Journal of Korea Multimedia Society Vol. 22, No. 2, February 2019(pp. 167-177)
https://doi.org/10.9717/kmms.2019.22.2.167

Real-time Multiple Pedestrians Tracking for Embedded Smart Visual Systems

Van Ngoc Nghia Nguyen[†], Thanh Binh Nguyen[††], Sun-Tae Chung[†††]

ABSTRACT

Even though so much progresses have been achieved in Multiple Object Tracking (MOT), most of reported MOT methods are not still satisfactory for commercial embedded products like Pan-Tilt - Zoom (PTZ) camera. In this paper, we propose a real-time multiple pedestrians tracking method for embedded environments. First, we design a new light weight convolutional neural network(CNN)-based pedestrian detector, which is constructed to detect even small size pedestrians, as well. For further saving of processing time, the designed detector is applied for every other frame, and Kalman filter is employed to predict pedestrians' positions in frames where the designed CNN-based detector is not applied. The pose orientation information is incorporated to enhance object association for tracking pedestrians without further computational cost. Through experiments on Nvidia's embedded computing board, Jetson TX2, it is verified that the designed pedestrian detector detects even small size pedestrians fast and well, compared to many state-of-the-art detectors, and that the proposed tracking method can track pedestrians in real-time and show accuracy performance comparably to performances of many state-of-the-art tracking methods, which do not target for operation in embedded systems.

Key words: Pedestrian Tracking, Object Detection, Deep Learning, Object Association

1. INTRODUCTION

Object tracking such as pedestrian tracking is an important task in the field of computer vision applications such as visual surveillance, traffic monitoring, pedestrian computer interactions, robot navigation, autonomous vehicle driving, biology and so on [1]. Many excellently performing object tracking methods are more concerned with tracking accuracy. Most of such methods are still computationally expensive for embedded systems.

In this paper, we propose a real-time multiple pedestrians tracking method with a good tracking accuracy for our developing PTZ camera with Jetson TX2 [2] as a main processor.

The proposed multiple pedestrians tracking method is based on Tracking-By-Detection (TBD) [3,4,5,6]. TBD based object tracking works in 2 main stages. First, object detection and next, association among detected objects between two consecutive frames.

For the detection part of the proposed tracking method, we design a light weight CNN-based pedestrian detector with a good detection performance of even small size pedestrians which produces pedestrian locations as bounding boxes and pose orientations of detected pedestrians. Since the lightly designed CNN-based pedestrian detector is not computationally light enough for embedded processing, we apply the detector and Kalman fil-

※ Corresponding Author : Sun- Tae Chung, Address: (06978) 369 Sangdo-ro, Dongjak-gu, Seoul, Korea, TEL : +82-2-820-0638, FAX : +82-, E-mail : cst@ssu.ac.kr
Receipt date : Jan. 11, 2019
Approval date : Feb. 11, 2019

[†] Dept. of Information and Telecommunication Eng., Graduate School, Soongsil University
(E-mail : nvnnghia@soongsil.ac.kr)[†]
[††] CublickDigital, Inc., Seoul
(E-mail : binh.nguyen@sectic.com)
[†††] Dept. of Intelligent Systems, Graduate School, Soongsil University

ter-based object predictor, alternatively. The possible prediction error (position and size of the object) in a frame will not accumulated since it is readjusted in the next frame by the CNN-based pedestrian detector, which is more reliable than the Kalman filter-based prediction. For association, we boost the accuracy of Hungarian algorithm [7] by incorporating pose orientation information together with Intersection Over Union (IOU) into the association metrics, which does not increase association processing speed further. The moving direction information, which is adopted in LKDeep [6] is determined not to be adopted in association metrics since moving direction under PTZ environments takes time to compute exactly.

Through experiments on Nvidia's embedded computing board, Jetson TX2 and comparisons to performance results of the sate-of-the-art object detectors and trackers [3,4,5,6], it is shown that the designed pedestrian detector detects fast and performs well even for small size pedestrian detection compared to many state-of-the-art object detectors, and that the proposed tracking method can operate in real-time for embedded systems like PTZ camera equipped with Jetson TX2 as a main processor and perform comparably to performances of many state-of-the-art tracking methods.

2. RELATED WORKS

The recent successfully object tracking methods are based on TBD due to significant improvements in object detection. In TBD, an object detector is first applied to find the target object bounding boxes, then an association rule is applied to associate the newly detected target objects in the current frame with the detected targets in the previous frame.

Recent object detection includes 2 main development trends, 2 stage detector such as Faster R-CNN [8], and single shot detector such as YOLOv2 [9] and SSD [10]. For the real-time processing purpose, single shot detector outperforms 2 stage detectors. On the other hand, single shot detectors perform worse in detecting small objects. Many proposed object detectors [11,12,13] including the third version of YOLO [14] (YOLOv3 [11]) improves the detection performance of small objects by employing several techniques or adopting new features. For example, YOLOv3 improves accuracy on small objects by designing a new backbone - darknet53 and making detection at 3 scales. All of those improved object detectors should pay more computational power for better detection performance. Thus, those improved object detectors are not suitable for real-time embedded applications. Tiny YOLOv3 is a faster version of YOLOv3, however its detection performance of small objects is sacrificed for faster processing. In this paper, we design a new CNN-based pedestrian detector which improves the detection performance of small size pedestrian without further more processing burden.

In TBD-based tracking, a conventional way to solve the association problem is to use Hungarian algorithm. Most of the previous TBD based methods using Hungarian for association merely depended on distance or the overlap ratio. Many of those methods such as SORT[3] and IOU tracker[5] are only based on bounding box position but such association methods usually confuse when objects are overlapped, which leads to very much identity switches in crowd scenes. To alleviate occlusion problems, [6] employs a moving direction information more for Hungarian association. Together with the development of Deep Learning, the authors in Deep-SORT [4] introduce deep association metric. that uses not only the location but also the appearance features. Similarly, [15] introduces a network to match a pair of object detections and use the similarity score for data association. However, calculating the appearance features and similarity score between objects are expensive and therefore may not be appropriate for

the embedded environment.

In this paper, we propose a new simple association algorithm employing pose orientation in addition to IOU for Hungarian association, which boosts the Hungarian association accuracy without sacrificing processing speed.

Pose orientation means where a person looks towards; front, back, right, and left. By incorporating this useful pose orientation information, the proposed association significantly reduces identity switches problem.

3. PROPOSED MULTIPLE PEDESTRIANS TRACKING METHOD

3.1 Overall Working Architecture of the proposed multiple pedestrians tracking method

Fig. 1 shows the overall work-flow of the proposed multiple pedestrians tracking method. The proposed tracking method operates alternatively between detection mode and prediction mode. In detection mode, first detect pedestrians in the current frame and associate them with pedestrians in the previous frame. In the prediction mode, predicting is tracking. The proposed tracking method starts in the detection mode. In the detection mode,

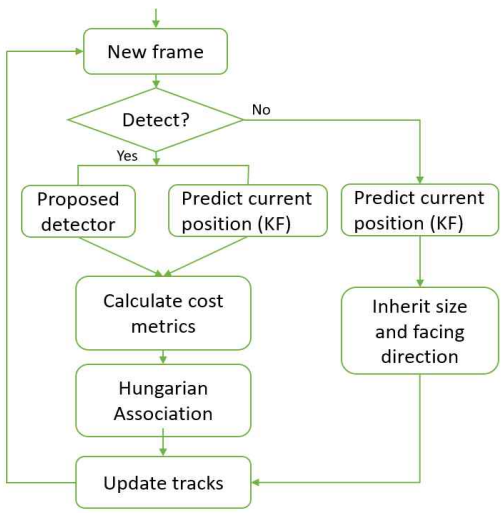

Fig. 1. The flowchart of the proposed multiple pedestrians tracking system.

the pedestrians and their pose orientations are extracted as tight bounding boxes of pedestrians and as one of 4 status (front, back, left, right) by our developed CNN-based pedestrian detector. pose orientation of a pedestrian represents the direction where pedestrian looks towards in a scene. For tracking, association between pedestrians of the previous frame and those of current frame is accomplished by Hungarian algorithm where association cost metrics is made from IOU between the predicted bounding box and detected bound box of a pedestrian pose orientation. Association algorithm of the proposed tracking method is explained in detail later. The predicted bounding box of a pedestrian is obtained from translating the previous bounding box to a new position by a motion vector, which is predicted by Kalman filter. Kalman filter contributes in improving object tracking performance as demonstrated in [16].

In the prediction mode, the tracking system applies Kalman filter to predict the motion vector of each bounding box towards the current frame. Motion vector indicates moving direction. If the system cannot predict the current positions, then it keeps the previous positions as the current positions. And, during prediction mode, size of bounding boxes and pose orientation are kept as the same as before. In the prediction mode, the predicted pedestrians are considered as tracked pedestrians.

The proposed tracking system keeps each pedestrian's information during consecutive 10 frames. After 10 frames, if the tracking system cannot find the corresponding updated locations, it considers the pedestrian is lost or disappear from the scene.

A newly detected pedestrian is kept to follow in next 3 frames. If the tracking system finds a match to it in that period, it is considered a new object and tracking of it starts. Our detector made error sometimes, if a newly detected person is instantly considered a new track.

Fig. 2. PTZ camera setup and detection result.

3.2 Pedestrian detector

We design a CNN-based pedestrian detector which can detect small pedestrians as well as large pedestrians fast enough for real-time embedded applications. As a backbone of the proposed pedestrian detector network, we employ the first 23 layers of Mobilenet [17]. Mobilenet is well-known as a light weight deep neural network for mobile and embedded vision applications, which are based on a streamlined architecture that uses depthwise separable convolutions. We stack 5 more convolution layers to the top of the backbone to get more semantic information. Determination of the additional 5 convolution layers is done after optimality testing through experiments. After stacking 5 more convolution layers, we construct the rest of network into 2 branches; branch A responsible for detecting small size pedestrian and branch B responsible for detection of large size pedestrian. Moreover, inspired by Feature Pyramid Networks [18] that extract features from different layers of different scales, we design the detection network to use features from different layers for accurate detection.

For smaller size pedestrian detection, we need to detect at higher resolution, therefore, we use a convolution layer together with an upsampling layer to increase the resolution of the high-level structure and then we concatenate it with low-level structures features from 11th layer of backbone. We stack 7 more convolution layers to reduce the aliasing effect of upsampling and concatenating. Experiments show that 7 more convolution layers is optimal for branch A. In this branch our image features include low-level structures from 11th layer and high-level structure after upsampling layer. Even though, low-level structures that are not effective for accurate pedestrian detection, it keeps the features of small pedestrian. By combining both low-level and high-level structures features, the feature maps in this branch can describe small pedestrian features with rich semantic information. Therefore, branch A is effective for detecting small size pedestrian.

Fig. 3 shows detail of our network architecture. The output of the Branch B for large size pedestrian detection is a matrix of size 20×20×18. 20x20 refers to the number of grid cells. Each cell contains 18 values, which divides into 2 bounding boxes. Thus, each bounding box has 9 values consisting of 4 bounding coordinates, 4 pose ori-

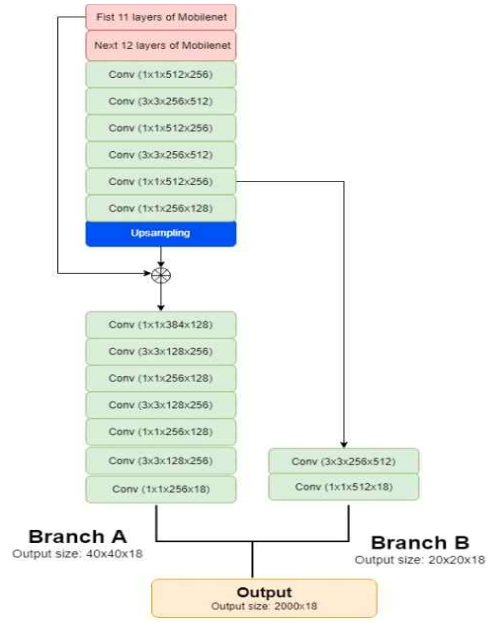

Fig. 3. Pedestrian Detector Network architecture.

entations and 1 for confidence scores. Similarly, in the Branch A, the output size is 40×40×18. Branch A is used to detect small pedestrians, and therefore, we use higher resolution feature maps and increase the number of cells. The final output size is the combination of 2 branches with 2000 candidate bounding boxes (40×40+20×20 = 2000). Each bounding box is responsible to predict 18 values. Therefore, our final output size is 2000×18. Similarly to YOLOv3, we use the anchors to make predicting people. We use K-means clustering on our training set to find suitable anchor values for our dataset. The anchors we use in our experiments are (5×24), (9×50), (17×80), (32×155), (58×251), (114×402).

Fig. 4 shows detection result of 2 branches on MOT16-12 sequence video, at frame 215. The result of branch A is on the right, it only detects the small pedestrian while branch B focus on detecting large pedestrian.

The loss function of the designed pedestrian detector network inherits from YOLO[14], The loss function penalizes four loss criteria. The detector network produces 2000 candidate bounding boxes. Each box is responsible for predicting 18 values, and then the detector network calculates loss for each cell. We obtain the final loss by summing up losses of every cells. For each box, the loss includes bounding box loss, pedestrian confidence loss, background loss and pose orientation loss.

Bounding box loss is the same as in YOLO[14] as shown in 1st and 2nd term in formula (2). Pedestrian confidence loss indicates the probability that a box contains a pedestrian (3rd term in formula (2)). Background loss is used to penalize when the background is detected as a pedestrian (4th term in formula (2)). In formula (2), $(x_{ij}, y_{ij}, w_{ij}, h_{ij}, c_{ij})$ is the center location, width, height and confidence score of jth box in grid cell i, $(\widehat{x_{ij}}, \widehat{y_{ij}}, \widehat{w_{ij}}, \widehat{h_{ij}}, \widehat{c_{ij}})$ is the predicted center location, width, height and confidence score of the corresponding box. 1_{ij}^{obj} indicates pedestrian, which is equal to 1 if the jth bounding box of cell i contains a pedestrian and 0 in other case. 1_{ij}^{nobj} indicates background, $1_{ij}^{nobj} = 1$ only if the bounding box jth of cell i doesn't contain a pedestrian.

We add one more term to penalize human pose orientation. In this paper, we consider 4 main directions, facing to the left, right, front and back. The adopted direction loss is based on cross entropy loss which is expressed as shown in formula (1) where p_{ijk} is the probability that a pedestrian in the jth bounding box of a grid cell i faced to kth direction. $\widehat{p_{ijk}}$ is the corresponding predicted probability.

$$\frac{1}{N}\sum_{k=1}^{N} 1_{ij}^{obj}\left[\, p_{ijk}\log\left(\widehat{p_{ijk}}\right)+(1-p_{ijk})\log\left(1-\widehat{p_{ijk}}\right)\right] \qquad (1)$$

Some constants α, λ, γ are weighting factors of each loss function. For example, if we want more precision on the location, we set a higher value for λ.

Fig. 4. Detection result of 2 branches on MOT16-12 sequence video, at frame 215.

$$
\begin{aligned}
Loss &= \lambda_{coord}\sum_{i=1}^{2000}\sum_{j=1}^{B}\mathbf{1}_{ij}^{obj}[(x_i-\hat{x}_i)^2+(y_i-\hat{y}_i)^2] \\
&+\lambda_{coord}\sum_{i=1}^{2000}\sum_{j=1}^{B}\mathbf{1}_{ij}^{obj}[(\sqrt{w_i}-\sqrt{\hat{w}_i})^2+(\sqrt{h_i}-\sqrt{\hat{h}_i})^2] \\
&+\alpha\sum_{i=1}^{2000}\sum_{j=1}^{B}\mathbf{1}_{ij}^{obj}[C_i\log(\hat{C}_i)+(1-C_i)\log(1-\hat{C}_i)] \\
&+\gamma\sum_{i=1}^{2000}\sum_{j=1}^{B}\frac{1}{N}\sum_{k=1}^{N}\mathbf{1}_{ij}^{nobj}[-(p_{ijk}\log(\hat{p_{ijk}})+(1-p_{ijk})\log(1-\hat{p_{ijk}})]
\end{aligned}
\tag{2}
$$

where $\mathbf{1}_{i}^{obj}$ denotes if object appears in cell i and $\mathbf{1}_{ij}^{obj}$ denotes that the j^{th} bounding box predictor in cell i is responsible for that prediction.

We trained our network on our collected dataset for 70 epochs. The dataset contains roundly 6000 images from a part of MOT16 [20] training set and from the internet. We need to relabel the dataset because there is no suitable dataset that provide pose orientation information.

3.3 Tracking Association

We propose an association algorithm that utilizes features of the association cost metrics of SORT [3] and Deep-SORT[4]. We first integrate IOU, pose orientation all together to form a cost metrics. Then we apply Hungarian association method on that cost metrics. Fig. 5 shows the integration.

Calculating Cost metric

Input: List of predicted tracking pedestrian objects $T = \{t_1, t_2, \ldots t_N\}$
List of detecting pedestrian objects: $D = \{d_1, d_2, \ldots d_M\}$
Output: Cost metrics $C = \{c_{i,j} | i = 1 \rightarrow N, j = 1 \rightarrow M\}$

Calculating:

For i **in** N:
  For j **in** M:
    $c_{i,j} = IOU(t_i, d_j)$
    If $c_{i,j} > 0$:
      If $fd(t_i) == fd(t_j)$:
        $c_{i,j} = \min(1, c_{i,j} += 0.1)$
      Else:
        $c_{i,j} = \max(0, c_{i,j} -= 0.05)$

Fig. 5. Calculating cost metric.

where $fd(t_i)$ and $fd(d_i)$ is pose orientation of predicted tracking object t_i and pose orientation of detecting object d_i.

The IOU between predicted tracking object t_i and detecting object d_i is calculated as bellow:

$$IOU(t_i, d_j) = \frac{Area(t_i) \cap Area(d_j)}{Area(t_i) \cup Area(d_j)} \tag{3}$$

4. EXPERIMENTS

4.1 Experimental environments

For the evaluation of the proposed pedestrian detector, we employ the well-known pedestrian dataset - INRIA person dataset [19] which includes roundly 1000 images. Furthermore, in order to evaluate the efficiency of the proposed pedestrian detector's detectability of small size pedestrians, we used MOT16 [20] pedestrian detection test set which includes 7 challenge videos with total 5919 image frames, 759 tracks and 182,326 target bounding boxes. Video frame rate vary from 14 frame per second to 30 frame per second, frame size is 640×480 (MOT16-06 sequence) and 1920×1080 (the remaining sequences). Pedestrian density varies from 8.1 to 45.3 pedestrian per frame.

For evaluation of the proposed multiple pedestrians tracking method, we utilize MOT16 benchmark [20]. And compare the proposed tracking method with other the state-of-the-art tracking methods, SORT[3], Deep-SORT[4], IOU-Tracker [5] which have been reported in MOT 2016 Benchmarks [20]. MOT16 evaluation dataset include 7 challenge videos (MOT16-01, MOT16-03, MOT16-06, MOT16-07, MOT16-08, MOT16-12, MOT-16-14). We compare proposed tracking method with other state-of-the-art tracking methods using proposed detector on 2D MOT 2015 benchmark[20]. The evaluation dataset includes 11 sequence videos (TUD-Stadtmitte, TUD-Campus, PETS09-S2L1, ETH-Sunnyday, ADL-Rundle-6, ADL-Rundle-8, ETH-Bahnhof, ETH-Pedcross2,

KITTI-13, KITTI-17, Venice-2) with total 5500 image frames, 39905 target bounding boxes. Video frame rate vary from 7 to 30 frames per second, frame size from 640×480 to 1920×1080. The pedestrian density varies from 2.2 to 11.9 pedestrian per frames.

We use a PC with Intel™ Core™ i7-4770 CPU @ 3.40GHz, 16GB of RAM with GeForce GTX Titan X Graphic Card to train the proposed pedestrian detector. The embedded system for experiments is a Jetson TX2 board [2] with HMP Dual Denver 2/2 MB L2 + Quad ARM® A57/2 MB L2, CPU @ 2GHz, 8GB of RAM. NVIDIA Pascal™, 256 CUDA cores.

4.2 Experimental metrics

To evaluate pedestrian detection performance, we use Average Precision (AP), precision and recall. For the detail, reader may refer to [21].

For evaluating tracking performance, we use performance metrics suggested by MOT bench [20] that include MOTA, MOTP, MT, ML, FP, FN, ID SW, Frag. If the tracked object is an actual target object, then the tracked object is called true positive, and if not, then the tracked object is called false positive. If the actual target is missed to track, then it is called false negative. FP represents the total number of false positives, and FN represents the total number of false negatives. ID SW (Identity Switch) counts the number of mismatched objects in a frame. If a tracked object does not match its actual ground truth target, it is said that its ID is switched. MT (Mostly Tracked targets) means the ratio of ground-truth trajectories that are covered by a track hypothesis for at least 80% of their respective life span. ML (Mostly Lost targets) is the ratio of ground-truth trajectories that are covered by a track hypothesis for at most 20% of their respective life span. Frag is the total number of times a trajectory is fragmented (i.e. interrupted during tracking), and Hz is processing speed (in frames per second excluding the detector) on the benchmark.

MOTA (Multiple Object Tracking Accuracy) measures three error sources: false positives, missed targets and identity switches and is defined as:

$$MOTA := 1 - \frac{\sum_t (FN_t + FP_t + IDSw_t)}{\sum_t (GT_t)} \tag{4}$$

where t is the frame index and GT is the number of ground truth objects. MOTP (Multiple Object Tracking Precision) measure the misalignment between the annotated (ground truth) and the tracked bounding boxes and is defined as:

$$MOTP := \frac{\sum_{t,i} d_{t,i}}{\sum_t C_t} \tag{5}$$

where C_t denotes the number of matches in frame time t and $d_{t,i}$ is the IOU (3) between tracked object i and its ground truth object at frame time t. MOTP thereby gives the average overlap between all correctly matched hypotheses and their respective objects and ranges between t_d = 50% and 100%.

For the detail description of each metric, readers may refer to [6, 20]. For object tracking point of view, MOTA is considered to be more important than the MOTP.

4.3 Experimental results

4.3.1 Performance of Pedestrian detector

Experiment results about performance of the proposed pedestrian (object) detector against INRIA and MOT16 dataset are summarized in Table 1.

YOLOv3 JTA is the YOLOv3 trained on JTA dataset, YOLOv2 is the version 2 of YOLO, and DPM is Deformable Part Model [23]. The results for the proposed one and Tiny YOLOv3 in Table 1 have been obtained from experiments on Jetson TX2 board and AP of YOLO JTA, YOLOv2, Faster R-CNN, and DPM against MOT16 are taken from MOT website [20], but speed is measured on Jetson TX2 board.

Table 1 shows that the proposed pedestrian de-

Table 1. Performance comparison between state-of-the-art pedestrian detector and proposed pedestrian detector

	AP against INRIA	AP against MOT16	MOT16 Precision	MOT16 Recall	Speed(FPS) (On Jetson TX2)
Proposed Pedestrian Detector	0.66	0.49	55.5	56.4	10 FPS
Tiny YOLOv3	0.51	0.27	32.9	35.4	14 FPS
YOLOv3 JTA [20]	-	0.62	84.7	66.2	1 FPS
YOLOV2[20]	-	0.46	84.3	69.9	3 FPS
Faster R-CNN[20]	-	0.72	89.8	77.3	10 times slower than YOLOv2[*]
DPM[20]	-	0.61	64.8	68.1	-

(*) In [9], the author claimed that YOLOv2 is about 10 times faster than Faster R-CNN.

tector performs significantly better than Tiny YOLOv3 against MOT16 testing set, which contain many small size pedestrians. YOLOv3 performs more reliable in pedestrian detection over the proposed one. However, YOLOv3 is very slow on an embedded computing device, Jetson TX2 board. It takes around 1 second to process a frame. For the proposed tracking method, we apply the detector every other frame, which makes the entire processing of the proposed tracking method operate in real-time even for embedded systems.

4.3.2 Performance of The Proposed Multiple Pedestrian Tracker

Experimental results on performance of the proposed tracker against MOT16 benchmark are summarized in Table 2. We compare our result with some open source state-of-the-art tracking algorithms such as SORT[3], Deep-SORT[4]. and IOU tracker[5], and our previous tracker, LKDeep [6], SORT[3], Deep-SORT[4], and IOU tracker[5] are not the best among the stat-of-art tracking algorithms, but are chosen since they released open source codes.

The results of LKDeep in Table 2 is taken from [6]. The results of other open source trackers in Table 2 is taken from [20], and the arrow ↑ indicates the higher score is better, and arrow ↓ indicates the lower score is better. IOU tracker, SORT, and Deep-SORT utilizes Faster R-CNN detection data provided by MOT16 challenge, which produces more precise detection than single shot

Table 2. Comparing tracking results on MOT16 challenge Benchmarks

	MOTA ↑	MOTP ↑	MT ↑	ML ↓	FP ↓	FN ↓	ID SW ↓	Frag ↓	Association speed ↑	Computing Environemnt
Proposed Tracker(**)	41.58	74.0	10.7	42.4	7569	97738	1345	1566	148	Jetson TX2
Proposed Tracker v2(*)	42.3	74.2	11.7	41.2	7590	96425	1207	2533	84.6	Jetson TX2
SORT[3] (*)	59.8	79.6	25.4	22.7	8698	63245	1423	1835	59.5	2.6 GHz, 1 Core
Deep-SORT [4](*)	61.4	79.1	32.8	18.2	12852	56668	781	642	17.4	2.6 GHz, 1 Core
IOU Tracker [5](*)	57.1	77.1	23.6	32.9	5702	70278	2167	1839	3004	3.4 GHz, 1 Core
LKDeep[6](*)	32.3	76.4	5.7	62.1	1193	121333	953	943	32	3.4GHz, CPU

detector like the proposed detector, but is slow. LKDeep, which was our previous tracker, utilizes DPM detection data provided by MOT16. And, Proposed Tracker v2 is the same as proposed tracker except that pedestrian detector is applied for every frame. In Table2, (*) indicates that a detector in the tracker is applied for every frame, and (**) indicates that a detector is applied for every other frame.

From experimental results in Table 2, one can notice the following facts.

First, the proposed tracker achieves comparable performance overall compared to our previous tracker, but with significant association speed-up, which makes the proposed tracker more suitable for embedded application. Second, the proposed tracker shows much better association speed compared to other open source tracker except IOU tracker. Even though IOU tracker shows very high association speed due to very simple association algorithm, it cannot work in real-time since it utilizes Faster R-CNN detector which is slow as shown in Table 1.

In order to have a fair comparison, we evaluated the open source tracking methods, SORT tracker, Deep-SORT tracker and IOU tracker by replacing theirs detectors by our designed pedestrian detector against 2D MOT 2015 benchmark. Table 3 summarizes the experimental results from replacement of our pedestrian detector, which are produced by Multiple Object Tracking Challenge Development Kit [22].

As stated in 4.2, from tracking point of view, MOTA (Multiple Object Tracking Accuracy) is the most important performance metrics to evaluate a tracking method. Proposed tracker v2 where pedestrian detector is appled for every frame produces higher MOTA and MOTP compared to other trackers. Table 3 shows that by incorporating pose orientation into association cost metrics, the proposed tracker has less ID SW(Identity Switches) of 431 compared to 528 of SORT while keeping almost the same processing time. Our association also can keep tracking the target for longer period with the number MT is 129 compared to 111 of SORT. IOU tracker do a simple and less accuracy association, so that leads to a lot of ID SW(Identity Switches) problem with 1117 identity switches. With high quality deep appearance features more for association, Deep-SORT is the best at keep tracking target, with ID SW of 285. It also keeps tracking the target for long period with the highest MT 165. However, extracting deep appearance features takes a lot of time which makes Deep-SORT becomes the slowest tracker in Table 3.

The proposed tracker reduces processing time by applying pedestrian detection every other frame. From Table 3, one can see that by applying detection for every other frame we made a big improvement with respect to processing speed without sacrificing too much tracking accuracy. The proposed tracker is the fastest among trackers in

Table 3. Comparing tracking results on the same detector

	MOTA ↑	MOTP ↑	MT ↑	ML ↓	FP ↓	FN ↓	ID SW ↓	Frag ↓	FPS ↑
Proposed Tracker (**)	37.1	73.8	129	163	8044	16575	496	1018	14.5
Proposed Tracker v2 (*)	39.1	74.7	134	152	7588	16268	431	1000	9.2
SORT (*)	38.2	74.4	111	156	7043	17077	528	847	9.3
SORT v2 (**)	36.4	73.9	109	182	7200	17608	576	869	14.5
Deep-SORT(*)	38.1	74.4	165	139	9342	15065	285	851	4.2
IOU Tracker(*)	36.1	74.6	174	122	9588	14776	1117	1130	10

((*) ; Detector is applied for every frame, (**) ; Detector is applied for every other frame.)

Table 3. We also tested SORT v2 where our designed detector is applied for every other frame. Accuracy performance of SORT v2 deteriorates even though speed performance improves. Overall, the proposed tracker can be considered as more suitable than other trackers listed in Table for embedded applications such as multiple pedestrians tracking under PTZ camera.

5. CONCLUSION

In this paper, we proposed a real-time multiple pedestrians tracking method for embedded applications such as embedded surveillance. Compared to the state-of-the-art multiple objects tracking methods, which show excellent tracking accuracy performance, the proposed tracker traded off accuracy performance against speed performance, but operates in real-time and performs accurately good enough for some embedded applications, which is shown through comparison experiments on Jestson TX2 embedded board.

REFERENCE

[1] H. Yanga, L. Shaoa, F. Zhenga, L. Wangd, and Z. Songa, "Recent Advances and Trends in Visual Tracking: A Review," *Journal of Neurocomputing*, Vol. 74, No. 18, pp. 3823-3831, 2011.

[2] Jetson TX2, https://elinux.org/Jetson_TX2 (accessed Feb., 12, 2019).

[3] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple Online and Realtime Tracking," *Proceeding of IEEE International Conference on Image Processing*, pp. 3464-3468, 2016.

[4] N. Wojke, A. Bewley, and D. Paulus, "Simple Online and Realtime Tracking with a Deep Association Metric," *Proceeding of IEEE International Conference on Image Processing*, pp. 3645-3649, 2017.

[5] E. Bochinski, V. Eiselein, and T. Sikora. "High-Speed Tracking-by-Detection Without Using Image Information," *Proceeding of International Workshop on Traffic and Street Surveillance for Safety and Security at IEEE Advanced Video and Signal-based Sureillance*, pp. 1-8, 2017.

[6] Q.D. Vu, T.B. Nguyen, and S.T. Chung, "Simple Online Multiple Pedestrian Tracking Based on LK Feature Tracker and Detection for Embedded Surveillance," *Journal of Korea Multimedia Society,* Vol. 20, No. 6, pp. 893-910, 2017.

[7] Hungarian Algorithm, https://en.wikipedia.org/wiki/Hungarian_algorithm (accessed Feb., 12, 2019).

[8] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-time Object Detection with Region Proposal Networks," *Proceedings of the 28th International Conference on Neural Information Processing Systems* - Vol. 1, pp. 91-99, 2015.

[9] J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," *Proceeding of 2017 IEEE Conference on Computer Vision and Pattern Recognition,* pp. 6517-6525, 2017.

[10] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.Y. Fu, et al., "SSD: Single Shot MultiBox Detector," *Proceeding of European Conference on Computer Vision,* pp. 1-17, 2016.

[11] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," *arXiv:1804.02767,* 2018.

[12] S. Bell, C.L. Zitnick, K. Bala, and R. Girshick. "Inside-outside Net: Detecting Objects in Context with Skip Pooling and Recurrent Neural Networks," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,* pp. 2874-2883, 2016.

[13] J. Li, X. Liang, Y. Wei, T. Xu, J. Feng, S. Yan, et al., "Perceptual Generative Adversarial Networks for Small Object Detection," *Proceeding of 2017 IEEE Conference on Computer Vision and Pattern Recognition,* pp. 1222-1230, 2017.

[14] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once Unified, Real-time Object Detection," *Proceeding of 2016 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 779-788, 2016.

[15] L. Leal-Taix, C.C. Ferrer, and K. Schindler, "Learning by Tracking: Siamese CNN for Robust Target Association," *Proceeding of 2016 IEEE Computer Vision and Pattern Recognition Conference Workshops*, pp. 418-425, 2016.

[16] D. Kim, J. Park, and C. Lee "Object-tracking System Using Combination of CAMshift and Kalman Filter Algorithm," *Journal of Korea Multimedia Society,* Vol. 16, No. 5, pp. 619-628, 2013.

[17] A.G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, et al., "Mobile Nets: Efficient Convolutional Neural Networks for Mobile Vision Applications," *arXiv: 1704.04861,* 2017.

[18] T.Y Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, S. Belongie, et al., "Feature Pyramid Networks for Object Detection," *Proceeding of 2017 IEEE Conference on Computer Vision and Pattern Recognition,* pp. 936-944, 2017.

[19] INRIA Person Dataset, http://pascal.inrialpes.fr/data/pedestrian/ (accessed Feb., 12, 2019).

[20] MOT Challenge, https://motchallenge.net/ (accessed Feb., 12, 2019).

[21] M. Everingham, L. Gool, C.K. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes (VOC) Challenge," *Journal of Computer Vision.* Vol. 88, Issue 2, pp. 303-338, 2010.

[22] Multiple Object Tracking Challenge Development Kit, https://bitbucket.org/amilan/motchallenge-devkit/ (accessed Feb., 12, 2019).

[23] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan, "Object Detection with Discriminatively Trained Part Based Models," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol. 32, No. 9, pp. 1627-1645, 2010.

Van Ngoc Nghia Nguyen

He received the B. Eng. Degree in Electronics & Telecommunications Engineering from The University of Danang - University of Science and Technology, Danang, Vietnam, in 2017. He is currently a research assistant at Embedded Real-time Computing Laboratory, Soongsil University, Seoul, South Korea. His main areas of research interests include machine learning, deep learning, smart embedded systems, image processing, visual surveillance and recognition systems.

Thanh Binh Nguyen

He received the B. Eng. Degree in computer science from the University of Science, Ho Chi Minh, Vietnam, in 2005, the M. Sc. degree in information and telecommunication engineering from the University of SoongSil, Seoul, South Korea, in 2010, and the Ph.D. degree in engineering at the University of Soongsil, Seoul, South Korea, in 2017. He is currently a principal software R&D researcher at CublickDigital, Inc., Seoul, South Korea. South Korea. His research interests cover the design and analysis of various smart embedded software system, IoT and also intelligent imaging systems, video analytic algorithms which are applied to visual surveillance, recognition systems, and etc.

Sun-Tae Chung

He received B.E. degree from Seoul National University, and M.S. degree and Ph.D. degree in Electrical Eng. and Computer Science from the University of Michigan, Ann Arbor, USA, in 1986 and 1990, respectively.

Since 1991, he had been with the School of Electronic Eng. at the Soongsil university, Seoul, Korea where he is now a full professor. Now, he has been with the Dept. of Smart Systems Software, at the Soongsil Univ. since 2015. His research interests include: computer vision, visual surveillance, digital signage systems, and digital marketing.