evaluation

In order to not only assess the performance of our tracker but also to compare with the performance of SOTA, the TMC is calculated from trajectories produced by our method as well as the SOTA tracker, DeepSort (Wojke et al., 2017). The resulting TMCs for all source-destination pairs together with ground truth counts for both trackers are given in Table 1. Both ground truth and automatic counts show that there is almost no traffic coming out from *2* which agrees with the fact that *2* (*Lange Vlierstraat*) is a one-way street allowing only incoming traffic from *Nationalestraat*. The ground truth indicates only a single case of motorized road user coming out of *2* and turning into *3* which is in fact a truck reversing out of the *2* after making a possibly wrong turn.

Table 1 shows that most road users move between *1* and *3*: both are straight through movements. U-turns at the intersection are also identified and counted by both automatic methods but DeepSort is overestimating u-turn counts by a large margin. These false positive counts of DeepSort are caused by identity switching between road users as well as merg-

Table 1: Automatic TMC counts at intersection of 3 road segments together with ground truth counts (automatic TMC/ground truth).

		Destination		
		1	**2**	**3**
Origin	**1**	1/1	9/10	119/123
	2	1/0	1/0	1/1
	3	207/209	30/30	6/3

(a) Our tracker

		Destination		
		1	**2**	**3**
Origin	**1**	8/1	9/10	110/123
	2	0/0	0/0	1/1
	3	194/209	27/30	6/3

(b) DeepSort

ing of multiple trajectories. Fortunately, the constrained Bayesian tracking of our tracker is robust against these problems, resulting in TMCs closer to the ground truth, and achieving a precision of 96% and a recall of 95%. Since the precision and recall of DeepSort is 92% and 87% respectively, our tracker outperforms DeepSort in both performance metrics. The quantitative evaluation results are summarized in Table 2.

Table 2: Detailed quantitative evaluation results.

	TP	**FP**	**Precision**	**Recall**
DeepSort	327	28	92%	87%
Ours	359	15	96%	95%

More detailed analysis of the resulting trajectories sheds some light on the outperformance of our tracker over the tracker in comparison. High number of fragmentations as well as merging of trajectories are found in trajectories produced by the DeepSort tracker. When a trajectory of a particular road user is fragmented into multiple segments, the automatic TMC fails to identify the source or destination of the road user. Moreover, TMC often incorrectly identifies the source and destination of a trajectory which is the result of merging of trajectories of multiple road users. These issues not only decrease the true positive TMCs but also increase the false positive TMCs. Our proposed double thresholding and constrained Bayes filter cascade are more robust against these issues. Thus, there is a significant reduction in trajectory fragmentation and merging.

4.3 Qualitative Evaluation

Furthermore, automatic TMC is applied to trajectories produced by our tracker on three days long video to observe the turning movement behavior over a longer period of time. To observe the time varying TMC along the course each day, TMC is computed for every hour. Graphs in Figure 4 show how two straight trough traffics ($1 \rightarrow 3$ and $3 \rightarrow 1$) vary between 7AM and 7PM for each day. These percentages of TMC for a specific source–destination pair is based on the total of TMC for all source–destination pair for the given period of time. It shows that traffic going from *3* to *1* (inbound to Antwerp city center) is usually denser than in the opposite direction, *i.e.*, traffic from *1* to *3*, which is in line with the observation from Table 1.

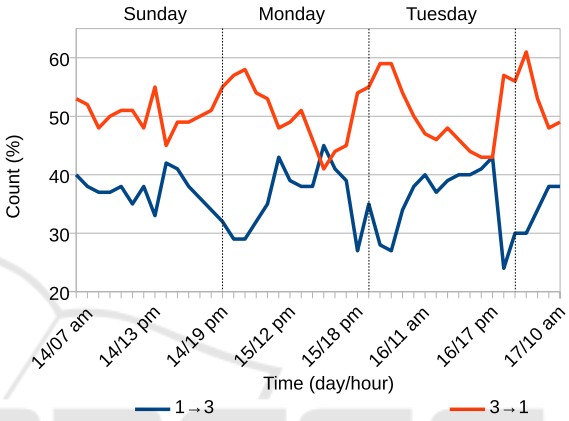

Figure 4: Percentage TMC counts by our method in relation to total movement counts for $1 \rightarrow 3$ and $3 \rightarrow 1$ movements.

Moreover, graphs in Figure 4 shows that $3 \rightarrow 1$ traffic is the highest ($\sim 60\%$) in the morning and decreases while $1 \rightarrow 3$ traffic from its lowest ($\sim 30\%$) increases over weekdays (Monday and Tuesday) until late afternoon, around 4PM, where traffic in both directions is almost the same (40%). Then, $3 \rightarrow 1$ traffic increases and $1 \rightarrow 3$ traffic decreases back. For Sunday, $3 \rightarrow 1$ traffic is fluctuating around 50% while $1 \rightarrow 3$ traffic is varying around 35%.

In addition to hourly TMCs, we also computed TMCs for each day. The traffic distribution at the intersection is then computed from daily TMCs and plotted as pie charts in Figure 5. All three pie charts show that $3 \rightarrow 1$ movement constitutes approximately 50% of the traffic passing trough the intersection. Furthermore, the $1 \rightarrow 3$ movement makes up approximately 35% of the total traffic flow at the intersection. This indicates that on each day, there is more motorized traffic towards the city center than outbound direction passing through the intersection. Moreover, it shows that traffic turning into *2* from *3* is approximately two times more frequent than traffic turning into *2* from *1*: $\sim 8\%$ and $\sim 4\%$ respectively. The remaining traffic which is about 3% consists of motorized road users taking u-turns at the intersection.

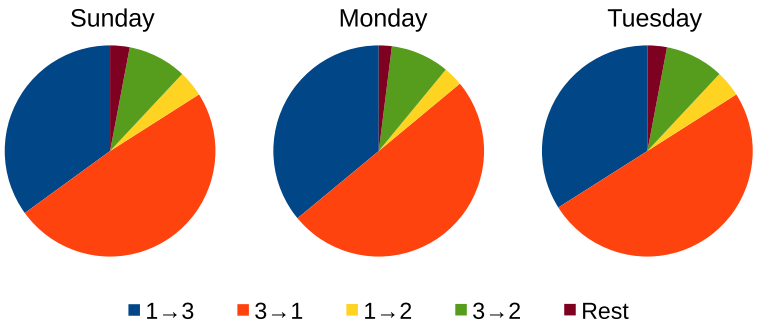

Figure 5: Daily traffic distribution computed from TMC at the intersection.

5 CONCLUSION

In this paper, we proposed a real-time road user tracker which is robust against fragmentation and merging of trajectories. This robustness is mostly contributed by double thresholding on object detector responses and the constrained matching of the Bayes filter cascade. Moreover, quantitative performance comparison to the SOTA method was also conducted and outperformance of our method over state-of-the-art tracker was validated.

ACKNOWLEDGEMENT

This work was funded by EU Horizon 2020 ECSEL JU research and innovation programme under grant agreement 876487 (NextPerception).

REFERENCES

Antwerp smart zone. https://antwerpsmartzone.be/. Accessed: 2021-09-08.

Bewley, A., Ge, Z., Ott, L., Ramos, F., and Upcroft, B. (2016). Simple online and realtime tracking. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 3464–3468.

Bochinski, E., Eiselein, V., and Sikora, T. (2017). High-speed tracking-by-detection without using image information. In *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6.

Gloudemans, D. and Work, D. B. (2021). Fast vehicle turning-movement counting using localization-based tracking. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 4150–4159.

Kanaci, A., Zhu, X., and Gong, S. (2018). Vehicle re-identification in context. In *Pattern Recognition - 40th German Conference, GCPR 2018, Stuttgart, Germany, September 10-12, 2018, Proceedings*.

Kuhn, H. W. and Yaw, B. (1955). The hungarian method for the assignment problem. *Naval Res. Logist. Quart*, pages 83–97.

Lu, J., Xia, M., Gao, X., Yang, X., Tao, T., Meng, H., Zhang, W., Tan, X., Shi, Y., Li, G., and Ding, E. (2021). Robust and online vehicle counting at crowded intersections. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 3997–4003.

Nyan, B. B., Slembrouck, M., Veelaert, P., and Philips, W. (2020). Distributed multi-class road user tracking in multi-camera network for smart traffic applications. In Blanc-Talon, J., Delmas, P., Philips, W., Popescu, D., and Scheunders, P., editors, *Advanced Concepts for Intelligent Vision Systems (ACIVS)*, pages 517–528. Springer International Publishing.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788.

Rinner, B. and Wolf, W. (2008). An introduction to distributed smart cameras. *Proceedings of the IEEE*, 96(10):1565–1575.

Tran, D. N.-N., Pham, L. H., Nguyen, H.-H., Tran, T. H.-P., Jeon, H.-J., and Jeon, J. W. (2021). A region-and-trajectory movement matching for multiple turn-counts at road intersection on edge device. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 4082–4089.

Wojke, N. and Bewley, A. (2018). Deep cosine metric learning for person re-identification. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 748–756. IEEE.

Wojke, N., Bewley, A., and Paulus, D. (2017). Simple online and realtime tracking with a deep association metric. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 3645–3649.

Xu, L. and Niu, R. (2021). Tracking visual object as an extended target. In *2021 IEEE International Conference on Image Processing (ICIP)*, pages 664–668.