International Journal of Engineering

Journal Homepage: www.ije.ir

A Multilayer Motion Direction Based Model for Tracking Vehicles at Intersections

M. Delavarian*[a], O. Marouzi[b], H. Hassanpour[a]

[a] *Faculty of Computer Engineering, Shahrood University of Technology, Shahrood, Iran*
[b] *Faculty of Electrical & Robotic Engineering, Shahrood University of Technology, Shahrood, Iran*

PAPER INFO

Paper history:
Received 18 March 2020
Received in revised form 20 May 2020
Accepted 26 May 2020

Keywords:
Machine Vision
Motion Flows
Multilayer Model
Multi-object Tracking
Tracking at Intersection
Vehicle Tracking

ABSTRACT

Visual vehicle tracking is an important topic in intelligent transportation systems. Intersections are challenging locations for visual systems to track vehicles which are simultaneously moving in different directions. In addition, normal traffic flow may change at intersections due to accidents. Congestion, occlusion and undetermined motion flows are the nominated challenging issues of vehicle tracking at intersections. In this paper, a method for tracking multiple vehicles is proposed considering the vehicle motion directions to overcome undetermined motion flows. For this purpose, a multilayer model is presented, which assigns each motion flows to distinct layers. Moreover, we introduce different neighborhoods for various layers considering the regular motion flows in a layer. Hence, vehicles entering from the same side of intersection with the same motion direction are assigned to the same layer. Then the tracking is performed on different layers separately. In special cases such as vehicles crossing each other, misdetections or occlusion, the proposed tracking method can predict the vehicles tracks by using the stored tracking history and considering neighborhoods in that layer. Experimental results show consistency between proposed tracking method results and ground truth, also outperformance of other tracking methods in tracking vehicles crossing the intersection.

***doi**: 10.5829/ije.2020.33.10a.12*

1. INTRODUCTION

Visual tracking methods are generally used in various applications to increase the quality of factors such as safety managements and accident avoidance. The demand for fast and efficient tracking system algorithms is increasing for traffic management and roads safety. In recent years, monitoring at intersections has gained attention due to a growth in the accident rate. Several challenging issues were conveyed regarding to Vehicle Tracking at Intersection (VTI) such as different and undetermined motion flows (Figure 1), occlusion and congestion [1, 2]. Driving at intersections is significantly more dangerous in comparison to other locations due to a higher possible rate of conflict between vehicles motion flows (see Figure 1). Therefore, it is of the concern by many researches in the literature. As a result, tracking vehicles is the main phase of vehicle monitoring at intersections.

Vehicle monitoring process consists of two stages: vehicle tracking and vehicle behavior analysis. Active and passive sensors can be used for vehicle sensing at intersection [3]. Cameras are used as one of the passive sensors, and are often used in intersections due to their lower cost and wider field of view. The studying on vehicle tracking is an ongoing research as the systems are not yet reliable and robust for tracking vehicles under highly non-similar circumstances.

Researches have recently paying special attention to vision-based detection and tracking systems [4–8]. Lots of researches have been conducted on vehicle tracking and traffic monitoring in general [9–11]. Tracking at intersection has gained more attention over the past decade. Lately, special reviews are focused on intersection monitoring [1, 2]. In the research done by Datondji et al. [1], they reviewed vision based systems about intersection monitoring including sensing technologies, datasets, vehicles detection, tracking and

*Corresponding Author Email: *delavarian.mohadeseh@shahroodut.ac.ir* (M. Delavarian)

Please cite this article as: M. Delavarian, O. Marouzi, H. Hassanpour, A Multilayer Motion Direction Based Model for Tracking Vehicles at Intersections, International Journal of Engineering (IJE), IJE TRANSACTIONS A: Basics Vol. 33, No. 10, (October 2020) 1939-1950

monitoring methods and their challenges. In another study reported in literature [2], they focused on behavior and safety analysis of vehicles, drivers and pedestrians at intersections besides techniques for automating visual sensing.

Tracking at intersection is usually discussed in a Bayesian framework [12, 13]. Among the Bayesian tracking methods, Kalman filter is one of the most popular algorithms. The Kalman filter tries to estimate the probability of the next state using previous information and measurements. Song and Nevatia [13] proposed a Markov Chain Monte Carlo (MCMC) in order to segment multiple overlapped vehicles into separate ones with respective orientation. In another research, a two level tracker, low-level and high-level, was introduced for VTI [14]. The first part tracks binary blobs. The high-level tracker models the target and estimates the route via the frame information. They used Kalman filter for predicting movements. Their proposed method could not track motionless vehicles and in crowded scenes.

Nateghinia et al. [15] introduced a video-based system for vehicle detection and tracking via dynamic texture modeling for background estimation. They used a point tracking method with weighted recursive least square. Liu et al. [16] proposed a three-dimensional particle filter tracking method. By collaborative tracking via visual information complementation, the robustness of tracking is improved.

For vehicle sensing at intersections, most systems require one or more cameras installed at certain positions which could cover most of the intersection area [17]. In some researches one camera above the ground or fisheye camera is used [18–20]. Wang et al. [19] proposed a real time system for tracking and counting multiple vehicles. They used fisheye camera based on simple feature points. In another research, drone floating camera is used [20]. In their proposed method, in the first phase they used background subtraction. Then, Kalman filter is applied to footage from a drone-floating camera. Another real time vision system at urban crossroad is reported in literature [12]. They used monocular images from pole-mounted video cameras. Their proposed system consists of segmentation with robust background updating and feature based motion tracking method.

A video-based system for obtaining traffic-flow statistics is presented for road intersections by training a deep learning architecture from a pre-trained model [21]. Some studies used more than one camera. Subedi et al. [22] proposed 3D vehicle tracking for a multiple-camera system. A calibration method is introduced, then vehicle silhouettes are detected, and tracking is performed by Kalman filtering.

In the research done by Li et al. [23], they developed an unsupervised vehicle tracking system for urban environments. This method is based on tracklets. Raw tracklets are considered as sample points and are grouped to build different vehicle candidates. Min et al. [24] introduced an approach for tracking multiple vehicles. In this method, they used an improved ViBe algorithm and the gray-scale spatial information for accurate vehicle detection. They used SVM (Support Vector Machine) and CNN (Convolutional Neural Network) classifiers to address occlusions in their tracking. Bedruz et al. [25] proposed an algorithm for detecting and tracking vehicles at intersections in real time. The algorithm is based on blob analysis for main tracking and mean shift kernel tracking when a blob merging occurred.

Some researches track all road users at intersection. Tracking all road users at intersection can be very useful for transportation systems, as all trajectories can be used for monitoring and safety analysis. Jodoin et al. [26] developed a method to track all road users at intersection. At first their method starts from background subtraction to extract the potential a priori unknown road users. Afterwards, each user is tracked by key points inside the detected region. In the Ko-PER project, a dataset for intersection monitoring has been released [27]. This dataset contains laser-scanners and video cameras information. Also, they developed a system for tracking all road users by using all measurements of this dataset including laser-scanners and video frames information. They presented a general-purpose multi-sensor tracking algorithm. Therefore, a multiple-model method is needed in their approach. So, an extension of PHD (Probability Hypothesis Density) filter is proposed [28].

The method proposed by Yang and Bilodeau [29] tracks road users at intersection. Their method combines background subtraction and KFC (Kernelized Correlation Filter) tracker for data association and when an occlusion happens. In another research, a deep learning detection approach is used for object classification and labeling, then labels were employed in tracking users at intersection [30]. After that, they expanded the method to use background subtraction and detection labels by color and class to predict tracks in MOT [31]. Chan et al. [32] proposed city tracker that is a tracking framework in urban traffic scenes that applies a predetermined deep learning based detector and predicts tracks based on DeepSORT.

In this paper, a vehicle tracking method is proposed to overcome undetermined motion flows and the occlusion caused in VTI systems. Our focus is on tracking multiple vehicles at intersection from a camera view. The camera may be installed somewhere around the intersection and has almost a full view. A multilayer model is proposed to distinguish motion flows in distinct layers. The movements are based on the layers acceptable motion flows and the side through which vehicles enter the intersection. Different neighborhoods are introduced for each and distinct layers. Neighborhoods are constructed based on regular and acceptable motion

flows associated with the entering side of an intersection. Acceptable motion flows for an intersection are depicted in Figure 1. After constructing the multilayer and neighborhoods for an intersection, in tracking phase, vehicles entering to each side of intersection are assigned to its associated layer. Tracking is performed in each layer separately based on vehicles movements and the neighborhoods. In case of occlusions or misdetections during tracking, the proposed method considers history of tracks and routes the vehicles based on neighborhoods of layers. The proposed tracking method is based on initial model introduced in literature [33].

The rest of the paper is organized as follows: In section 2 the proposed tracking method is described. Experimental results are presented in section 3. In section 4 experimental scenarios are discussed. Section 5 concludes the paper.

2. THE PROPOSED METHOD

VTI is a challenging task due to the involvement of various parameters, including the simultaneous movement of multiple vehicles and pedestrians. Once vehicles cross the intersection, different directions are possible to be taken. Figure 1 depicts different motion flows at an intersection from each entry-exit side. It can be seen that motion flows may collide with each other, consequently accidents may happen.

One standard approach in multi-object tracking algorithms is tracking-by-detection [34]. Detecting is performed separately. Therefore, detection performance does not affect tracking performance. The proposed method initially detects vehicles only, (excluding pedestrians or bicycles), using an existing detection method. It is assumed that vehicles crossing intersection only follow predefined directions (see Figure 1).

2. 1. Multilayer Model Construction A multilayer model is employed for tracking, in which different motion flows are distributed in distinct layers.

The proposed model assigns one layer for each entry-exit side, so based on the number of entry-exit sides, we have layers, $1 \le l \le L$, L is the total number of layers. In this study, we assume that each intersection has four entry-exit sides, so $L=4$. However, there is no limitation in the total number of layers (entry-exit sides).

The area of tracking, i.e. the field of view (FOV) is indicated manually. The sides of FOV, that are associated with each entry side and layer, are labeled clockwise, say from 1 to 4 starting from top side as it can be seen in Figure 2. After that a grid is constructed over the ground of intersection as it is shown in Figure 2. The size of grid cells is determined by the distance between grid lines. This distance is a user determined, variable in pixels, which shall be called the *step* parameter. The y-intercept of the grid lines that is called *y Intercept Change* should be modified as shown in Equation (1), where a is the line slope.

$$y\ Intercept\ Change = step \Big/ \sqrt{1/(1+a^2)} \quad (1)$$

Afterwards all crossing points of grid lines are stored in matrix p. Then, a grid cell matrix C is created where each $C(i,j)$ is associated with four points of the matrix p like $[\ p(i,j),\ p(i,j+1),\ p(i+1,j+1),\ p(i+1,j)\ ]$. So, each cell of grid matrix C represents a part of the intersection. The sample grids for the values 20 and 40 of *step* parameter for Ko-PER dataset intersection [27] are shown in Figure 3. As can be seen with a smaller *step* parameter, the grid matrix has more cells with a smaller size. The size and number of cells, as results of the value of *step* parameter, have direct effect in tracking results and run time that will be discussed in section 3.

Vehicles in each layer can only move in permissible directions. Also, layers are independent of each other, which means vehicles cannot move between layers. We introduced neighborhoods to model allowed acceptable motion directions for each layer separately. Hence, the difference between the layers is equivalent to the cell's neighborhoods in each layer. If a vehicle is at cell $C(i,j)$ of layer k at time t, it can be at cell $C(i,j)$ or cell $C(i',j')$

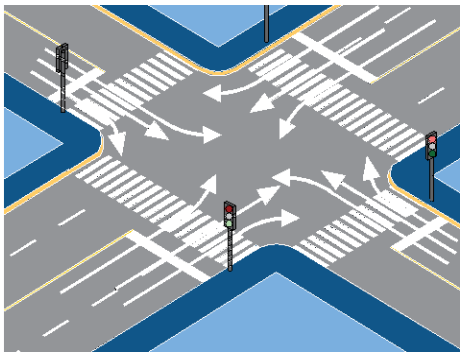

Figure 1. Motion directions of each side for a sample intersection

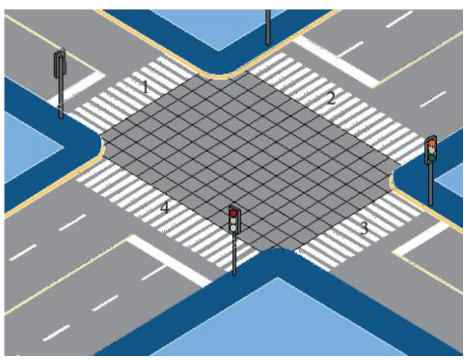

Figure 2. Labeled sides, layers and grid cells of FOV for a sample intersection

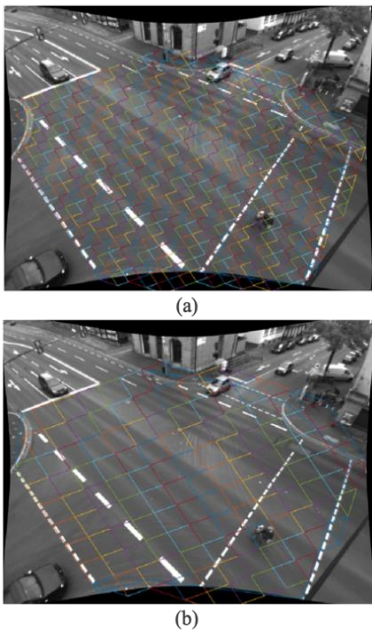

Figure 3. Grid cells of FOV in Ko-PER dataset for *step* parameter equals to (a) 20, (b) 40

where *(i',j')* is neighbors of cell (i,j) at time *(t+1)* and layer *k*. Neighbors are based on the layer's motion directions. For example, neighbors of *(i,j)* in layer 1 can be *(i, j-1)* (for turning left), *(i,j+1)* (for turning right), *(i+1,j)* (for going straight) and finally *(i+1,j-1)* and *(i+1,j+1)* (for turning left and right, respectively). So, two of the neighbors are adjacent and the other three are in front of those two and itself (the front direction is determined relative to the movement direction of the layer). The set of these five neighborhood cells are called *Near* neighborhood. This neighborhood is more useful when the *step* parameter is big or the recording frame rate is high.

When the frame rate is low or the value of *step* parameter is small, a vehicle can move more than one neighboring cell. Another case is that due to variations of camera view angle for different grid cells, there is some confusion in positioning far and near cells to the camera. Therefore, in addition to the *Near* neighborhood, a bigger neighborhood is needed. Hence, *Far* neighborhood is introduced. In the *Far* neighborhood, each cell of the *Near* neighborhood has another neighboring cell in the motion directions of that layer. For instance, in layer 4 (left side) cell *C(i,j)* has *(i,j+1)* as its neighbors in *Near* neighborhood. Cell *C(i,j+2)* is added for *Far* neighborhood and this goes on for other neighbors. The *Near* and *Far* neighborhoods of layer 2 are defined in Table 1. The index *(i,j)* is the cell where the vehicle is currently located. The cells in the gray background are *Near* neighbors and the cells with white background are *Far* neighbors. Each layer has different *Near* and *Far* neighborhood cells based on its motion direction flows. The neighborhoods of other layers are computed the same way as it shows for layer 2 in Table 1 and layer 1 in previous paragraph.

2. 2. The Tracking Algorithm Multilayer model including layers and neighborhoods is constructed for an intersection before tracking is performed. During tracking, when a vehicle enters the intersection, a new tag number, *(tag>0)*, is assigned to it. Also, the vehicle is assigned to the associated layer. Afterwards, tracking is performed for each layer separately. During the tracking phase, track records are stored. Track records for vehicles consist of main information and are saved in the format of Equation (2):

$$TrackRecord=[time, layer, tag, i, j, middlePoint] \quad (2)$$

where *time* is the frame number or tracking time, *layer* and *tag* are the layer number and vehicle identification number, respectively; *i* and *j* are cell index of the grid matrix *C* which is the vehicle position on FOV and *middlePoint* for middle point of their bounding boxes.

For the next frames, the detected vehicles may remain motionless or move to one of their neighboring cells. Each grid cell can only contain one vehicle at a time. For the following frames, we looked for previously recognized vehicles' trajectories in neighborhoods of detected vehicles cells. This repeats for every detected vehicle on next frames. When a vehicle enters the intersection, after incrementing the previous tag number, it is assigned to the vehicle. However, if it had been entered previously, it already has a trajectory, layer and tag number. So, it gets the tag and layer number from the trajectory that this vehicle belongs to based on the above procedure.

We introduced two neighborhoods including *Near* and *Far*. Therefore, for checking neighborhoods, three cases can be considered. In the first case, only the *Near* neighborhood is checked. This neighborhood can be used when the *step* parameter is big or scenes are crowded. In the second one, the *Far* neighborhood is checked which is used when *step* parameter is small. In the third case, both neighborhoods are considered. At first, the temporary destination is checked based on *Near* neighborhood and if there is no track in *Near* neighborhood's previous frame then the *Far* neighborhood is checked.

TABLE 1. *Near* and *Far* neighborhoods for right side layer (Layer 2)

i-2 , j-2	*i-2 , j-1*	*i-2 , j*
i-1 , j-2	*i-1 , j-1*	*i-1 , j*
i , j-2	*i , j-1*	*i , j*
i+1 , j-2	*i+1 , j-1*	*i+1 , j*
i+2 , j-2	*i+2 , j-1*	*i+2 , j*

The previous steps are performed for vehicles contained in each frame. Each track ends when the vehicle crosses other side lines or the detections are out of FOV.

One of the advantages of the proposed tracking algorithm is that no collision occurs. Because during the tracking phase vehicles that come from different sides are assigned to different layers. The other advantage is that if an occlusion or misdetection occurs according to the history of tracks and checking the neighborhoods based on *Far* neighborhood of *Near* neighborhood cells, the tracking algorithm can predict tracks. Finally, at the end or any time in the middle of tracking, all tracks can be computed by using the tag number and saved history of the tracks. The final proposed tracking method based on the multilayer model, *Near* and *Far* neighborhoods is summarized in Algorithm 1. The preprocessing is done once for each intersection and tracking is performed for each frame and detection.

For a better demonstration of the proposed tracking method, two sample consecutive frames during tracking are shown in Figure 4. At time *t* and *t+1*, middle point of detected and tracked vehicles are shown. Different colors show vehicles in different layers. *Near* neighborhood cells of some vehicles are drawn with red dots. The correspondent arrows with the same layer color show the correspondent tracked vehicle in frame *t+1*. As the figure illustrates, the vehicles appear in their neighborhoods based on the layers' motion flows.

3. EXPERIMENTAL RESULTS

The proposed tracking method contains a number of parameters including values of *step* and different neighborhoods. By the *step* value, resolution of tracking is controlled. Before the tracking phase, the multilayer model is constructed as a preprocessing phase and is computed just once for each intersection.

ALGORITHM 1. Proposed tracking method based on multilayer model

```
Preprocessing:
    // Multilayer model construction
    Indicating FOV
    Labeling entry-exit sides clockwise starting from top side
    Determining step parameter value
    Constructing grid cells over the ground of intersection
    Assigning a layer to each entry-exit side
    Determining Near and Far neighborhoods for all layers and cells
Tracking:
for each frame t do
    Extract detections from a detector for frame t
    for each detection det in t do
        if det entering the intersection then
            l ← layer number
            tag ← tag number
            middlePoint ← the centroid of det
            [i,j] ← the cell that middlePoint is located at
            Create TrackRecord = [t, layer, tag, i, j, middlePoint]
        else
            middlePoint ←the centroid of det
            [i',j'] ← the cell that middlePoint is located at
            if (check Near neighborhood based on layer and tag number) then
                Add TrackRecord = [t, layer, tag, i', j', middlePoint]
            else
                if (check Far neighborhood based on layer and tag number) then
                    Add TrackRecord = [t, layer, tag, i', j', middlePoint]
                else
                    //occlusion and misdetection handling
                    if (history checking and predicting this detection belongs to which track) then
                        Add TrackRecord = [t, layer, tag, i', j', middlePoint]
                    end if
                end if
            end if
        end if
    end for
end for
Tracks ← Compute tracks based on track records, tags, and multilayer model
```

Figure 4. Two sample consecutive frames showing tracked vehicles and their correspondence at time *t* and *t+1*

3. 1. The Datasets The Ko-PER dataset is used in our experiments [27] as the main dataset. In Germany, the intersection is an actual four way crossing and cameras are monochrome. The first camera has a complete FOV such that the whole intersection information can be acquired. In our experiments, sequence 1 is used that it is divided into four parts named 1a, 1b, 1c and 1d that have the same duration and are considered separately. The detection results that have been released with the Ko-PER dataset is used and extracted from the dataset viewer. In research done by Meissner et al. [28], information from all views and laser-scanners are used, so the detection results are accurate enough. The sequences are considered crowded since there are several vehicles that are crossing the street at the same time. The sample frame and view of the Ko-PER dataset is shown in Figure 3.

Another dataset that is used for comparison is Urban Tracker dataset [26, 35]. It includes vehicles and pedestrians for tracking. Here from the dataset, three sequences are selected as Sherbrooke, Rouen, St-Marc. The sample frames of Urban Tracker dataset are presented in Figure 5. As can be seen the cameras placement and views are different in each sequence. However, they are less crowded in comparison with Ko-PER dataset.

3. 2. Evaluation Metrics For evaluating multi-object trackers CLEAR MOT metrics are known as a standard metric [36]. In CLEAR MOT metrics, the accuracy of trackers is defined by Multi-Object Tracking Accuracy (MOTA) that is defined as Equation (3):

$$MOTA = 1 - \frac{\sum_t (m_t + fp_t + mme_t)}{\sum_t g_t} \quad (3)$$

where g_t is the number of ground truth detections, fp_t is the number of false positives, m_t is the number of misdetections, and mme_t is the number of identity switches. The MOTA is computed by comparing tracking results with Ground Truth (GT).

There exists quality metrics that were presented in [37] to compare trackers all by comparison with GT and are defined as follows:

- Mostly Tracked (MT) - the percentage of tracks that are successfully tracked for more than 80%.
- Mostly Lost (ML) - the percentage of tracks that are tracked for less than 20%.
- Partially Tracked (PT) - the percentage of tracks that are tracked between 20 and 80% *(1-MT-ML)*.
- Identity Switch (IDS) - the number of times two tracks switch their IDs.

For these metrics, higher MT and lower ML, PT, IDS is preferred.

3. 3. Evaluation of the Proposed Tracking Method For validation of the proposed tracker and parameters' effect, Ko-PER dataset is selected since it is the longest and crowded one. During our experiments to assess the proposed tracking algorithm, we noted that for the Ko-PER dataset, due to camera placement, vehicles that enter from the right side and make a sharp turn to the bottom side can easily be lost. To solve this, we modified neighborhoods of the right-side layer (layer 2). We added some neighbors to the right side of both *Near* and *Far* neighborhoods.

To assess the neighborhood effect, experiments are done on three cases considering neighborhoods: *Near*, *Far* or both neighborhoods. Besides to neighborhoods, another important parameter is the value of *step*. To evaluate its effect, the tracking procedure is performed

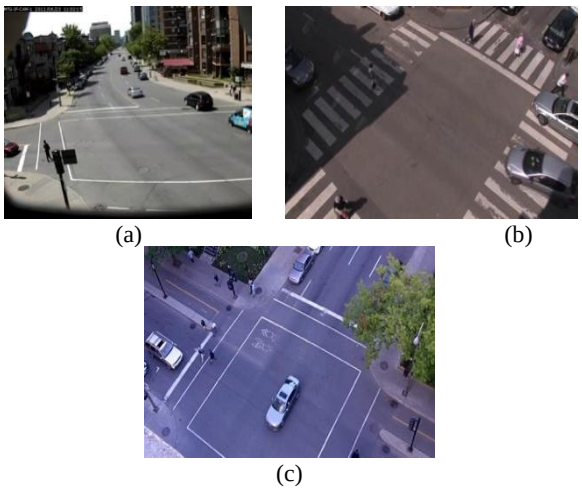

Figure 5. Urban Tracker dataset (a) Sherbrooke, (b) Rouen, (c) St-Marc

for *step* values from 10 to 40 in steps of 5 to demonstrate its effect. In Figure 3, FOV cells of Ko-PER dataset for *step* values 20 and 40 are depicted. Here, the effect of *step* value in the number and size of grid cells can be seen visually. As mentioned for predicting the tracks when collision, occlusion or misdetections occur, the history of tracks is checked. The parameter for checking the history is α which means checking $(t\text{-}\alpha)$ tracks history. For all experiments, α is equal to 2.

Three cases of neighborhoods and different *step* values are considered to investigate the effect of them in tracking results. Each case is reviewed separately.

Case 1. Tracking based on *Near* neighborhood: The results based on MT, ML, PT and IDS for all four sequences of Ko-PER dataset are revealed in Figure 6 for comparison. As the figure shows in all four sequences for small values of *step*, the value of ML and PT is bigger. The reason is that when only checking *Near* neighborhood with a small value of *step* that results in small cells, a vehicle could cross more than just one cell of its neighborhood. So, it does not exist in the cell that is predicted based on *Near* neighborhood. However, as *step* value increases, this does not happen. As a result, using *Near* neighborhood for bigger values of *step* can lead to better results.

Case 2. Tracking based on *Far* neighborhood: The tracking results are shown in Figure 7. More identity switches happen when *step* value is bigger in comparison with when using just *Near* neighborhood (see the difference between Figures 6 and 7, IDS column especially for bigger value of *step*). A lot of identity switches mean tracks appear in each other's neighborhood and this is the result of checking bigger neighborhood when it is not necessary. Also, we have fewer ML and PL tracks when *step* is small in comparison with using just near neighborhood. Therefore, this neighborhood works better with smaller values of *step*.

Case 3. Tracking using both neighborhoods: The tracking is performed based on both neighborhoods respectively. First, *Near* neighborhood is checked and if a trajectory cannot be found, then the *Far* neighborhood is checked. The results are presented in Figure 8. This case has the benefit of the two previous cases. When the *step* value is small, ML and PT are less. Also, when *step* has bigger values identity switches occur less. However, still ML and PT tracks and IDS exist especially when *step* value is very small or very big. The reason is for parts of the intersection that are far from the camera, from camera viewpoint the movement seems bigger as the size of the

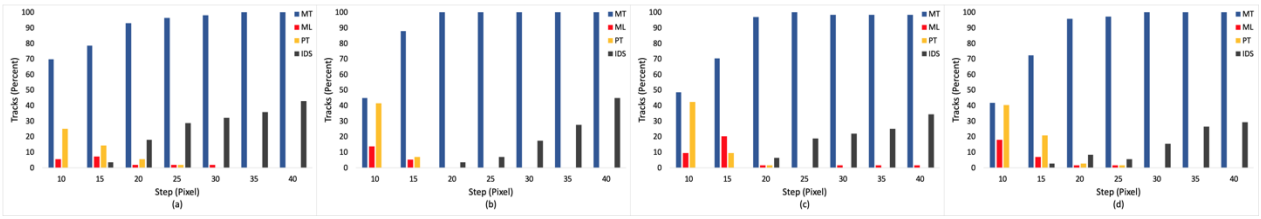

Figure 6. Tracking results based on *Near* neighborhood and different values of *step* (a) sequence 1a, (b) sequence 1b, (c) sequence 1c, (d) sequence 1d

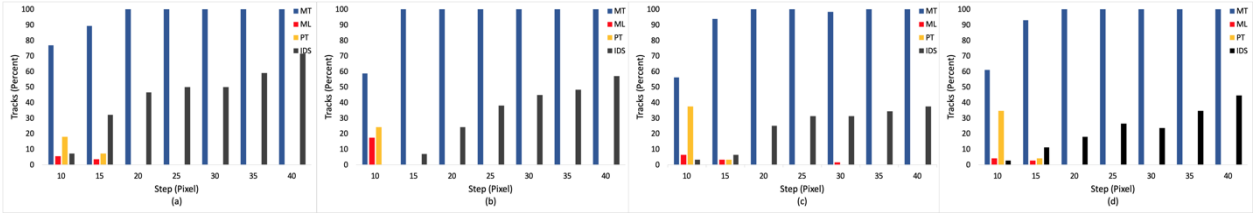

Figure 7. Tracking results based on *Far* neighborhood and different values of *step* (a) sequence 1a, (b) sequence 1b, (c) sequence 1c, (d) sequence 1d

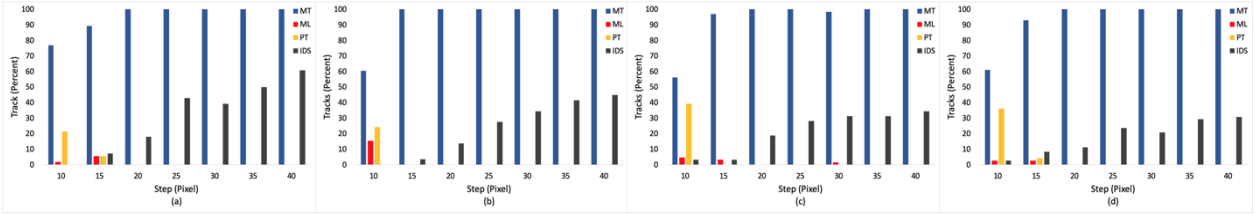

Figure 8. Tracking results based on both neighborhoods and different values of *step* (a) sequence 1a, (b) sequence 1b, (c) sequence 1c, (d) sequence 1d

cells are the same. For identity switches that happen in bigger values of *step* the same reason applies.

After studying the results of the three cases, the best strategy is case 3 using both *Near* and *Far* neighborhoods, respectively. As discussed before, *step* is an important parameter, smaller values of *step* result in more cells with smaller size. The importance can be demonstrated by the results obtained so far. Smaller or bigger values of *step* are not suitable choices and affect the results negatively (more ML and PT or more IDS respectively). The best *step* value for the Ko-PER dataset intersection is 20 or 25. It provides the highest number of MT tracks and fewer ML, PT or IDS in comparison with other values.

After determining the best neighborhood strategy and value of *step* parameter, all evaluation metric for all four sequences of Ko-PER dataset are presented in Table 2. In all sequences above 96% of trajectories are tracked completely and the accuracy is above 0.9; however, still identity switches happen.

To evaluate the tracker runtime, we run the tracking algorithm for all four sequences for *step* values from 10 to 40. Out of the three possible cases for checking the neighborhood, the third case is selected as previous results show its better performance. Each runtime is the average of running the algorithm for 10 times. Also, we divided the whole sequence to batches of 60 frames. The last frame of each batch and first frame of the next batch overlap so that the tracks from the previous batch can continue correctly. Using batches has the advantage that small tracks can be acquired quickly, since vehicles cross the intersection in frames that are a lot fewer than the whole sequence. Our proposed multilayer tracker was implemented in MATLAB on a system with 2.2GHz Intel Corei7 and 16GB of RAM. Since the camera records 25 Frames Per Second (25FPS) [27], and we considered 60 frames in each batch, the algorithm running time up to 2.4 seconds per batch is considered real time.

Runtimes for all sequences are presented in Figure 9. As the figure illustrates for *step* value equals to 10, as a result of too many small cells, the running time has a high value. However, for *step* parameter equals to 40, the number of cells is less so the algorithm running time decreases too. For *step* value equals to 20 and more, the tracks can be acquired in real time in this dataset.

Based on the results of neighborhoods and execution time, setting the *step* parameter equal to 20 is the best choice for this intersection. Also, the results show that our proposed tracking algorithm can be applied in real time applications.

Sample tracking results are shown in Figure 10. Tracks in different color belong to different layers, red for layer 1 (top side layer), green for layer 2 (right side layer), black for layer 3 (bottom side layer), yellow for layer 4 (left side layer). The blue color shows that vehicles are out of FOV. The first row in Figure 10 shows tracks while tracking is performed, and the second row shows final tracks after tracking is done. The tracks are from Ko-PER dataset for *step* value equals to 20 and using both neighborhoods.

3.4 Comparison with Other Methods In this part, to validate our proposed tracker, it is compared with other trackers. The results are reported on Urban Tracker dataset for Sherbrooke, Rouen, St-Marc videos in Table 3. The cars part of the Urban Tracker dataset is selected as here tracking vehicles at intersection is considered. One of the trackers is Traffic Intelligence that is a feature-

TABLE 2. All evaluation metrics for sequences of Ko-PER dataset for proposed tracking method (parameters: *step* equals to 20 and using both neighborhoods)

	MT	ML	PT	IDS	MOTA
Sequence 1a	98.25	1.75	0.00	17.54	0.9411
Sequence 1b	100.00	0.00	0.00	13.79	0.9452
Sequence 1c	96.92	1.54	1.54	18.46	0.9080
Sequence 1d	100.00	0.00	0.00	11.11	0.9497

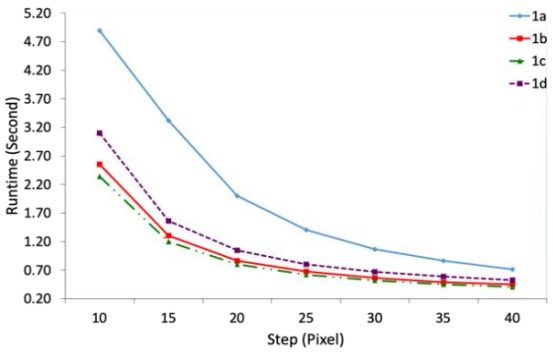

Figure 9. Runtime for all sequences of Ko-PER dataset with different *step* values using both neighborhoods

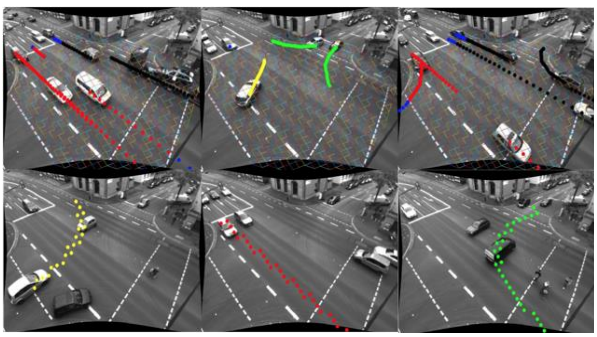

Figure 10. Sample tracking results, first row: tracks while tracking is performed, second row: sample final tracks

based tracking algorithm [38]. The other, which is Urban Tracker, is introduced with the dataset and tracks all road users, detection are done by performing background subtraction and tracking is performed by Kalman filter [26, 35]. Mendes et al. [39] combined background subtraction with blob detection and optical flow. MKCF (Multiple Kernelized Correlation Filter) tracker is introduced in literature [29], that applies background subtraction and multiple KCF trackers. As can be seen, our proposed multilayer tracker outperforms the accuracy of the other trackers in all three videos of the dataset.

The MT, ML and PT of Urban Tracker dataset for proposed tracker and two other trackers are reported in Table 4 for better comparison.

4. DISCUSSION

Vehicles are detected before tracking, and detections are not always accurate. For demonstrating the effect of misdetections (vehicles are not detected) and false positives (other objects being identified as vehicles) that are caused by reasons such as occlusion, shadow and collision, some experiments are performed and the results are invstigated here. To study how these, affect the tracking results and how the proposed tracking method deals with them, tracking sequences are recreated from the original ones. The reported results are the average of 10 runs with different missed and false detections on each run. The other tracking parameters are as follows: third case (using both neighborhoods respectively), *step* value equals to 20 and checking the history of two previous frames. Checking more frames for history leads to better results but because real time execution is pursued, it is not used.

Different scenarios are created. The first one is applying misdetections. So, detections are deleted randomly. We considered 2, 5 and 10% of misdetections. As a result, sequences detections are recreated based on these values randomly. The results are presented in Table 5. As it shows for 2% misdetections, tracking results do not show much difference from the original one (see Table 2). But, as the number of misdetections increases so does the number of ML and PT tracks, and MOTA decreases. However, even in 10% misdetections most of the tracks are found and the accuracy is around 0.8.

The other scenario is applying false detections. False detections are created randomly with two strategies called addition and replacement. In the addition strategy, detections that are not vehicles are added randomly. In the replacement strategy, a number of detections are deleted from their original frames and are added randomly to other frames detections, so misdetections are created simultaneously. The results are presented in Table 6 for 2, 5 and 10% of false detections with addition strategy and in Table 7 for replacement strategy.

The results of the addition strategy demonstrate that it does not affect the evaluation metrics. Even with 10% false detections with addition strategy, the results are unaffected. Therefore, the proposed tracker is robust to this strategy. However, because false positives are added, MOTA decreases. Table 7 shows the results of false detections created with the replacement strategy, since misdetections are added at the same time, there are more ML tracks, and the results are similar to just adding misdetections. But here, since there are miss and false detections at the same time, MOTA decreases more.

The proposed tracking method based on multilayer model thoroughly tracks vehicles by considering their motion flows without any visual features, although in

TABLE 3. Trackers comparison for Urban Tracker dataset

	MOTA				
	Proposed Multilayer Tracker	**MKCF [29]**	**Urban Tracker[26]**	**Mendes et al. [39]**	**Traffic Intelligence [38]**
Sherbrooke (Cars)	0.962	0.789	0.887	0.707	0.825
Rouen (Cars)	0.980	0.813	0.897	0.918	0.185
St-Marc (Cars)	0.939	0.590	0.889	0.713	-0.178

TABLE 4. Trackers MT, ML and PT comparison for Urban Tracker dataset

	Proposed Multilayer Tracker			**Urban Tracker[26]**			**Traffic Intelligence [38]**		
	MT (%)	**ML (%)**	**PT (%)**	**MT (%)**	**ML (%)**	**PT (%)**	**MT (%)**	**ML (%)**	**PT (%)**
Sherbrooke (Cars)	96.2	1.8	2.0	85.0	5.0	10.0	60.0	25.0	15.0
Rouen (Cars)	98.0	0.0	2.0	75.0	0.0	25.0	37.5	37.5	25.0
St-Marc (Cars)	92.9	1.5	5.5	64.3	7.1	28.6	60.7	17.9	21.4

specific situations losing tracks and identity switch could happen. Since detections and tracks are distributed among different layers, as the results show, tracking time becomes real time.

TABLE 5. Tracking results for all sequences with misdetections of Ko-PER dataset

	MT	ML	PT	IDS	MOTA
Misdetection 2%					
Sequence 1a	100.00	0.00	0.00	21.05	0.9177
Sequence 1b	100.00	0.00	0.00	17.24	0.9324
Sequence 1c	93.85	3.08	3.08	13.85	0.9107
Sequence 1d	98.61	0.00	1.39	11.11	0.9382
Misdetection 5%					
Sequence 1a	89.47	1.75	8.77	17.54	0.9140
Sequence 1b	93.10	5.17	1.72	17.24	0.8702
Sequence 1c	95.38	4.62	0.00	18.46	0.8948
Sequence 1d	98.61	0.00	1.39	11.11	0.9364
Misdetection 10%					
Sequence 1a	89.47	5.26	5.26	24.56	0.8512
Sequence 1b	84.48	12.07	3.45	17.24	0.8518
Sequence 1c	76.92	10.77	12.31	24.62	0.7729
Sequence 1d	86.11	8.33	5.56	11.11	0.8658

TABLE 6. Tracking results for all sequences of Ko-PER dataset with false detections with addition strategy

	MT	ML	PT	IDS	MOTA
False detection 2%					
Sequence 1a	98.25	1.75	0.00	17.54	0.865
Sequence 1b	100.00	0.00	0.00	13.79	0.9317
Sequence 1c	98.46	0.00	1.54	18.46	0.8948
Sequence 1d	100.00	0.00	0.00	11.11	0.9355
False detection 5%					
Sequence 1a	98.25	1.75	0.00	17.54	0.9348
Sequence 1b	100.00	0.00	0.00	13.79	0.9135
Sequence 1c	98.46	1.54	0.00	18.46	0.8737
Sequence 1d	100.00	0.00	0.00	11.11	0.9213
False detection 10%					
Sequence 1a	98.25	1.75	0.00	17.54	0.8023
Sequence 1b	100.00	0.00	0.00	13.79	0.8846
Sequence 1c	98.46	1.54	0.00	18.46	0.8519
Sequence 1d	100.00	0.00	0.00	11.11	0.8902

TABLE 7. Tracking results for all sequences of Ko-PER dataset with false detections with replacement strategy

	MT	ML	PT	IDS	MOTA
False detection 2%					
Sequence 1a	98.25	1.75	0.00	17.54	0.9252
Sequence 1b	98.28	0.00	1.72	13.79	0.9151
Sequence 1c	100.00	0.00	0.00	21.54	0.8800
Sequence 1d	98.61	0.00	1.39	11.11	0.9308
False detection 5%					
Sequence 1a	94.74	0.00	5.26	21.05	0.8843
Sequence 1b	89.66	6.90	3.45	24.14	0.8567
Sequence 1c	92.31	3.08	4.62	24.62	0.8365
Sequence 1d	97.22	0.00	2.78	5.56	0.9097
False detection 10%					
Sequence 1a	87.72	3.51	8.77	24.56	0.8144
Sequence 1b	86.21	5.17	8.62	17.24	0.7807
Sequence 1c	86.15	4.62	9.23	21.54	0.7774
Sequence 1d	86.11	8.33	5.56	15.28	0.8118

5. CONCLUSON

In this paper, we proposed a new method for tracking vehicles at intersection. It uses a multilayer model for tracking, a distinct layer for each entering side of the intersection. The layers are constructed considering the permissible motion flows and neighborhoods for the associated entering side. Then in tracking procedure each entering vehicle is detected and assigned to its associated layer and the movements are tracked. Indeed, tracks can be predicted as each vehicle has permissible directions to follow depending to its entering side. To evaluate performance of the proposed method, we tested the method with various scenarios, each of which imposed up to 10% false and misdetections on the Ko-PER dataset. The results showed that the proposed method can thoroughly track vehicles at intersection in many scenarios. Also, it outperforms other trackers. Therefore, we can claim that the proposed multilayer tracking method is computationally effective and can be used in real time tracking applications.

6. REFRENCES

1. Datondji, S. R. E., Dupuis, Y., Subirats, P., and Vasseur, P. "A Survey of Vision-Based Traffic Monitoring of Road Intersections." ***IEEE Transactions on Intelligent Transportation Systems***, Vol. 17, No. 10, (2016), 2681–2698. https://doi.org/10.1109/TITS.2016.2530146

2. Shirazi, M. S., and Morris, B. T. "Looking at Intersections: A Survey of Intersection Monitoring, Behavior and Safety Analysis of Recent Studies." ***IEEE Transactions on Intelligent Transportation Systems***. Vol. 18, No. 1, (2017), 4-24. https://doi.org/10.1109/TITS.2016.2568920
3. Gandhi, T., and Trivedi, M. M. "Pedestrian protection systems: Issues, survey, and challenges." ***IEEE Transactions on Intelligent Transportation Systems***, Vol. 8, No. 3, (2007), 413–430. https://doi.org/10.1109/TITS.2007.903444
4. Buch, N., Velastin, S. A., and Orwell, J. "A review of computer vision techniques for the analysis of urban traffic." ***IEEE Transactions on Intelligent Transportation Systems***. Vol. 12, No. 3, (2011), 920-939 https://doi.org/10.1109/TITS.2011.2119372
5. Sivaraman, S., and Trivedi, M. M. "Looking at vehicles on the road: A survey of vision-based vehicle detection, tracking, and behavior analysis." ***IEEE Transactions on Intelligent Transportation Systems***, Vol. 14, No. 4, (2013), 1773–1795. https://doi.org/10.1109/TITS.2013.2266661
6. Hasanzadeh, R. P. R., Shahrouzi, S. N., and Mahdavi, M. "A Novel Method for Tracking Moving Objects using Block-Based Similarity." ***International Journal of Engineering - Transactions B: Applications***, Vol. 22, No. 1, (2009), 35–42. Retrieved from http://www.ije.ir/article_71756.html
7. Sadegh Moghadasi, S., and Faraji, N. "An efficient target tracking algorithm based on particle filter and genetic algorithm." ***International Journal of Engineering - Transactions A: Basics***, Vol. 32, No. 7, (2019), 915–923. https://doi.org/10.5829/ije.2019.32.07a.03
8. Jain, N. K., Saini, R. K., and Mittal, P. A review on traffic monitoring system techniques. ***Advances in Intelligent Systems and Computing***, Vol. 742, (2019), 569-577. https://doi.org/10.1007/978-981-13-0589-4_53
9. Kastrinaki, V., Zervakis, M., and Kalaitzakis, K. "A survey of video processing techniques for traffic applications." ***Image and Vision Computing***, Vol. 21, No. 4, (2003), 359–381. https://doi.org/10.1016/S0262-8856(03)00004-0
10. Yilmaz, A., Javed, O., and Shah, M. "Object tracking: A survey." ***ACM Computing Surveys***. Vol. 38, No. 4, (2006). https://doi.org/10.1145/1177352.1177355
11. Sun, Z., Bebis, G., and Miller, R. "On-road vehicle detection: A review." ***IEEE Transactions on Pattern Analysis and Machine Intelligence***. Vol. 28, No. 5, (2006), 694-711. https://doi.org/10.1109/TPAMI.2006.104
12. Messelodi, S., Modena, C. M., and Zanin, M. "A computer vision system for the detection and classification of vehicles at urban road intersections." ***Pattern Analysis and Applications***, Vol. 8, No. 1–2, (2005), 17–31. https://doi.org/10.1007/s10044-004-0239-9
13. Song, X., and Nevatia, R. "Detection and tracking of moving vehicles in crowded scenes." In 2007 IEEE Workshop on Motion and Video Computing, WMVC, (2007). https://doi.org/10.1109/WMVC.2007.13
14. Veeraraghavan, H., Masoud, O., and Papanikolopoulos, N. P. "Computer vision algorithms for intersection monitoring." ***IEEE Transactions on Intelligent Transportation Systems***, Vol. 4, No. 2, (2003), 78–89. https://doi.org/10.1109/TITS.2003.821212
15. Nateghinia, E., and Moradi, H. "Video-based multiple vehicle tracking at intersections." In 2014 2nd RSI/ISM International Conference on Robotics and Mechatronics, ICRoM 2014, (2014), 215–220. Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/ICRoM.2014.6990903
16. Liu, L., Xi, Z., and Sun, Q. "Multi-vision tracking and collaboration based on spatial particle filter." ***Journal of Visual Communication and Image Representation***, Vol. 59, (2019), 316–326. https://doi.org/10.1016/j.jvcir.2018.12.050
17. Shi, Y., and Real, F. D. "Smart cameras: Fundamentals and classification." In Smart Cameras (pp. 19–34). Springer. https://doi.org/10.1007/978-1-4419-0953-4_2
18. Liu, L., Xing, J., and Ai, H. "Multi-view vehicle detection and tracking in crossroads." In 1st Asian Conference on Pattern Recognition, ACPR, (2011), 608–612. https://doi.org/10.1109/ACPR.2011.6166688
19. Wang, W., Gee, T., Price, J., and Qi, H. "Real time multi-vehicle tracking and counting at intersections from a fisheye camera." In Proceedings - 2015 IEEE Winter Conference on Applications of Computer Vision, WACV, (2015), 17–24. https://doi.org/10.1109/WACV.2015.10
20. Lira, G., Kokkinogenis, Z., Rossetti, R. J. F., Moura, D. C., and Rúbio, T. "A computer-vision approach to traffic analysis over intersections." In IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC, (2016), 47–53. https://doi.org/10.1109/ITSC.2016.7795530
21. Dey, B., and Kundu, M. K. "Turning video into traffic data - An application to urban intersection analysis using transfer learning." ***IET Image Processing***, Vol. 13, No. 4, (2019), 673–679. https://doi.org/10.1049/iet-ipr.2018.5985
22. Subedi, S., and Tang, H. "Development of a multiple-camera 3D vehicle tracking system for traffic data collection at intersections." ***IET Intelligent Transport Systems***, Vol. 13, No. 4, (2019), 614–621. https://doi.org/10.1049/iet-its.2018.5163
23. Li, C., Chiang, A., Dobler, G., Wang, Y., Xie, K., Ozbay, K., Ghandehari, M., Zhou, J., and Wang, D. "Robust vehicle tracking for urban traffic videos at intersections." In 2016 13th IEEE International Conference on Advanced Video and Signal Based Surveillance, AVSS, (2016), 207–213. https://doi.org/10.1109/AVSS.2016.7738075
24. Min, W., Fan, M., Guo, X., and Han, Q. "A New Approach to Track Multiple Vehicles with the Combination of Robust Detection and Two Classifiers." ***IEEE Transactions on Intelligent Transportation Systems***, Vol. 19, No. 1, (2018), 174–186. https://doi.org/10.1109/TITS.2017.2756989
25. Bedruz, R. A., Sybingco, E., Bandala, A., Quiros, A. R., Uy, A. C., and Dadios, E. "Real-time vehicle detection and tracking using a mean-shift based blob analysis and tracking approach." In HNICEM 2017 - 9th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management (Vol. 2018), (2018), 1–5. https://doi.org/10.1109/HNICEM.2017.8269528
26. Jodoin, J. P., Bilodeau, G. A., and Saunier, N. "Tracking All Road Users at Multimodal Urban Traffic Intersections." ***IEEE Transactions on Intelligent Transportation Systems***, Vol. 17, No. 11, (2016), 3241–3251. https://doi.org/10.1109/TITS.2016.2545245
27. Strigel, E., Meissner, D., Seeliger, F., Wilking, B., and Dietmayer, K. "The Ko-PER intersection laserscanner and video dataset." In 2014 17th IEEE International Conference on Intelligent Transportation Systems, ITSC, (2014), 1900–1901. https://doi.org/10.1109/ITSC.2014.6957976
28. Meissner, D., Reuter, S., Strigel, E., and Dietmayer, K. "Intersection-based road user tracking using a classifying multiple-model PHD filter." ***IEEE Intelligent Transportation Systems Magazine***, Vol. 6, No. 2, (2014), 21–33. https://doi.org/10.1109/MITS.2014.2304754
29. Yang, Y., and Bilodeau, G. "Multiple Object Tracking with Kernelized Correlation Filters in Urban Mixed Traffic." In 2017 14th Conference on Computer and Robot Vision (CRV), (2017), 209–216. https://doi.org/10.1109/CRV.2017.18
30. Ooi, H.-L., Bilodeau, G.-A., Saunier, N., and Beaupré, D.-A. "Multiple Object Tracking in Urban Traffic Scenes with a Multiclass Object Detector." In Advances in Visual Computing (pp. 727–736). Cham: Springer International Publishing.

31. Ooi, H.-L., Bilodeau, G.-A., and Saunier, N. "Tracking in Urban Traffic Scenes from Background Subtraction and Object Detection." In Image Analysis and Recognition (pp. 195–206). Springer International Publishing.
32. Chan, Z. Y., and Suandi, S. A. "City Tracker: Multiple Object Tracking in Urban Mixed Traffic Scenes." In Proceedings of the 2019 IEEE International Conference on Signal and Image Processing Applications, ICSIPA, (2019), 335–339. https://doi.org/10.1109/ICSIPA45851.2019.8977783
33. Delavarian, M., and Maarouzi, O. "Vehicle tracking at intersection in image sequences with multilayer concept." In 2017 3rd Iranian Conference on Intelligent Systems and Signal Processing (ICSPIS), (2017), 131–135. https://doi.org/10.1109/ICSPIS.2017.8311603
34. Ciaparrone, G., Luque Sánchez, F., Tabik, S., Troiano, L., Tagliaferri, R., and Herrera, F. "Deep learning in video multi-object tracking: A survey." ***Neurocomputing***, Vol. 381, (2020), 61–88. https://doi.org/10.1016/j.neucom.2019.11.023
35. Jodoin, J. P., Bilodeau, G. A., and Saunier, N. "Urban Tracker: Multiple object tracking in urban mixed traffic." In 2014 IEEE Winter Conference on Applications of Computer Vision, WACV, (2014) 885–892. https://doi.org/10.1109/WACV.2014.6836010
36. Bernardin, K., and Stiefelhagen, R. "Evaluating multiple object tracking performance: The CLEAR MOT metrics." ***Eurasip Journal on Image and Video Processing***, Vol. 2008, No. 1, (2008), 1–10. https://doi.org/10.1155/2008/246309
37. Li, Y., Huang, C., and Nevatia, R. "Learning to associate: HybridBoosted multi-target tracker for crowded scene." In 2009 IEEE Conference on Computer Vision and Pattern Recognition, (2009), 2953–2960. https://doi.org/10.1109/CVPR.2009.5206735
38. Saunier, N., and Sayed, T. "A feature-based tracking algorithm for vehicles in intersections." In Third Canadian Conference on Computer and Robot Vision, CRV (Vol. 2006), (2006), 59-65. https://doi.org/10.1109/CRV.2006.3
39. Mendes, J. C., Gomes Campos Bianchi, A., and Júnior, Á. R. P. "Vehicle Tracking and Origin-destination Counting System for Urban Environment." In Proceedings of the 10th International Conference on Computer Vision Theory and Applications - Volume 3: VISAPP, (2015), 600–607. https://doi.org/10.5220/0005317106000607

Persian Abstract

چکیده

رهگیری تصویری یک موضوع مهم در سیستم‌های حمل و نقل هوشمند است. تقاطع‌ها مکان چالش برانگیزی برای رهگیری در سیستم‌های تصویری هستند، زیرا وسایل نقلیه در جهت‌های متفاوتی همزمان در حرکت هستند. علاوه بر این، در صورت رخداد تصادف جهت‌های ترافیکی نرمال ممکن است با تغییر مواجه شوند. ازدحام، انسداد و جهت حرکت نامعین از جمله چالش‌های مهم رهگیری وسایل نقلیه در تقاطع است. در این مقاله، یک روش برای رهگیری وسایل نقلیه در تقاطع با در نظر گرفتن جریان‌های حرکتی برای غلبه بر جهت‌های حرکت نامعین پیشنهاد شده است. برای این منظور، مدلی چند لایه پیشنهاد شده است که جهت‌های حرکتی را به لایه‌های جداگانه انتساب می‌دهد. علاوه بر این، همسایگی‌های متفاوتی برای هر لایه بر اساس جهت حرکت پیشنهاد شده است. بنابراین وسایل نقلیه‌ای که از یک سمت تقاطع وارد می‌شوند و جهت حرکتی مشابه دارند به لایه یکسان انتساب داده می‌شوند. سپس رهگیری در لایه‌های متفاوت به صورت جداگانه انجام می‌شود. در حالت‌های خاص انسداد، آشکارسازی از دست رفته و عبور وسایل نقلیه از کنار یکدیگر روش رهگیری پیشنهادی بر اساس تاریخچه رهگیری و همسایگی‌ها در هر لایه ادامه ره‌های وسایل نقلیه را پیش‌بینی می‌کند. نتایج نشان می‌دهد که سازگاری مناسبی بین نتایج روش پیشنهادی و حقیقت اصلی وجود دارد و در رهگیری وسایل نقلیه عبوری از تقاطع نسبت به سایر روش‌ها بهتر عمل می‌کند.