Journal of Korea Multimedia Society Vol. 20, No. 6, June 2017(pp. 893-910)
https://doi.org/10.9717/kmms.2017.20.6.893

Simple Online Multiple Human Tracking based on LK Feature Tracker and Detection for Embedded Surveillance

Quang Dao Vu[†], Thanh Binh Nguyen[††], Sun-Tae Chung[†††]

ABSTRACT

In this paper, we propose a simple online multiple object (human) tracking method, LKDeep (Lucas-Kanade feature and Detection based Simple Online Multiple Object Tracker), which can run in fast online enough on CPU core only with acceptable tracking performance for embedded surveillance purpose. The proposed LKDeep is a pragmatic hybrid approach which tracks multiple objects (humans) mainly based on LK features but is compensated by detection on periodic times or on necessity times. Compared to other state-of-the-art multiple object tracking methods based on 'Tracking-By-Detection (TBD)' approach, the proposed LKDeep is faster since it does not have to detect object on every frame and it utilizes simple association rule, but it shows a good object tracking performance. Through experiments in comparison with other multiple object tracking (MOT) methods using the public DPM detector among online state-of-the-art MOT methods reported in MOT challenge [1], it is shown that the proposed simple online MOT method, LKDeep runs faster but with good tracking performance for surveillance purpose. It is further observed through single object tracking (SOT) visual tracker benchmark experiment [2] that LKDeep with an optimized deep learning detector can run in online fast with comparable tracking performance to other state-of-the-art SOT methods.

Key words: Object Tracking; LK Feature Tracker; Deep Learning Object Detection; Occlusion Handling.

1. INTRODUCTION

Given some object of interest marked in one frame of a video, the goal of single object tracking (SOT) is to locate this object in subsequent video frames, despite object motion, changes in viewpoint, lighting changes, size and shape changes, other variations, and occlusion. Tracking multiple objects in a scene becomes more difficult since it should not only locate objects in frames but also maintain object identities despite of object occlusion, object missing/recovering, and object entrance/exiting in the scene.

Since Object tracking is an important task within the field of computer vision applications such as visual surveillance, traffic monitoring, human-computer interactions, robot navigation, autonomous vehicle driving, biology and so on [3], it has attracted intensive attention from researchers during the last three decades [4, 5, 6, 7], and benchmarking programs for SOT [6, 8] and MOT [1] performance testing environments have been reporting testing results about tracking performances of the state-of-the-art SOT or MOT methods.

Conventional detection free matching-based

※ Corresponding Author : Sun-Tae Chung, Address: (06978) Dept. of Smart Systems Software, Soongsil Univ., 369, Sangdo-Ro, Dongjak-Gu, Seoul, Korea, TEL : +82-2-820-0638, FAX : +82-2-821-7653, E-mail : cst@ssu.ac.kr
Receipt date : Feb 15, 2017, Revision date : April. 28, 2017

[†] Dept. of Information and Telecommunication Engineering, Soongsil University
(E-mail : azkan@bka.vn)
[††] Embedded Vision, Inc., Seoul, South Korea
(E-mail : binh.nguyen@ieev.org)
[†††] Dept. of Smart Systems Software, Soongsil Uniersity

tracking methods such as Kalman appearance tracker [9, 10], Lucas-Kanade (LK) feature tracker [11], Particle filter tracker [12], and Color Mean Shift-based tracker [10, 13] could not deal with lighting changes, size and shape changes, other variations, and occlusion properly, which usually occur in real environments. Recent works on object tracking based on correlation filter show excellent tracking performance in SOT [14], but successful application of correlation filter-based object tracking methods to MOT, especially under crowded scenes has not been reported yet [15].

On the other hand, in detection based tracking, called 'Tracking-By-Detection', objects are at first localized in each frame and then via object hypotheses, localized objects are associated and linked into trajectories. In these TBD-based object tracking methods, the performance of tracking accuracy is heavily affected by the detection and association results. Due to recent progress in reliable object detections like deep learning-based object detections, TBD approach has become the leading paradigm in MOT as well as in SOT. In order to resolve ambiguities in associating detected objects and to overcome possible detection failures or inaccuracies, many of recent works [16-22] process video sequences in a batch mode in which video frames from future time steps are also utilized to solve the data association problem. However, such non-causal systems are not suitable for online tracking applications like real-time visual surveillance monitoring, robot navigation and autonomous driving. Most of the state-of-the art online MOT methods reported in MOT challenge [1] utilize or develop complicated association mechanism using various feature cues. However, these excellently performing online MOT methods based on complicated association mechanism in addition to reliable detection by deep learning are computationally expensive and not appropriate for embedded systems like IP network cameras, PTZ cameras, and so on, which either do not have GPU or have less powerful GPU.

Usually, surveillance application does not require strict accuracy in object detection and tracking much since intrusion detection and overall behavior analysis of objects are main concerns for surveillance purpose. Thus, in visual surveillance where real-time automatic monitoring is desirable, real-time or fast processing becomes more important than in other application areas. Also, according to recent study [23], at higher frame rates, simple trackers such as correlation filters outperform complex methods based on deep networks. Thus, exquisite association and complicated detection mechanism with highly reliable but expensive computation do not always guarantee better tracking performance under every environment.

In this paper, we explore a pragmatic hybrid approach to online multiple object (human) tracking method for applications where real-time or fast monitoring is desirable as in embedded surveillance. Unlike TBD approach where tracking is accomplished by applying object detection for every frame and linking the detected objects at every frame via association, the proposed multiple human tracking method, called LKDeep, basically tracks human objects via a LK feature tracker but with compensation by a reliable object detection at necessary frame times. Lighting changes, size and shape changes, other variations, and occlusion during tracking may cause the LK feature tracker to predict objects poorly or not reliably at all. Unreliable prediction may cause object missing and poor prediction may incur accumulation of tracking errors. LKDeep reduces accumulated tracking errors by readjusting tracking human object bounding boxes periodically. Predicting object bounding boxes in every 4th frame are readjusted via association with detected object bounding boxes in the frame. The association rule utilizes two simple metrics; 1) distance between the predicting bounding box and detected object bounding boxes and 2) motion direction information. Information for

calculating two metrics is obtained from LK feature tracker and object detection. The feature cues utilized in LKDeep such as predicting bounding box, the number of LK feature points, and LK motion vector obtained by applying LK feature tracker are simpler and less heavy computationally than other exquisite but complicated feature cues utilized in other state-of-the-art MOTs' association rules.

LKDeep detects object missing or object occlusion by checking the number of LK feature points and detected objects. When occlusion or object missing is detected, then LKDeep tries to recover the occluded or missing (human) objects in consequent frames by associating prediction of occluded or missing objects with newly detected objects. The association also handles effectively object entrance/exit and reappearance.

Since LKDeep does not have to detect objects at every frame, it can save object detection processing time which is not small if reliable but computationally demanding object detection like deep leaning-based object detection is employed. Also, LKDeep does not use extra exquisite feature cues for association except LK feature points and LK optical flow extracted during LK tracking and detected objects' positions. LKDeep is pragmatic in the sense that it does not require complicate feature cues but adopts a simple rule for association so that implementation is simple and fast online, which is desirable for real-time surveillance application. For further computational saving, LKDeep employs pyramidal processing [25] for speedy search for corresponding LK feature points, and FAST corner point detector [26, 27] for the extraction of feature points. Fast corner point detector is reliable but faster to extract than Shi-Tomasi's Good Features to Track used in KLT (Kanade-Lucas-Tomasi) tracker [28].

The organization of the paper is as follows. Section 2 explains related work and Section 3 describes the proposed multiple object (human) tracking method. Experimental results are given in Section 4 and final conclusion follows in Section 5.

2. RELATED WORK

The aim of an object tracker is to generate the trajectory of an object over time by locating its position in every frame of the video. Object tracker may also provide the complete region in the image that is occupied by the object at every time instant. In many conventional detection-free object trackers such as Kalman appearance tracker [9], LK feature tracker [11], Particle filter tracker [12], and Color Mean Shift-based tracker [13], objects are tracked via prediction of the object region and correspondence by using information obtained from previous frames. In the case of LK feature tracker, it predicts a set of feature points across the video frames and tracks objects by estimating object regions based on predicted feature points. LK tracker makes use of spatial intensity information to direct the search for the position that yields the best match. LK tracker, especially the LK(Lucas-Kanade) tracker with pyramids [25], is faster than traditional techniques for examining far fewer potential matches between the images. LK tracking usually considers two adjacent frames at a time, and ignores nearly all the feature information of previous frames. Furthermore, the LK features of the tracking objects are affected by illumination change, object shape or size change and environments change, and thus LK tracker alone cannot handle variations of illuminations, object shapes, object sizes effectively even if it may handle occlusion [30]. Handling such variations can be helped by object detection. Particle filter is the name given to the sample-based (i.e., Monte Carlo) approximation of the Bayes recursion. The particle filtering-based tracking approach turns out to work well, but it is computationally heavy and moreover it suffers from occlusion, large deformation of the

target objects and illumination changes that result in the large difference between the current observations and the appearance model. Computationally, a deterministic approach like color Mean Shift [13] is light and has been proven successful. Mean Shift is a kernel based gradient descent procedure that finds the local maxima/minima of a function. Even though Color Mean Shift is a very efficient tracker, but it is also unable to cope with occlusions. Recently correlation filter-based object tracking methods [14] have been proposed. Correlation filter-based object trackers model the appearance of objects using filters trained on example images. The target is initially selected based on a small tracking window centered on the object in the first frame. From this point on, tracking and filter training work together. The target is tracked by correlating the filter over a search window in next frame; the location corresponding to the maximum value in the correlation output indicates the new position of the target. An online update is then performed based on that new location. In addition to the incorporation of multi-channel features, the correlation filter framework has been significantly improved lately by including scale estimation, non-linear kernels, a long-term memory, and by alleviating the periodic effects of circular convolution. These improvements show excellent tracking performance in SOT. However, successful application of correlation filter-based object tracking methods to MOT especially under crowded scenes has not been reported yet [15].

As opposed to SOT, MOT additionally requires maintaining the identities among multiple objects. Besides the common challenges in both SOT and MOT, the further key issues in making MOT challenging include (but not limit to): 1) frequent occlusions, 2) initialization and termination of tracks, 3) small size of objects, 4) similar appearance among objects, and 5) interaction among multiple objects. In order to deal with the MOT problem, a wide range of solutions have been proposed in recent years.

As in SOT, due to recent progress in reliable object detection, TBD-based approach starting from [32] has become the leading paradigm in MOT. There, the major challenge is how to associate noisy object detections in the current video frame with previously tracked objects. The basis for any data association algorithm is a similarity function between object detections and targets. To handle ambiguities in association, it is useful to combine different cues in computing the similarity, such as appearance, motion, location, and/or other feature vectors either by manually extracted feature vectors like LK feature points or deep learning trained feature vectors [18]. The performance of tracking accuracy is heavily affected by the association results.

Object trajectories can be found in a global optimization problem that processes entire video batches at once. More mathematical modeling approaches such as network flow formulations [16], continuous energy minimization [17] and probabilistic graphical models [18] have become popular frameworks of this type. However, batch processing required for these methods are not applicable in online scenarios where a target identity must be available at each time step.

More traditional methods are Multiple Hypothesis Tracking (MHT) [19] and the Joint Probabilistic Data Association Filter (JPDAF) [20]. These methods perform data association on a frame-by-frame basis. In the JPDAF, a single state hypothesis is generated by weighting individual measurements by their association likelihoods. In MHT, all possible hypotheses are tracked, but pruning schemes must be applied for computational tractability. MHT_DAM [21] and JPDA_m [22], which are revisited versions of JPDAF and MHT respectively under TBD scenario have shown promising results. However, the performance of these methods comes at increased computational and implementation complexity so that they have to be

processed in batch mode.

In [1], benchmark results about the state-of-the-art MOT methods are reported. Among them, we survey and describe top ranked online fast MOT methods with associated papers since we are interested in online fast tracking, not batch mode tracking in this paper.

'Simple Online and Real-time Tracking (SORT)' [33] is a pragmatic approach to MOT by considering MOT to be handled with main focus in associating objects efficiently for online and real-time applications. In pragmatic spirit, SORT is similar to the proposed LKDeep. SORT simply applies Kalman filtering [34] to predict object positions in the next frame and achieves object tracking by associating the detected object with the predicting objects based on IOU metric which measures overlap between predicting bound boxes and detected bounding boxes. Kalman filtering-based tracking is less reliable but faster than LK feature tracker since it predicts one center point or a few more points for each object while LK feature tracker needs to predict many feature points for each object. SORT compensates weak reliability of Kalman filtering by a very reliable but computationally heavy Faster Region CNN (FrRCNN) detection framework [35]. Bounding box overlap alone cannot handle occlusion effectively. DeepSORT [36] overcomes the weakness of SORT with respect to occlusion by replacing the association metric with a more informed metric that combines motion and appearance information learned by convolutional neural network (CNN) so that it increases robustness against misses and occlusions and is able to track objects through longer periods of occlusions by reducing the number of identity switches effectively. However, more feature cues increase computational cost.

Person in Interest (POI) [37] detects objects utilizing Faster-RCNN detector [35], and implements a simple online tracker, which uses Kalman filter [34] for motion prediction and Kuhn-Munkres Hungarian algorithm [38] for data association with Reid feature from QAN [39]. EAMTT [40] proposes an online multi-target tracker that exploits both high- and low-confidence target detections in a Probability Hypothesis Density Particle Filter framework. High-confidence (strong) detections are used for label propagation and target initialization. Low-confidence (weak) detections only support the propagation of labels, i.e. tracking existing targets. Moreover, it performs data association just after the prediction stage, and thus avoiding the need for computationally expensive labeling procedures such as clustering. MDPNN [41] presents an online method that encodes long-term temporal dependencies across multiple cues. In order to address the challenge to accurately track occluded targets or those which share similar appearance properties with surrounding objects, it presents a structure of Recurrent Neural Networks (RNN) that jointly reasons on multiple cues over a temporal window. CDA_DDAL [42] solves the online multi-object tracking problem by associating tracklets and detections in different ways according to their confidence values. For more reliable association between tracklets and detections, it also proposes a deep appearance learning method to learn a discriminative appearance model from large training datasets, since the conventional appearance learning methods do not provide rich representation that can distinguish multiple objects with large appearance variations. oICF [43] combines an appearance model based on Integral Channel Features with a simple motion model based on Kalman filter in order to estimate change in position and to smooth the trajectory. OVBT [44] proposed an online variational Bayesian model for multiple-person tracking. The computational efficiency of OVBT is due to closed-form expressions for both the posterior distributions of the latent variables and for the estimation of the model parameters. GMPHD_HDA [45] handles the false detection by revising Gaussian mixture probability

hypothesis density (GM-PHD) filter and handles miss detection by a hierarchical tracking framework to associate fragmented or ID switched tracklets.

3. THE PROPOSED MULTIPLE HUMAN TRACKING METHOD

3.1 Overall Working Architecture

Fig. 1 shows the overall workflow of the proposed online multiple object (human) tracking method, LKDeep. The proposed method tracks humans in a scene by LK feature tracker in cooperation with a object (human) detector like DPM [29] (Version 5) (DPM_V5). As mentioned in Introduction, for experiments with respect to SOT, YLSSOD (YOLO Simplified Single Object Class Detector) is developed and utilized for single object detection. Normally, LK feature tracker tracks humans (or object) by prediction based on feature points and their optical flow motion, but when tracked objects turn out to be missing or occluded or at every 4th frame, a detector (DPM_V5 or YLSSOD) is applied to resolve missing or occlusion or to alleviate the accumulated tracking error. When predicted bounding boxes are replaced by detected bounding boxes, new feature points are extracted from the detected bounding boxes and LK tracker starts to predict human object bounding boxes based on new feature points.

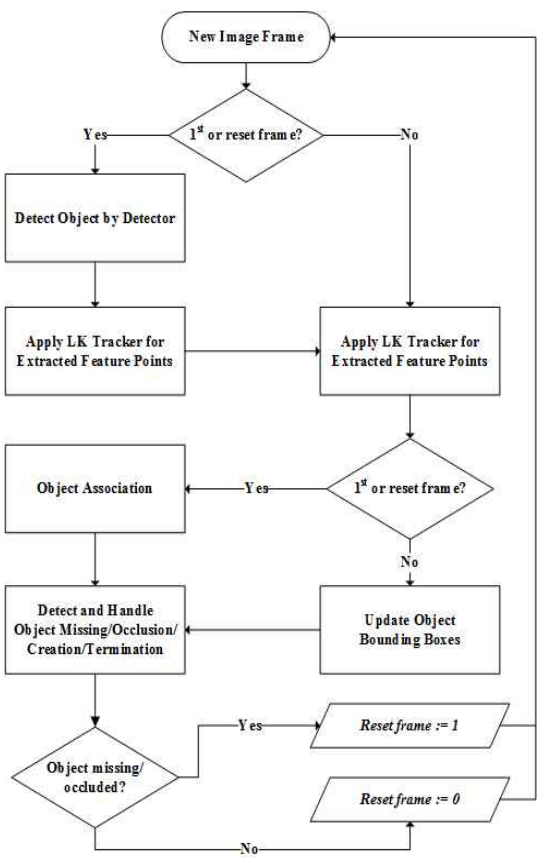

Fig. 1. The workflow of the proposed tracking method.

A) The Beginning Phase

In the beginning of the proposed tracking method, human objects (or a single object to be tracked) are first detected by a detector. The bounding box of a human (or other) object detected by the detector is used to define a region of interest (ROI) for feature extraction. LK tracker first extracts feature points in each ROI by using FAST algorithm [26, 27]. These feature points will then be used for tracking purpose in the following frame. Note that we make an assumption about object entrance (creation)/exit (termination) later in Section 3.3, but the assumption is not applied in the beginning phase.

B) LK Tracking Phase

In LK tracking phase (left hand flow in Fig. 1), when the new frame (current frame) comes in, LK tracker predicts a feature point in the current (new) frame which corresponds to each feature point in the previous frame by searching an area around the feature point in the previous frame based on Forward-Backward error algorithm [24]. In prediction, only if correlation between two points is higher than a threshold, then the feature point in the current frame is determined as the corresponding feature point. If the number of predicted feature points of an object is over a threshold (in this pa-

per, half of the number of feature points belonging to an object in the previous frame), the object represented by the feature points are considered as in tracked status. If not, the object is considered as missing object. If an object is in the tracked status, then the corresponding bounding box is predicted as translation of the object bounding box in the previous frame by a motion vector (Fig. 2(a)). The translational motion vector is calculated as an average of the all translational motion vectors between corresponding pair of feature points (Red dotted arrow in Fig. 2(a)). The object bounding box size is kept the same. The size readjustment of object bounding box is reflected in object detection phase. When the corresponding object bounding box is predicted, LK tracker considers the predicted bounding box and feature points as reliable, and continues to track based on these until the next object detection time. If the corresponding object bounding box cannot be predicted, LK tracker considers the object represented by a bounding box in the previous frame as missing (Fig. 2 (b)). If an object is in missing status, then the motion vector is predicted from the previous motion vector

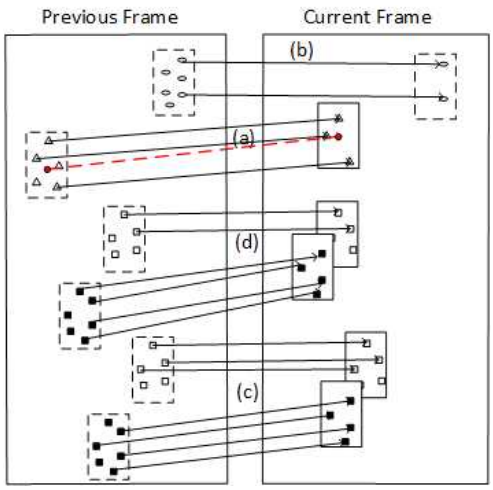

Fig. 2. Cases in LK tracking phase.
(a) object bounding box is well predited, (b) object is missing, (c) objects are overlapped and the occluded object is not missing (d) objects are overlapped and the occluded object is missng

and previous velocity of the object, and is kept for resolution of object missing in the future frames.

From the predicted object bounding boxes, one can easily check whether object bounding boxes are overlapped. Object occlusion can be classified into three categories: self-occlusion, inter-object occlusion, and occlusion by the background scene structure. Self-occlusion occurs when one part of the object occludes another. This situation most frequently arises while tracking articulated objects. Inter-object occlusion occurs when two objects being tracked occlude each other. Inter-object occlusion can be easily detected by checking overlapping among object bounding boxes. Self-occlusion and occlusion by background scene structure are hard to check. Object missing is determined by checking the number of predicted feature points. Inability of predicting feature points is caused by occlusion or noisy environments.

If predicting bounding boxes are overlapped, then the proposed tracking method considers that occlusion happens (Fig. 2 (c) and (d)). When two bounding boxes A and B are overlapped, then determining which is occluder and which one is occludee is not important for tracking purpose. If an object has enough number of predicting feature points, then the proposed LKDeep method can continue to track (Fig. 2(c)). If not, LKDeep considers the object is missing (Fig. 2(d)) and will keep watching to recover it.

C) Tracking Readjustment Phase

The proposed tracking method, LKDeep is supposed to readjust the object bounding boxes to track by bounding boxes detected by the detector applied at every 4th frame. Association of the newly detected bounding boxes with the predicting bounding boxes is done by calculating the shortest distance among any bounding box pairs around the predicting bounding box, and by checking motion vector direction, which is explained in more detail in Section 3.3.

D) Missing Object/New Object Handling Phase

When a tracking object becomes missing, the proposed method keeps predicting the position of the missing object in the next frames, based on the last tracked speed and direction of the object before missing happens and assumes the same bounding box size until the next detection phase. If missing or occlusion happens, a detector will be applied at the next frame.

A detector like DPM_V5 and YLSSOD performs well but it costs computational time so that applying it as less often as possible is desirable for real-time processing unless it is necessarily required. However, it is observed that applying a detector parsimoniously deteriorates tracking performance of the proposed method seriously.

After object association of the newly detected bounding boxes with the predicting bounding boxes of objects in tracked status is processed, the remaining detected bounding boxes (if any) are considered as potential candidates for missing objects or new objects or false positive results. The nearest association process is applied on remaining bounding boxes. If a remaining detected bounding box has the shortest distance to the latest predicting bounding box position of a missing object among the remaining detected bounding boxes and the distance is less than a pre-defined threshold, then the remaining detected bounding box is determined as the bounding box for the missing object. After that association process is over, the bounding boxes still left are considered as new objects if and only if they satisfy the assumption about object entrance/exit (mentioned later in Section 3.3). Otherwise they are considered as false positive detection results and will be discarded. If the number of detected objects are less than that of predicting objects, the remaining unassociated predicting objects are considered as missing but still reliable to be kept to track. The missing object handling phase can last no longer than over 10 consecutive frames. After 10 frames, if the proposed method cannot still find the missing objects, then these missing objects are marked as lost objects and they are not tracked anymore.

E) Tracking Speed

Common TBD approach first detects object bounding boxes in every frames and associates detected bounding boxes with previous objects. Reliable deep learning object detectors like YOLOv2 [31] require pretty long processing time to complete. Then, tracking method based on such object detector is hard to achieve real-time processing in embedded systems. In our proposed tracking method, with an aim for applying to embedded system, we develop YLSSOD (YOLO Simplified Single Object Class Detector), which is the optimized version of Tiny YOLOv2, so that it works reasonably fast with acceptable accuracy on CPU only. Secondly, we construct the proposed tracking method so as to apply YLSSOD at every 4th frame or when it is needed. Then, LK tracking algorithm will do the rest. From profiling PC program implementing the working flow of the proposed method, it is observed that the time consumed by the optimized YLSSOD is nearly 4 times longer than the total time consumed by LK tracking algorithm. On the other hand, the LK tracking of objects detected by YLSSOD during the next consecutive 3 frames is observed to be acceptable as long as object occlusion or object missing does not occur.

3.2 YOLO Single Object Class Detector (YLSSOD)

YOLOv2 object detector [31] is well known for faster multiple object detection processing speed compared to other state-of-the-art object detection methods. However, even Tiny YOLOv2, which is a reduced version of YOLOv2 for further faster processing, is still not fast enough on CPU only so that it may not run in real-time in embedded systems which are usually either not equipped with GPU accelerators or equipped with

less powerful GPUs.

For our purpose, object detector has to detect only one class (human), unlike Tiny YOLOv2 trained to detect 20 object classes. Thus, we simplify and optimize Tiny YOLOv2 more so that it can run in real-time on CPU only. Our developed human or one object class detector consists of 8 convolutional layers with less number of filters, 4 max pooling layers followed by an anchor boxes and output detection layer. Although the human (or one object class) detector developed for the proposed human tracking method, YLSSOD, leads to reduction of detection accuracy (increased false alarm rate and missing alarm rate), it does not affect the tracking performance of the proposed tracking method, LKDeep, much since the developed YLSSOD performs satisfactory and applying it at every 4th frame or on necessary events can have opportunity to readjust alleviate accumulated tracking errors due to imprecise object detection.

3.3 Object Tracking, Association, Missing, Creation and Termination

When the first frame comes in, human objects (or other class object) are detected initially as bounding boxes by a human (object) detector (DPM_V5 or YLSSOD). Feature points inside each bounding box are then extracted based on FAST algorithm [26, 27].

In the proposed object tracking method, object i at time t, o_t^i is identified by:

$$o_t^i = (p_t^i, TB_t^i, v_t^i, \alpha_t^i, F_t^i, S_t^i) \tag{1}$$

where p_t^i represents its center location (x_c^i, y_c^i) of the tracking bounding box TB_t^i which is represented by (x^i, y^i, w^i, h^i) where (x^i, y^i), w^i and h^i are the coordinates of the left upper point, width and height of the bounding box, respectively. v_t^i and α_t^i represent the object velocity and moving direction angle, respectively. F_t^i is the set of feature points that lie inside the object bounding box (or object ROI) TB_t^i. S_t^i is the variable denoting the current state of the object: tracked, missing, and lost.

At the beginning when t = 0 or at restarting time, v_t^i and α_t^i are all set to 0.

The set $O_t = \{o_t^i | i = 1...M-1\}$ is the set of all objects at time t. M is the maximum number of objects from the beginning until t.

During tracking, LK tracker predicts feature points in the new image frame which corresponds to feature points in the previous image frame. Specifically, given a feature point $u = (u_x, u_y)$ in the previous frame, LK tracker predicts its corresponding point $v = u + d = (u_x + d_x, u_y + d_y)$ in the current frame using the pyramidal implementation of the LK feature tracking [25]. Forward-Backward (FB) error checking [24] is adopted to make the prediction result more reliable. By applying LK tracker in the backward direction (from v to u), the new prediction u′ is obtained. The error is defined by the Euclidean distance between the initial point u and the newly obtained u′: $E(u) = \| u - u' \|$. Then, we determine the predicted feature point by LK tracker to be reliable when $E(u) < threshold$ (5 pixels in this paper). Otherwise the predicted feature point is considered as not reliable and thus as missing.

After corresponding feature points are predicted for the image frame at time t, and if the number of the predicting feature points are over a threshold, then v_t^i, α_t^i are calculated from F_t^i and F_{t-1}^i as follows:

$$v_t^i = \frac{\sum_{j=1}^{n} (u_t^j - u_{t-1}^j)}{n} \tag{2}$$

$$\alpha_t^i = \tan^{-1} \frac{(v_t^i)_y}{(v_t^i)_x} \tag{3}$$

where the number of predicted feature points in F_t^i is n, each feature point is indexed by $u_t^j (j = 1..n)$. If not, $v_t^i = v_{t-1}^i$, and $\alpha_t^i = \alpha_{t-1}^i$.

The predicting bounding box of object i at time

t, PB_t^i is calculated by $PB_t^i = TB_{t-1}^i + v_t^i$. During LK tracking times, the tracking bounding box will be set to the predicting bounding box PB_t^i, that is, $TB_t^i = PB_t^i$. If the number of success predicted feature points is over the threshold (in this paper, half of the number of feature points belonging to an object in the previous frame), then S_t^i is set to 'tracked'. If not, S_t^i is set to 'missing'. If missing continues over consecutive 10 frames then $TB_t^i = null$ and S_t^i is set to 'lost'. During detection times, the tracking bounding box is supposed to be corrected by a detected bounding box associated with the predicting bounding box.

During detection times, the association is achieved by utilizing 1) distance between the predicting bounding box and detected object bounding boxes, and 2) motion direction information, as follows.

Let us denote the number of the detected object bounding at time t as l, a detected bounding box at time t as $DB_t^k (k = 1..l)$. Also let us denote the Euclidean distance between the centers of predicting bounding box PB_t^i and a detected bounding box DB_t^k as $Dist(PB_t^i, DB_t^k)$. The moving direction of an object bounding box is said to be in the similar direction to another object bounding box if their intersection angle is less than 90^0.

Now, the pseudo-code for association rule is listed as below. That is, DB_t^k is associated with object j^*.

From the camera parameters, we can find the scale of current field of view. Thus, to simplify searching, only nearby objects within some distance are considered for the computation (4).

If $PB_t^{j^*}$ is a predicting bounding box of the missing object bounding box status at time t-1 , then the missing object is recovered by $TB_t^{j^*}$. If there are any remaining unassociated predicting bounding boxes, then the corresponding objects are considered as missing at time t.

If also there exists a detected bounding box which is not associated with a predicting bounding box, then the detected bounding box object is either a new object or a false positive. The unassociated detected bounding box can happen when the number of objects detected in the current frame is larger than that of the predicting bounding boxes. But it can also happen even when the number of detected objects are equal or less than that of the predicting objects because the adopted object detector can detect false positive objects.

In this paper, we make a weak assumption that there is no door or no big human objects or any tracking object classes (larger than 50% of the field of view (FOV)) appearing inside the FOV. Thus, human or any tracking object classes can appear or disappear only when they enter into or get out of the FOV. Therefore, the unassociated detected objects are determined to be new objects if they appear near the four borders. Otherwise they will be considered as false positive objects. Fig. 3

Association Rule

for $DB_t^k (k = 1..l)$ **do**
  $j^* = \arg min_j (Dist(DB_t^k, PB_t^j))$ (4)
  if $Dist(DB_t^k, PB_t^j) < threshold$ (5)
  and the moving direction of $PB_t^{j^*}$ is in the similar direction to DB_t^k **then** (6)
    $TB_t^{j^*} = DB_t^k$
    $S_t^i = 0$
  end if
end for

Fig. 3. YLSSOD's result, ground truth and tracking result at frame 86 of PET'09 S2.L1.
(a) Ground truth (b) YLSSOD detection result (c) Tracking result of the proposed method

shows how the proposed method handles the false positive result from YLSSOD. Green boxes in Fig. 3(a) represent ground-truth data, red boxes in Fig. 3(b) means YLSSOD detection results, and yellow box in Fig. 3(b) indicates the false positive entry. Fig. 3(c) shows the result of handling the false positive by the proposed tracking method.

4. EXPERIMENTAL RESULTS

4.1 Experimental Environment

We tested the proposed method with respect to MOT metrics, and we also tested it for single object tracking environment since the proposed tracking method can be also utilized for single object tracking, which is more realistically pursued in PTZ camera-based visual surveillance.

For evaluation for online multiple human tracking, we utilized MOT16 benchmark [7] to compare the proposed LKDeep with the other online state-of-the-art MOT methods listed in MOT16 results in [1]. MOT16 provides 7 testing video sequences; MOT16-1, MOT16-3, MOT16-6, MOT16-7, MOT16-8, MOT16-12, MOT16-14. For details about these testing video sequences including their visual scenes, one may refer to [1].

For evaluation for single object tracking, we utilized object tracking benchmark in [6] to compare with the other state-of-the art visual tracker listed in Visual Tracker Benchmark [2, 8]. Visual Tracker Benchmark [8] provides TB-50 video data set consisting of 50 video sequences. In this paper, since we want to test whether LKDeep can track a designated single object but also handle occlusion effectively in SOT, we chose TB-50 video sequences with occlusion in [8], which consists of 28 video sequences among TB-50 's 50 video sequences.

The workstation used for experiments is a PC with Intel® CoreTM i7-4770 CPU @ 3.40GHz, 16GB of RAM with GeForce GTX Titan X Graphic Card. For MOT testing as well as SOT testing, we used CPU only. GTX Titan X is utilized for training YLSSOD which is used as detector for SOT.

4.2 Evaluation Methodology

In [6], for quantitative evaluation of single object tracking, the precision plot and success rate plot are suggested. One widely used evaluation metric on tracking precision is the center location error, which is defined as the average Euclidean distance between the center locations of the tracked targets and the manually labeled ground truths. Then the average center location error over all the frames of one sequence is used to summarize the overall performance for that sequence. Precision plot proposed in SOT performance benchmark [6] shows the percentage of frames whose estimated location is within the given threshold distance of the ground truth. The reader should notice that the 'precision' concept in precision plot is different from 'precision' definition as in PASCAL measure [46] in object detection. As the representative precision score for each tracker, [6] uses the score for the threshold = 20 pixels. Another metric is about accuracy which is evaluated utilizing IOU between

the tracking bounding box, B_{TR} of an object and the ground truth bounding box B_{GT} of the object,

$$IOU := \frac{area(B_{TR} \cap B_{gt})}{area(B_{TR} \cup B_{gt})} \quad (6)$$

Then, to measure the accuracy performance on a sequence of frames, one can count the number of successful frames whose IOU is larger than the given threshold Th_o. The success plot shows the ratios of successful frames at the threshold Th_o varied from 0 to 1. As representative accuracy score for each tracker, one can use the area under curve (AUC) of each success plot to rank the tracking algorithms. The conventional way to evaluate trackers under SOT environments is to run them throughout a test sequence with initialization from the ground truth position in the first frame and report the precision plot and success rate plot. [6] refers this as one-pass evaluation (OPE). However, a tracker may be sensitive to the initialization, and its performance with different initialization at a different start frame may become much worse or better. Therefore, [6] proposes two other metrics to analyze a tracker's robustness to initialization, by perturbing the initialization temporally (i.e., start at different frames) and spatially (i.e., start by different bounding boxes). These tests are referred as temporal robustness evaluation (TRE) and spatial robustness evaluation (SRE) respectively.

For multiple object tracking, MOT challenge [1, 7] suggests CLEAR metrics proposed by [47]; MOTA, MOTP, FAF, MT, ML, FP, FN, IDSW, Frag, and Hz. 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. FAF means the average number of false positives per frame. IDSW (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:

$$MOTA := 1 - \frac{\sum_t (FN_t + FP_t + IDSw_t)}{\sum_t (GT_t)} \quad (7)$$

where t is the frame index and GT is the number of ground truth objects. [1, 7] report the percentage MOTA $(-\infty; 100]$ in MOT benchmark. Note that MOTA can also be negative in cases where the number of errors made by the tracker exceeds the number of all objects in the scene.

MOTP (Multiple Object Tracking Precision) measure the misalignment between the annotated (ground truth) and the tracked bounding boxes and is defined as:

$$MOTP := \frac{\sum_{t,i} d_{t,i}}{\sum_t C_t} \quad (8)$$

where C_t denotes the number of matches in frame time t and $d_{t,i}$ is the IOU (6) 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%.

4.3 Experimental Results for MOT environments

Table 1 shows experimental results of the proposed method with the public human object detector (DPM_V5) in comparison with other state-

Table 1. Comparisons to other online state-of-the-art MOT mothods in [1]

Online MOT Method/Detector	MOTA ↑	MOTP ↑	FAF ↓	MT(%) ↑	ML(%) ↓	FP ↓	FN ↓	ID Sw ↓	Frag ↓	FPS ↑	Computing Environment
POI/FrRCNN	66.1	79.5	0.9	34	20.8	5.061	55.914	805	3,093	9.9	i7-4790 (3.6GHz) and GTX970
DeepSORT_2/ FrRCNN	61.4	79.1	2.2	32.8	18.2	12,852	56,688	781	2,008	17.4	2.6 GHz CPU and Quadro M 6000 GPU.
SORTwHPD16/ FrRCNN	59.8	79.6	1.5	25.4	22.7	8,698	63,245	1,423	1,835	59.5	2.6 GHz CPU
EAMTT/Particle filter-based	52.5	78.8	0.7	19.0	34.9	4,407	81,223	910	1,321	12.2	i7 3.40GHz
MDPNN16/Public	47.2	75.8	0.5	14.0	41.6	2,681	92,856	774	1,675	1.0	TITAN X, 3GHz CPU
CDA_DDALv2/ Public	43.9	74.7	1.1	10.7	44.4	6,450	95,176	676	1.795	0.5	3.1GHz CPU
oICF_16/Public	43.2	74.3	1.1	11.3	48.5	6,651	96,515	381	1,404	0.4	2.30GHz CPU
OVBT/Public	38.4	75.4	1.9	7.5	47.3	11,517	99,463	1,321	2,140	0.3	2.53GHz CPU
GMPHD_HDA/ Public	30.5	75.4	0.9	4.6	59.7	5,169	120,970	539	731	13.6	3.5 GHz CPU
LKDeep/Public	**32.3**	**76.4**	**0.2**	**5.7**	**62.1**	**1,193**	**121,333**	**953**	**943**	**32**	**3.4 GHz, CPU**

of-the-art online MOT methods listed in MOT16 Results [48] of [1] against MOT16 testing 7 video sequences. The overview of these sequences in the MOT16 benchmark is explained in detail in [7], and can be download from [1]. The tracking performance data of other online MOT methods in Table 1 are taken from [1]. The arrow ↑ shown in some performance metrics in Table 1 indicates that higher measuring score is better, and ↓ indicates the other case. What each MOT method means is well documented in [1].

In Table 1, MOT methods in the above white rows operate with private detectors like FrRCNN. MOT challenge [1] calls its provided human object detector, DPM_V5 as the public detector, and call other detectors as private detectors. Private detectors adopted in online MOT methods in Table 1 are known to detect objects better than DPM_V5.

For the experiments in Table 1, we wanted to investigate how the proposed simple online MOT method can handle tracking by the simple association and handle object missing/occlusion. Thus, experimental results in Table 1 are mainly concerned with comparison with other online public MOT methods (in the below non-white rows). The proposed MOT method, with respect to MOTA, MOTP and other MOT performance metrics, shows worse MOT tracking performance than those of MDPNN16 [41], CDA_DDALv2 [42], oICF_16 [43], and OVBT [44] except GMPHD_HDA [45]. However, the proposed LKDeep shows significantly better processing speed compared to most of other state-of-the-art MOT methods listed in Table 1. Compared to other online MOT methods, MOTA of LKDeep is lower, which is mainly due to larger number of FN and IDSW. Definitely, FN and IDSW are affected by detecting performance of the adopted detector. That is why online private MOT methods Table 1 show better tracking performance, MOTA than those based on the public detector. In this experiment, the proposed MOT method, adopts a conservative DPM detection policy by choosing only positive confidence scoring bound boxes as reliable human object bounding

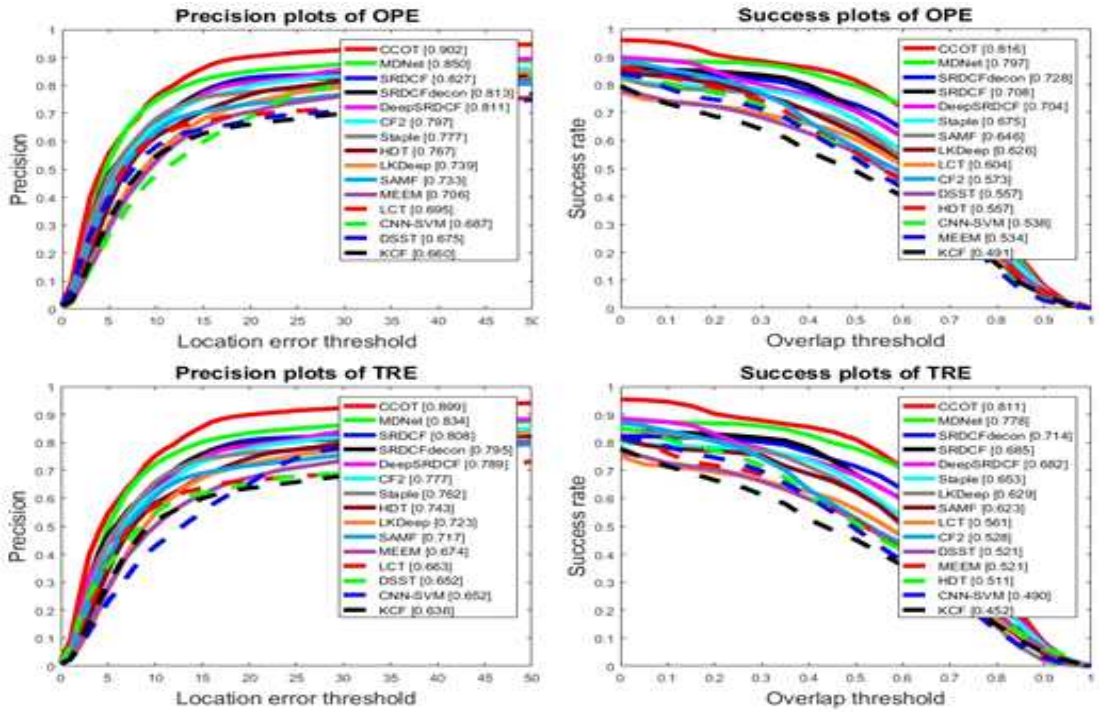

Fig. 4. Plots of OPE and TRE on TB-50 dataset.
The representative score for each tracker is shown in the legend

boxes. By tuning out optimal confidence score threshold, FN and IDSW will be made to have larger decreasing rates than the increasing rate of FP. Then, MOTA of LKDeep is expected to show better tracking performance.

4.4 Experimental Results for SOT environments

In PTZ camera environments, single object tracking is usually more demanding than multiple object tracking since PTZ camera cannot track multiple objects simultaneously. In this experiment, we wanted to test how much the proposed tracking method, LKDeep with a single object detector, YLSSOD can effectively track single object against TB-50 video sequences with occlusion. TB-50 dataset consists of difficult 50 video sequences among 100 video sequences in TB-100 dataset. Contents of TB-50 and TB-100 dataset can be seen in [8].

Fig. 4 shows the experimental results of OPE/TRE precision plot and success plot of the proposed LKDeep for single object tracking on TB-50 dataset in comparison with other state-of-the art SOT methods listed in [2]. For the detailed information about each SOT tracker like CCOT(C-COT), MDNet, and other tracker in the graph means, the reader should refer to [2].

Fig. 4 shows the proposed method obtains the precision at 73.9% (rank 9) and success rate at 62.6% (rank 7) in OPE compared to other state-of-the-art algorithms. Meanwhile, since the proposed method has re-initialize mechanism, the precision and success rate in TRE are almost same as OPE, at 72.3% and 62.9% respectively.

Comparison with respect to processing speed of each tracker listed in Fig. 4. cannot be accomplished because processing speed information of them are not publicly available. The proposed LKDeep with YLSSOD detector is observed to process frames over 23 fps for TB-50 video se-

quences on i7-4770 CPU @ 3.40GHz with 16GB of RAM.

5. CONCLUSIONS

In this paper, we proposed a simple online multiple human tracking method which can run fast and reliably on CPU core for embedded surveillance purpose. The proposed multiple human tracker runs faster than most of the state-of-art MOT methods based on 'Tracking-By -Detection(TBD)'. In TBD - based MOT, a reliable but computationally expensive deep learning detector is applied on every frame. On the other hand, the proposed multiple human tracker mainly tracks objects based on less computational LK feature tracker and readjusts tracking objects by a simple association between the LK predicting objects and detected objects periodically or on necessary times.

Through experiments in comparison with other MOT methods using the public DPM detector among online state-of the-art MOT methods reported in MOT challenge [1], it was shown that the proposed online multiple human tracking method runs faster with reasonable tracking performance for surveillance purpose. It was also observed that the proposed object tracker, LKDeep with an optimized deep learning detector can run in online fast with comparable tacking performance to other state-of-the-art methods in SOT visual tracker benchmark [2].

These experimental results imply that a simple association with an optimized reliable detector can be a practical solution to an embedded real-time single object tracking application task with less strict tracking requirement as one under a practical PTZ surveillance camera without a competitive GPU.

REFERENCES

[1] Multiple Object Tracking Benchmark, https://motchallenge.net/ (accessed May, 20, 2017).

[2] Visual Tracker Benchmark Results, https://github.com/foolwood/benchmark_results (accessed May, 20, 2017).

[3] 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.

[4] A. Smeulders, D. Chu, R. Cucchiara, S. Calderena, A. Dehghan, and M. Shah, "Visual Tracking: An Experimental Survey," *Journal of IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol. 36, No. 7, pp. 1442-1468, 2014.

[5] W. Luo, X. Zhao, and T. Kim, "Multiple Object Tracking: A Literature Review," *arXiv Preprint* 1409.7618, 2014.

[6] Y. Wu, J. Lim, and M. Yang, "Object Tracking Benchmark," *Journal of IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol. 37, No. 9, pp. 1834-1848, 2015.

[7] A. Milan, L. Leal-Taix´e, I. Reid, S. Roth, and K. Schindler, "Multiple Object Tracking Benchmark16: A Benchmark for Multi-Object Tracking," *arXiv preprint* 1603.00831, 2016.

[8] Visual Tracker Benchmark, http://www.visual-tracking.net (accessed May, 20, 2017).

[9] H.T. Nguyen and A.W.M. Smeulders, "Fast Occluded Object Tracking by a Robust Appearance Filter," *Journal of IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol. 26, No. 8, pp. 1099-1104, 2004.

[10] D.Y. Kim, J.W. Park, and C.W. Lee, "Object-Tracking System Using Combination of CAMshift and Kalman Filter Algorithm," *Journal of Korea Multimedia Society,* Vol. 6, No. 5, pp. 619-628, 2013.

[11] B. Lucas and T. Kanade, "An Iterative Image Registration Technique with an Application to Stereo Vision," *Proceeding of International Joint Conference on Artificial Intelligence,*

pp. 674-679, 1981.

[12] B. Ristic, S. Arulampalam, and N. Gordon, *Beyond the Kalman Filter: Particle Filters for Tracking Applications*, Artech House, Boston, Massachusetts, USA, 2003.

[13] D. Comaniciu, V. Ramesh, and P. Meer, "Real-Time Tracking of Non-Rigid Objects Using Mean Shift," *Proceeding of IEEE Conference on Computer Vision and Pattern Recognition,* pp. 142-149, 2000.

[14] Z. Chen, Z. Hong, and D. Tao, "An Experimental Survey on Correlation Filter-Based Tracking," *arXiv Preprint* 1509.05520, 2015.

[15] Yang and G. Bilodeau, "Multi-Kernel Correlation Filter for Visual Tracking," *arXiv Preprint* 1611.02364, 2016.

[16] L. Zhang, Y. Li, and R. Nevatia, "Global Data Association for Multi-Object Tracking Using Network Flows," *Proceeding of IEEE Conference on Computer Vision and Pattern Recognition,* pp. 1-8, 2008.

[17] A. Milan, S. Roth, and K. Schindler, "Continuous Energy Minimization for Multitarget Tracking," *Journal of IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol. 36, No. 1, pp. 58-72, 2014.

[18] B. Yang and R. Nevatia, "Multi-Target Tracking by Online Learning of Non-Linear Motion Patterns and Robust Appearance models," *Proceeding of IEEE Conference on Computer Vision and Pattern Recognition,* pp. 1918-1925, 2012.

[19] D.B. Reid, "An Algorithm for Tracking Multiple Targets," *Journal of IEEE On Automatic Control,* Vol. 24, No. 6, pp. 843-854, 1979.

[20] T.E. Fortmann, Y. Bar-Shalom, and M. Scheffe, "Sonar Tracking of Multiple Targets Using Joint Probabilistic Fata Association," *Journal of IEEE Oceanic Engineering Society,* Vol. 8, No. 3, pp. 173-184, 1983.

[21] C. Kim, F. Li, A. Ciptadi, and J.M. Rehg, "Multiple Hypothesis Tracking Revisited," *Proceeding of IEEE International Conference on Computer Vision*, pp. 4696-4704, 2015.

[22] S.H. Rezatofighi, A. Milan, Z. Zhang, Q. Shi, A. Dick, and I. Reid, "Joint Probabilistic Data Association Revisited," *Proceeding of IEEE International Conference on Computer Vision*, pp. 3047-3055, 2015.

[23] H.K. Galoogahi, A. Fagg, C. Huang, D. Ramanan, and S. Lucey, "Need for Speed: A Benchmark for Higher Frame Rate Object Tracking," *arXiv Preprint* 1703.05884, 2017.

[24] Z. Kalal, K. Mikolajczyk, and J. Matas, "Forward-Backward Error: Automatic Detection of Tracking Failures," *Proceeding of International Conference on Pattern Recognition,* pp. 23-26, 2010.

[25] J.Y. Bouguet, "Pyramidal Implementation of the Affine Lucas-Kanade Feature Tracker Description of the Algorithm," *Journal of Intel Corporation,* Vol. 1, No. 2, pp. 1-9, 2001.

[26] E. Rosten and T. Drummond, "Fusing Points and Lines for High Performance Tracking," *Proceeding of IEEE International Conference on Computer Vision,* pp. 1508-1515, 2015.

[27] E. Rosten and T. Drummond, "Machine Learning for High-Speed Corner Detection," *Proceeding of European Conference on Computer Vision,* pp. 430-443, 2006.

[28] J. Shi, and C. Tomasi, "Good Feature To Track," *Proceeding of IEEE Conference on Computer Vision and Pattern Recognition,* pp. 593-600, 1994.

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

[30] C. Zhang, J. Xu, A. Beaugendre, and S. Goto, "A KLT-Based Approach for Occlusion Handling in Human Tracking," *Proceeding of*

Picture Coding Symposium, pp. 337-340, 2012.

[31] J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," *arXiv Preprint* 1612.08242, 2016.

[32] H. Li, Y. Li, and F. Porikli, "DeepTrack: Learning Discriminative Feature Representations Online for Robust Visual Tracking," *arXiv preprint* 1503.00072, 2015.

[33] A. Bewley, G. Zongyuan, F. Ramos, and B. Upcroft, "Simple Online and Real-Time Tracking," *Proceeding of IEEE International Conference on Image Processing,* pp. 3464-3468, 2016.

[34] R. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Journal of Basic Engineering,* Vol. 82, No. Series D, pp. 35-45, 1960.

[35] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *Proceeding of Neural Information Processing Systems,* pp. 91-99, 2015.

[36] N. Wojke, A. Bewley, and D. Paulus, "Simple Online and Real-Time Tracking with a Deep Association Metric," *arXiv Preprint* 1703.07402, 2017.

[37] F. Yu, W. Li, Q. Li, Y. Liu, X. Shi, and J. Yan, "POI: Multiple Object Tracking with High Performance Detection and Appearance Feature," *arXiv*: 1610.06136, 2016.

[38] H.W. Kuhn, "The Hungarian Method for the Assignment Problem," *Journal of Naval Research Logistics Quarterly,* Vol. 2, No. 1, pp. 83-97, 1955.

[39] Y. Liu, J. Yan, and W. Ouyang, "Quality Aware Network for Set to Set Recognition," *arXiv preprint* 1704.03373, 2017.

[40] R. Sanchez-Matilla, F. Poiesi, and A. Cavallaro, "Online Multi-Target Tracking with Strong and Weak Detections," *Proceeding of Benchmarking Multi-target Tracking, European Conference on Computer Vision,* pp. 84-99, 2016.

[41] A. Sadeghian, A. Alahi, and S. Savarese, "Tracking The Untrackable: Learning To Track Multiple Cues with Long-Term Dependencies," *arXiv Preprint* 1701.01909, 2017.

[42] S. Bae and K. Yoon, "Confidence-Based Data Association and Discriminative Deep Appearance Learning for Robust Online Multi-Object Tracking," *Journal of IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol. 99, 2017.

[43] H. Kieritz, sS. Becker, W. Hübner, and M. Arens, "Online Multi-Person Tracking Using Integral Channel Features," *Proceeding of IEEE Conference on Advanced Video and Signal-based Surveillance,* pp. 122-130, 2016.

[44] Y. Ban, S. Ba, X. Alameda-Pineda, and R. Horaud, "Tracking Multiple Persons Based on a Variational Bayesian Model," *Proceeding of Benchmarking Multi-Target Tracking,* pp. 52-67, 2016.

[45] Y. Song and M. Jeon, "Online Multiple Object Tracking with the Hierarchically Adopted GM-PHD Filter Using Motion and Appearance," *Proceeding of IEEE International Conference on Consumer Electronics-Asia,* pp. 256-259, 2016.

[46] P. Dollar, C. Wojek, B. Schiele, and P. Perona, "Pedestrian Detection: An Evaluation of the State of the Art," *Journal of IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol. 34, No. 4, pp. 743-761, 2011.

[47] R. Stiefelhagen, K. Bernardin, R. Bowers, J. S. Garofolo, D. Mostefa, and P. Soundararajan, "The Clear 2006 Evaluation," *Proceeding of Classification of Events, Activities and Relationships,* pp. 1-44, 2006.

[48] MOT16 Result, https://motchallenge. net/results/MOT16/?det=All (accessed May, 20, 2017).

Quang Dao Vu

He received the Engineer degree in Electrical & Electronic Engineering from Hanoi University of Science and Technology, Hanoi, Vietnam, in 2014. He is currently a research assistant at Embedded Real-time Computing Laboratory, Soongsil University, Seoul, South Korea. His main areas of research interest include embedded systems, image processing, visual surveillance, recognition systems, and deep learning.

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 Embedded Vision Inc., Seoul, South Korea, assistant at Embedded Real- time Computing Lab, University of Soongsil, Seoul, South Korea. His research interests cover the design and analysis of various smart embedded software system, I.O.T and also intelligent image, video analytic algorithms which is 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, biometrics, and digital signage.