Online Multiple Pedestrian Tracking using Deep Temporal Appearance Matching Association

Young-Chul Yoon, Du Yong Kim, Kwangjin Yoon, Young-min Song and Moongu Jeon*

Abstract—In online multiple pedestrian tracking it is of great importance to construct reliable cost matrix for assigning observations to tracks. Each element of cost matrix is constructed by using similarity measure. Many previous works have proposed their own similarity calculation methods consisting of geometric model (e.g. bounding box coordinates) and appearance model. In particular, appearance model contains information with higher dimension compared to geometric model. Thanks to the recent success of deep learning based methods, handling of high dimensional appearance information becomes possible. Among many deep networks, a siamese network with triplet loss is popularly adopted as an appearance feature extractor. Since the siamese network can extract features of each input independently, it is possible to adaptively model tracks (e.g. linear update). However, it is not suitable for multi-object setting that requires comparison with other inputs. In this paper we propose a novel track appearance modeling based on joint inference network to address this issue. The proposed method enables comparison of two inputs to be used for adaptive appearance modeling. It contributes to disambiguating target-observation matching and consolidating the identity consistency. Intensive experimental results support effectiveness of our method.

Index Terms—multi-target tracking, deep learning, appearance model.

I. INTRODUCTION

The purpose of multi-target tracking is to provide accurate trajectories of moving targets from given observations. The produced trajectories are used for position prediction or re-identification. For instance in autonomous vehicle application, it prevents traffic accidents by predicting movement of pedestrians or vehicles. An intelligent surveillance system is supposed to identify criminals using reconstructed trajectories and re-identification algorithms. Since these applications are closely related to public safety, it is important to devise a robust tracking algorithm.

Multi-target tracking algorithm can be categorized into two types of methods according to perspective of handling the given data. One is an online method that processes the current data frame. It is applicable to real-time tasks such as autonomous vehicle. The other is an offline method that exploits the data of whole frames. Although, offline methods show better performance than online methods, in general, it is not suitable for time critical applications. It is computationally expensive because of global optimization process (e.g. linear programming, network flow, graph-cut). The main interest of this paper is real-time tracking, and thus the following sections will be focused on online tracking framework.

Bayesian filtering is widely adopted as an online multi-target tracking framework. It consists of state transition, observation likelihood, and data association. Appearance and motion features are general characteristics of object trajectory. There are relevant Bayesian filtering algorithms such as Kalman filter [1], particle filter [2], probability hypothesis density filter [3]. Usually, Bayesian filters are implemented for point-wise observations in classical target tracking problems such as radar/sonar. However, it is not desirable for visual measurements from RGB videos because targets are often occluded by scene structures. Thus, point based tracking cannot construct long reliable trajectories. From this reason, appearance and other features have been considered for better performance.

In visual multi-target tracking, tracking-by-detection is regarded as a most popular tracking paradigm due to the good quality of bounding box detection algorithms. As tracking performance is dependent upon the quality of detections, a public set of detections is used for fair comparison for tracking. Appearance feature in the bounding box area was regarded as a most popular tracking paradigm due to the good quality of bounding box detection algorithms. As tracking performance is dependent upon the quality of detections, a public set of detections is used for fair comparison for tracking. Appearance feature in the bounding box area was often used for further improvement. The simplest appearance model is a color histogram. Several works [4]–[6] used RGB or HSV based color histogram from targets and observations,

* Corresponding author
Conventional network training

Joint inference network cannot extract target specific feature

Problem definition

negative anchor

inference stage.

Joint-Inference Network since it doesn't copy itself neither in training or related to the method in conference paper but also propose [9]. We not only provide diverse explanation and experiments appearance confidences. In this paper, we extend the work of framework. Similarities between observation and historical to accommodate adaptive appearance modeling in the online trackers. Specifically, historical appearance matching is used for appearance modeling in online trackers [7], [8] because it cannot be used for target-specific appearance feature update according to appearance confidence. In other words, the joint inference network has been a method for node-to-node scoring in offline trackers not for appearance modeling in online trackers.

The first approach to overcoming this limitation of joint inference network has been proposed by Yoon et al. [9]. Specifically, historical appearance matching is used to accommodate adaptive appearance modeling in the framework. Similarities between observation and historical appearances of a track are associated by corresponding appearance confidences. In this paper, we extend the work of [9]. We not only provide diverse explanation and experiments related to the method in conference paper but also propose a new data-driven association model. Our contributions are summarized as follows.

- Hand-crafted association method using appearance confidence measure is proposed to associate outputs of joint inference network;
- To minimize a fine-tuning and heuristic parameters, we propose a data-driven method to associate temporal appearance matching features;
- Different from other works, we regard width and height of pedestrian bounding box as a part of appearance feature and include them into the data-driven method;
- Experimental results provide sufficient evidence of necessity of our methods and provide insights to readers on effect of appearance model in multi-target tracking;

II. RELATED WORKS

In this section, we provide related works in four different aspects, online multi-target tracking, target-specific appearance feature, appearance model with attention mechanism and JI-Net.

Online multi-target tracking: Most online multi-target trackers follow a Bayesian tracking process. It predicts a state of each track using previously assigned observations. Based on this prediction, likelihoods between tracks and new observations are calculated to form a cost matrix. In this subsection, previous works, which focused on modeling geometric states and solving data association, are revisited. Several works [6], [10], [12] modeled track state based on geometric characteristics. Bewley et al. [11] simply modeled track state using Kalman filter and calculate similarity using IOU. Yoon et al. [10] used relative motion analysis to handle temporal motion error or abrupt camera motion. The same author devised a structural constraint [6] to handle assignment problems in video with camera motion more precisely. It additionally used color histogram as a simple appearance model. Milan et al. [12] presented a novel RNN based multi-target tracker using bounding box information. Trackers which merely focused on motion model couldn’t achieve state-of-the-art performance although they exploited a deep Long-Short-Term-Memory (LSTM) network. Thus, we use a simple Kalman filter for a motion analysis and concentrate on an appearance model. After similarity score calculation, cost matrix should be solved satisfying one-to-one constraint. There are a few works [13], [14] concentrated on solving assignment problems. Milan et al. [13] tried to solve assignment problems using LSTM. Rezatofighi et al. [14] revisited complex Joint Probabilistic Data Association (JPDA) and proposed method to take top-N combinations for simplicity. However many state-of-the-art trackers [15], [16] used the conventional hungarian algorithm [17] and showed competitive performance. Because data association algorithm is out of main focus in this paper, we simply use the hungarian algorithm for data association.

Fig. 2: Conventional siamese network in training and test step. In the training step, each image of the triplet input is fed into the siamese network and outputs the triplet loss. The triplet loss tries to make a relative distance between positive pairs closer than the distance between negative pairs. Note that the weights of the network are shared when it processes the triplet input. In inference phase, the siamese network with a single pipeline is used since the distance between inputs is considered rather than the triplet loss.

However performance improvement is not gained much. That is because a simple histogram model contains redundant background information and suffers from changes in imaging conditions including illumination change. To solve this, many handcrafted features have been used.

Recently deep learning based feature extraction has also been adopted to have more discriminative power. The siamese network (Fig. 2) has popularly been used as a deep feature extractor. The network shares weight during training and outputs feature vectors from last fully connected layer. Compared to hand-crafted features, it showed an outstanding accuracy. However, there is a weakness in the siamese network during inference. It only sees one sample during inference and extracts feature of it without considering a counterpart(Fig. 1 right-top box). This weakness aggravates the performance especially when bounding boxes are not well located on target or containing occluded targets (Fig. 1 left 2 boxes). Joint inference structure (Fig. 3) can solve this problem since it takes a concatenated input and infers similarity considering two images simultaneously. However, it has been adopted only for offline trackers [7], [8] because it cannot be used for target-specific appearance feature update according to appearance confidence. In other words, the joint inference network has been a method for node-to-node scoring in offline trackers not for appearance modeling in online trackers.

1It sometimes is regarded as a sort of siamese network. But, we name it Joint-Inference Network since it doesn’t copy itself neither in training or inference stage.

II. RELATED WORKS

In this section, we provide related works in four different aspects, online multi-target tracking, target-specific appearance feature, appearance model with attention mechanism and JI-Net.

Online multi-target tracking: Most online multi-target trackers follow a Bayesian tracking process. It predicts a state of each track using previously assigned observations. Based on this prediction, likelihoods between tracks and new observations are calculated to form a cost matrix. In this subsection, previous works, which focused on modeling geometric states and solving data association, are revisited. Several works [6], [10], [12] modeled track state based on geometric characteristics. Bewley et al. [11] simply modeled track state using Kalman filter and calculate similarity using IOU. Yoon et al. [10] used relative motion analysis to handle temporal motion error or abrupt camera motion. The same author devised a structural constraint [6] to handle assignment problems in video with camera motion more precisely. It additionally used color histogram as a simple appearance model. Milan et al. [12] presented a novel RNN based multi-target tracker using bounding box information. Trackers which merely focused on motion model couldn’t achieve state-of-the-art performance although they exploited a deep Long-Short-Term-Memory (LSTM) network. Thus, we use a simple Kalman filter for a motion analysis and concentrate on an appearance model. After similarity score calculation, cost matrix should be solved satisfying one-to-one constraint. There are a few works [13], [14] concentrated on solving assignment problems. Milan et al. [13] tried to solve assignment problems using LSTM. Rezatofighi et al. [14] revisited complex Joint Probabilistic Data Association (JPDA) and proposed method to take top-N combinations for simplicity. However many state-of-the-art trackers [15], [16] used the conventional hungarian algorithm [17] and showed competitive performance. Because data association algorithm is out of main focus in this paper, we simply use the hungarian algorithm for data association.
Target-specific appearance feature: There have been many existing works on appearance modeling for multi-target tracking. Most of these works suggest extraction of target-specific features from cropped RGB images. In this manner, many hand-crafted features were proposed such as color histogram \( [4], [5] \), optical flow \( [18], [19] \) and histogram of gradients \( [20] \) to name a few. However, performance of those trackers is still limited. Since deep learning was introduced in computer vision, there are several online and offline multi-target trackers that adopts deep learning for appearance modeling. Kim et al. \( [21] \) extracted appearance features through a siamese network and associated those features using LSTM. Then, solved the tracking problem in MHT (Multiple Hypothesis Tracking) framework. Bae et al. \( [15] \) used the siamese network with a triplet loss for appearance modeling and adaptively trained the network during tracking. Son et al. \( [22] \) extended the triplet loss to quadruplet loss with additional margin parameters. It is undeniable that deep architecture brought improvement in tracking performance. However, target-specific feature based methods has a weakness when taking noisy inputs. It cannot consider a counterpart to be compared.

Appearance model with attention mechanism: To get more precise target-specific features, recently, several researchers attempted to apply attention mechanism on raw feature map. Chu et al. \( [23] \) assigned a deep network to each target and trained it during tracking to infer target-specific attention area from extracted features. Although it showed a good tracking performance, memory and time consumption may explode since it assigns a network for each target and conducts an online learning. Zhu et al. \( [24] \) proposed a dual matching attention network. It first extracts features of each input from a bounding-box area independently and computes cosine similarity between two feature vectors. The cosine similarity is used to obtain the attention area. Then, it associate matching features using LSTM. This is one of similar works with ours. But, we much simplified the complex process of making input features for LSTM by adopting straightforward JI-Net. He et al. \( [25] \) similarly applied cosine similarity on global feature map extracted by Fully Convolutional Network (FCN). It used the weighted feature map as an input feature for Recurrent Neural Network (RNN). Although aforementioned trackers tried to get precise appearance features, feature extraction network has fundamentally not been trained for target-specific feature extraction. And they still do not consider a counterpart to be compared.

Joint Inference Network: A Joint-Inference Network (JI-Net) was proposed to address aforementioned issues and has been adopted in offline multi-target tracking problems. Taixe et al. \( [7] \) used the JI-Net to extract appearance similarity feature. It fuses the appearance feature with geometric information using gradient boosting algorithm and solves a global optimization through the linear programming. Tang et al. \( [8] \) additionally concatenated pose information to an input of JI-Net. Output similarity is used to edge cost for global multi-cut problem. Although it shows effective performance in offline framework, it is not suitable to online tracking due to the absence of target-specific features. In this paper we provide ways to apply the JI-Net into the online multi-target tracking framework.

III. PROBLEM FORMULATION

The focus of this paper is online multi-target tracking. Thus, we formulate the problem as the Bayesian filtering framework and discuss appearance based observation likelihood model learning.

An online tracking problem can be represented as a Bayesian recursion formula as follows,

\[
p(x_t|Z_{t-1}) = \int p(x_t|x_{t-1})p(x_{t-1}|Z_{t-1})dx_{t-1}, \quad (1)
\]

\[
p(x_t|Z_t) = \frac{p(z_t|x_t)p(x_t|Z_{t-1})}{p(z_t|Z_{t-1})}, \quad (2)
\]

where \( x_t \) denotes a single target state at frame \( t \). \( Z_t = \{ z_j | j = 1, \ldots, t \} \) indicates a set of observations up to frame \( t \). Eq. (1) describes a prediction of the state by using state transition density \( p(x_t|x_{t-1}) \) and Eq. (2) represents a measurement update by using Bayes rule with observation likelihood density \( p(z_t|x_t) \).

For multi-target tracking we assign a single tracker for each target. Then, it is necessary to construct a robust cost matrix \( \text{Cost} \) for data association between potential tracks and current measurements. The cost at frame \( t \) can be designed by each element as,

\[
\text{Cost}_t(i,j) = -\Lambda_t(i,j), \quad (3)
\]

where \( \Lambda_t(i,j) \) is a similarity, \( p(z_t^t|x_t^i) \), between \( i \)-th target and \( j \)-th observation at frame \( t \). The similarity matrix is depicted as,

\[
\Lambda_t(i,j) = \Lambda_{geo}(z_t^x|x_t^i)p_a(z_t^v|x_t^i), \quad (4)
\]

where \( \Lambda_{geo}(z_t^x|x_t^i) \) and \( p_a(z_t^v|x_t^i) \) represent observation likelihood functions for geometric information (motion and shape) and appearance, respectively. The geometric information can be modeled by the linear/non-linear model of Kalman filter \( [1] \) or particle filter \( [2] \). For multi-target tracking setting Gaussian mixture model is considered in Gaussian mixture probability density filter (GM-PHD) under random finite set formulation \( [26] \). There is an RNN based geometric information modeling \( [12] \), but, the simple linear model \( [24] \) or kalman filter based tracker \( [15] \) still shows comparable performance. So, we adopt the Kalman filter for geometric state modeling in our tracker.

Different from geometric state (2 or 4-dimensional vector in our tracker), appearance feature cannot be simply modeled...
Fig. 4: Activation map comparison between JI-Net and siamese network. Input and corresponding activation maps are marked in same color. We specified layer numbers, correspond to Fig. [10] from which each activation map was activated. (a) Depending on combination of anchor-counterpart, JI-Net extracts different features. Interestingly, an arm part of the anchor image (red circled) was activated only when compared with a negative counterpart. (b) siamese network extracts same features for an anchor image (green) regardless of compared counterpart. So, it resulted high similarities both for positive and negative counterparts.

because of its complexity, i.e., height * width * channel. Popular appearance modeling consists of feature extraction and feature update process. There are two possible feature update methods: linear combination Eq. (5) and likelihood based selection Eq. (6).

\[
f(x_i) = (1 - p(z_i^{|x_i^*} | x_i^*) / \lambda_f) f(x_{i-1}) + \left\{ \begin{array}{ll} f(z_i^{|x_i^*} | x_i^*) / \lambda_f & \text{if } p(z_i^{|x_i^*} | x_i^*) > \tau_a \ f(x_{i-1}), & \text{otherwise} \end{array} \right. \tag{5}
\]

where \( f \) denotes the target-specific feature, modeled by either of color histogram [4], HOG [20], PCA [27] or output of siamese neural network [15]. \( x_i^* \) indicates matched observation index of the target-i after association. \( \lambda_f \) and \( \tau_a \) are update control parameter and feature substitution threshold respectively.

Similar forms of Eq. (5) have frequently been adopted by trackers [4], [6], [25] for appearance modeling. It linearly updates features according to matching likelihood. Eq. (6) substitutes previous feature with a new feature when target-observation likelihood is higher than the predefined threshold. Both methods intend to maintain a robust target-specific appearance feature, but, Eq. (5) enables adaptive appearance feature update according to the detection likelihood. This also reflects the first-order Markov transition density \( p(x_i | x_{i-1}) \) in Eq. (1). So, we set Eq. (5) as a smoothing method for the baseline target-specific feature based trackers.

As we mentioned in previous sections, target-specific feature based methods do not consider a counterpart. This paper incorporates a counterpart in the feature extraction function. Ideally, the goal is to extract a feature of a considering its counterpart \( b \) as denoted by \( f(a|b) \), but, conventional feature extraction is done independently meaning \( f(a|b) = f(a) \).

In this paper we consider an adaptive appearance likelihood model with effective counterpart. Then, the appearance likelihood model is described as

\[
p_a(z_i^i | x_i^i) \propto p_a(f(z_i^i | x_i^i) | f(x_i^i | z_i^i)), \tag{7}
\]

where \( i \) and \( j \) are indices for each detection and target, respectively. So, the likelihood is calculated based on aforementioned counterpart-considering feature \( f(a|b) \).

The conventional method does not take into account the counterpart during the feature extraction, thus, it collapses to \( p_a(z_i^i | x_i^i) \equiv p_a(f(z_i^i) | f(x_i^i)) \).

Joint Inference Network (JI-Net), appearance comparison model in our paper, takes a concatenated input. Thus, it can resolve Eq. (7). Fig. 4 shows an example how JINet outperforms conventional siamese networks. Contrary to the conventional target-specific feature based model, JI-Net exploits deep features in resulting appearance likelihood.

However, it is hard to know the target-specific features from JI-Net, i.e., \( f(x^i) \) cannot be extracted from \( f(x^i | z^j) \) or \( f(z^j | x^i) \). Therefore, we made a new notion, historical appearance, which indicates reliable previous appearance image of target. From this, a new likelihood calculation method is devised as,

\[
p_a(z_i^i | x_i^i) = \sum_{n=1.\ldots N(ha_i^i)} w_n \cdot p_a(f(z_i^i | ha_i^{i,n}) | f(ha_i^{i,n} | z_i^i)), \tag{8}
\]

where \( N(ha_i^i) \) is the size of current historical appearance cue and \( ha_i^{i,k} \) indicates the \( k \)-th template saved in a historical appearance cue. \( w_k \) is a weight of each likelihood when the \( k \)-th historical appearance is considered. This change in the likelihood function makes the model to alleviate the difficulty in finding the most discriminative appearance feature. Especially when track is in ambiguous state, it helps to disambiguate a track-observation matching. Fig. 5 describes the situation that our method resolve matching ambiguity in the object occlusion.

The weight of each likelihood term is the key parameter of the appearance model learning. The proposed method to obtain this key parameter will be subsequently explained in
the following sections.

IV. PROPOSED METHODS

In this section, our proposed method is explained in detail. An overall tracking framework is presented to clarify the flow of the proposed method. The siamese network and JI-Net are explained as a baseline of the framework. Then, two matching association algorithms, Confidence based Temporal Appearance Matching Association (C-TAMA) and Deep-TAMA, are detailed. Finally, historical appearance cue and track management are explained. Remind readers that every hyperparameters, appearing in this section, are specified in Section [V-A] and [V-B].

A. Tracking framework

Our tracking framework is described in Fig. 6. The input for the tracking framework is from publicly accessible dataset and selectively filtered out using NMS and detection confidence threshold before tracking. We calculate the likelihood between detections and tracks and construct a cost matrix with similarities ($\Delta(i,j)$) for each pair. The cost optimization is solved by the hungarian algorithm [17].

According to association results, historical appearance cue and state of each track are updated. For unassigned detections, new tracks are initialized. For initialization, we applied a hierarchical approach. Tracks, not associated with any detection for pre-defined length of consecutive frames, are terminated. More detail is explained in following subsections.

B. Baseline features

In this paper simple image histogram and deep learning feature are considered as baseline features. A combination of HSV and RGB histogram is used as a simple image feature as in [5], [6]. The conventional siamese neural network is considered as a deep learning based feature extractor. For the network structure, we adopt the network and the triplet loss in [15]. The triplet loss is represented as,

\[
Loss = \max(d_{siam}(a,p) - d_{siam}(a,n) + m, 0)
\]  

(9)

where $d_{siam}(a, b) = \|f_{siam}(a) - f_{siam}(b)\|^2$ is a siamese network feature distance between input image $a$ and $b$ and $f_{siam}$ denotes feature extraction function of siamese network. Image patches are denoted as anchor $a$, positive $p$, and negative $n$ patches, respectively, $m$ is a predefined margin for training.

In Fig. 2 illustrates the training step of the siamease network. Color histogram feature, $f_{col}$, consists of normalized HSV-RGB histogram with 8bins per each color field, $N(f_{col}) = 48$.

There are two types of methods to calculate similarity between features, inverse exponential of feature distance [4], [15] and sum of element-wise multiplication [6]. Since both methods fundamentally share a same supposition that corresponding elements from two vectors should be similar, there would not exist a huge gap in performance between them. We adopted the former one for a feature, extracted from a siamese network, and the latter one for a color histogram as,

\[
p_a^{siam} (z_j | x_i) \propto \exp (- \| f_{siam}(z_j) - f_{siam}(x_i) \|^2 ),
\]  

(10)

\[
p_a^{col} (z_j | x_i) \propto \sum_{k \in \{1, \ldots, N(f_{col})\}} \sqrt{f_{col}(z_j) \odot f_{col}(x_i)},
\]  

(11)

where $\odot$ indicates element-wise multiplication. $x_i$ and $z_j$ indicate i-th track and j-th observation respectively. $k$ indicates $k$-th element of $f_{col}$. It is worth noting that this likelihood function is oblivious of counterpart during feature extraction. In Section [V], Eq. (10) and Eq. (11) are adopted by baseline methods, triplet-siamese and color histogram, respectively.

C. Joint inference network

Different from target-specific feature extraction methods, the JI-Net method directly outputs a normalized similarity score in the range from 0 to 1. Fig. 3 illustrates our JI-Net structure. Since it is similar to a binary classification problem, a softmax binary cross-entropy loss is adopted as described in the following equation,

\[
Loss = -(y \cdot \log(g(s_p)) + (1 - y) \log(1 - g(s_n))),
\]  

(12)

\[
g(s_i) = \frac{\exp(s_i)}{\sum_{j \in \{p,n\}} \exp(s_j)},
\]

where $y$ is a ground-truth label (1 or 0 in our case), and $g(s)$ is a softmax function for an input $s$. $s_p$ and $s_n$ indicate raw...
output values from the last fully connected layer. Processed by the softmax function, probabilities of positive or negative class are obtained. In the test time, we use the probability of the positive class $g(s_p)$ as an appearance likelihood function.

$$p_a(I^J(z|x)) \equiv g(s_p), \quad (13)$$

With the appearance similarity likelihood function [13], the remaining problem is an adaptive target appearance modeling. To overcome the absence of target-specific feature, we propose two methods, confidence based and data-driven matching association in following sections. Note that for JI-Net, one of our baseline in experimental section, takes (13) directly as a final appearance likelihood without applying association methods, described in following sections.

**D. Confidence based temporal appearance matching association**

We first propose a method to associate output scores of the JI-Net according to association results. This method was originally proposed in our conference paper [9]. Since each JI-Net according to association results. This method was originally proposed in our conference paper [9]. Since each JI-Net has has recent appearance confidence and $\lambda_c$, $\gamma$-th historical appearance, $h_{a_n}^i$, respectively. These likelihoods are associated through appearance confidence variables, $c_{rcnt}^i$, $w_n^i$, where $c_{rcnt}^i$ is the most recent appearance confidence and $w_n^i$ indicates normalized appearance confidence of the $n$-th historical appearance in the cue. $N(h_{a_n}^i)$ is the number of saved historical appearances. To control the effect of recent appearance confidence, $\lambda_c$ has been adopted. $w_n^i$ is derived by following equation,

$$w_n^i = \frac{c_n^i}{\sum_{k=1}^{i} N(h_{a_k}) \cdot c_k^i}, \quad (15)$$

where $c_n^i$ is an appearance confidence of the $n$-th historical appearance in the cue. Through this association, matching scores between the $j$-th observation, $z^j$, and the $n$-th historical appearance, $h_{a_n}^i$, are all considered. The method is described in Fig. 7. According to the recent appearance confidence, $c_{rcnt}^i$, dependency on the recent appearance $a_{rcnt}^i$, is decided. Matching scores with saved historical appearances are associated through corresponding appearance confidence. The appearance confidence is calculated equivalent to Eq. (4) and jointly managed with historical appearances as follows,

$$c_{rcnt}^i = \Lambda_{rcnt}(i, j^*), \quad (16)$$

$$H_{cue}^i = [(c_1^i, h_{a_1}^i), \ldots, (c_N(h_{a_N})^i, h_{a_N}^i)]$$

where $j^*$ is a recently associated observation index. For track $i$, there must exist an observation, $z^j_{rcnt}$, which has recently been associated with it in the previous frame, $c_{rcnt}$, Note that the corresponding $a_{rcnt}^i$ is a cropped image of $z^j_{rcnt}$. Management and update protocol of the historical appearance cue, $H_{cue}^i$, is explained in Section IV-F with detail. The historical appearance cue is similarly used in following data-driven method.

In summary, the predicted appearance likelihood is calculated in the form of weighted combination. The contribution of the recent appearance is proportional to its appearance confidence, $c_{rcnt}^i$ and the matching scores with saved historical appearances associated through corresponding appearance confidences.

C-TAMA calculates weights, $w_n^i$, of Eq. (8) using appearance confidence. Although tracking performance was improved as proved in [9] and following ablation studies, it still has limitations of non-adaptive appearance confidence. To alleviate this limitation, we present an adaptive association method via data-driven approach in the following section.

**E. Data-driven temporal appearance matching association**

C-TAMA contains user selected parameters and it may need additional tuning depending on scene condition. Inspired by [16], [21], [23], that associate geometric and appearance features through recurrent neural network (RNN), we adopted LSTM networks [29] (one of well-known RNNs) for matching feature association. Since it uses a deep network for association, we call it Deep-TAMA. LSTM shows good performance when processing time-series data like video [30] or periodic climate data [31]. Since historical appearances can be regarded as sequential data, which means the $(k-1)$-th historical appearance must have been appeared earlier than the $k$-th historical appearance, those data fits the purpose of LSTM. Intermediate feature of JI-Net, represented in Fig. 3 is used as an input feature of LSTM. Structure of Deep-TAMA is described in Fig. 8
Input feature of LSTM consists of intermediate feature of JI-Net, $f_{itm}$, and relative shape difference. The intermediate feature $f_{itm}$ is a hidden layer before the last fully connected layer that outputs $s \in \mathbb{R}^2$ of Eq. (12) from $f_{itm}$. Relative shape difference is one of our contributions which reshapes the performance of Deep-TAMA. Both in the JI-Net and conventional siamese network, input image should be reshaped into the designated size before insertion. So, shape information is lost before inference and this shape information should be manually added for better inference. LSTM input feature is defined as following equations,

$$
\begin{align*}
    f^k_{in} &= [f^k_{itm}, r^k_d], \\
    r^k_d &= \Delta \text{width}_{k_j} / \text{width}_{j}, \Delta \text{height}_{k_j} / \text{height}_{j},
\end{align*}
$$

(17)

where $[A, B]$ indicates concatenation of two feature $A$ and $B$. $r^k_d$ is a relative shape difference between the $k$-th historical appearance $h_{a_k}$ and the $j$-th observation $z_j$. For readability, $i$ and $t$ have been omitted. Relative difference was inspired from bounding-box regression loss of recent detectors [32], [33]. The relative difference is required to handle the various size of the bounding-boxes and to prevent biased inference (e.g., bigger bounding-box usually make bigger shape difference). So we divide the shape difference by the anchor shape. The final concatenated feature size is 152, (150 + 2). We adopted the conventional LSTM cell and trained it to associate the $f^k_{in}$. An LSTM cell consists of following gates ($F_k$, $I_k$, $O_k$ are forget, input and output gates),

$$
\begin{align*}
    F_k &= \sigma(W_f \cdot [H_{k-1}, f^k_{in}]), \\
    I_k &= \sigma(W_i \cdot [H_{k-1}, f^k_{in}]), \\
    O_k &= \tanh(W_o \cdot [H_{k-1}, f^k_{in}]),
\end{align*}
$$

(18)

where $H_{k-1}$ is a hidden state generated by the previous $(k - 1)$-th LSTM cell. $f^k_{in}$ is an input feature of current LSTM cell. The weights $W \in \mathbb{R}^{D_{row} \times D_{col}}$ are learnable and shared by all LSTM cells. Each matrix projects size $D_{col}$ input vector to size $D_{row}$ output vector. Here, $D_{col}$ is equivalent to the sum of the size, $f^k_{in}$ and $H_k$, i.e., (152 + $D_{row}$). $D_{row}$ will be explained in implementation details. Sigmoid ($\sigma$) and hyperbolic tangent ($\tanh$) activate the result of matrix multiplication. Each of these three gates performs a role of controller as

$$
\begin{align*}
    c_k &= F_k \circ c_{k-1} + I_k \circ \tilde{c}_k, \\
    \tilde{c}_k &= \sigma(W_c \cdot [H_{k-1}, f^k_{in}]), \\
    H_k &= O_k \circ \tanh(c_k),
\end{align*}
$$

(19)

where each gate controls input, cell state or hidden state by hadamard product, $\circ$. $c_k$ is a cell state of the $k$-th LSTM cell and $\tilde{c}_k$ is data to be used for cell state update. The computed $k$-th cell output $c_k$ is propagated to the $(k + 1)$-th LSTM cell. Note that the intermediate feature of pair, $(a_{rent}^i, z_j^i)$, is extracted for the last LSTM cell, $H_N$, as described in Fig. 8. The last N-th hidden state ($H_N$) is projected into the size-2 vector through one fully connected layer. These output vectors are identically treated and trained as an output vector of the JI-Net, Eq. (12)-(13).

The first element of the vector corresponds to the usable likelihood, $p_a(z_j^i | x^i)$. So, the final likelihood of Deep-TAMA is represented as,

$$
\begin{align*}
    p_a(z_j^i | x^i) &= \frac{\exp(s_p)}{\sum_{j \in \{p, n\}} \exp(s_j)},
\end{align*}
$$

(20)

where $\circ$ indicates element-wise product and $w_{fc}^j$ indicates learnable weights of fully connected layer which projects the input vector to the $j$-th output element, $s_j$.

Deep-TAMA substituted non-adaptive parts of C-TAMA, Eq. (14)-(15), to deep neural network. Although C-TAMA showed a good performance compared to baseline methods in experiments, there may exist better parameters for $c_k$ that are hard to tune. Deep-TAMA successfully removed this concern by deriving $w_{n}$ of Eq. (8) through data-driven way. Various experiments will be delivered in Section V to validate effectiveness of our methods.

F. Historical appearance cue management

So far, we have explained methods to get reliable appearance likelihood using historical appearances. Thus, deletion and addition protocol of historical appearance cue, $H_{cue}$, should be addressed. In this subsection, we introduce four management protocols which include the maximum length of cue, the maximum age of historical appearance, the minimum interval between each addition and confidence threshold.

Deletion protocol: It is obvious that our exhaustive matching association method takes much longer time than baseline methods. So, the maximum length of the cue is necessary to relieve Big-O time complexity. Let us suppose the number of tracks, observations and length of historical appearance cue as $N_{rk}$, $N_{obs}$ and $N_{cue}$ respectively. Then Big-O time complexity becomes $O(N_{rk}N_{cue}N_{obs})$. If the new historical appearance is stacked in the cue without removing aged ones,
the time complexity may explode. So, we limit the maximum size of cue as,

\[ N_{\text{cue}} \leq \tau_{\text{cue}}, \]  

(21)

where \( \tau_{\text{cue}} \) is a predefined cue size threshold. Next, as a historical appearance gets older, it gets farther from a recent appearance of the target, e.g. pose, illumination, size. The aging gets accelerated as frame-per-second (FPS) decreases. So, we related the maximum age of historical appearance to FPS as,

\[ fr_{\text{cur}} - fr(ha_k^t) \leq \text{FPS} \cdot \beta_{\text{age}}, \]  

(22)

where \( \beta_{\text{age}} \) and \( fr_{\text{cur}} \) indicate a control parameter and a current frame respectively. \( fr(ha_k^t) \) is a frame that \( ha_k^t \), the \( k \)-th historical appearance of track \( i \), appeared.

### Addition protocol

As an extension of Eq. (22), it is desirable to prevent saving too similar historical appearances in the cue. So, the minimum update interval is required for diversity of historical appearances. If we take matched appearances of the \( t \)-th frame and the \((t+1)\)-th frame as historical appearances, those could almost be duplicate. Degree of difference between two patches is proportional to FPS of video in common sense, i.e., degree of difference is larger in higher FPS video and vice versa. The minimum update interval is defined as,

\[ fr(ha_k^t) - fr(ha_{k-1}^t) \geq \text{FPS} \cdot \beta_{\text{intv}}, \]  

(23)

where \( \beta_{\text{intv}} \) is a control parameter. Lastly, historical appearance cue only can include reliable appearance of target. From this reason, only \( a_{\text{rcnt}}^i \), having high confidence, \( c_{\text{rcnt}}^i > \tau_{\text{hist}} \), can be added on \( H_{\text{cue}} \).

Here we summarize a composite update protocol as,

\[ H_{\text{cue}}^t = \begin{cases} H_{\text{cue}}^t \setminus (c_{\text{rcnt}}^i, ha_k^t), & \text{Eq. (21)} | \text{Eq. (22)} \\ [H_{\text{cue}}^t, (c_{\text{rcnt}}^i, a_{\text{rcnt}}^i)], & \text{Eq. (23)} \cup \text{Eq. (24)} \end{cases}, \]  

(24)

where \( \setminus \) indicates exclusion. Historical appearance number is a relative number in the cue. So when the first pair of historical appearance and confidence is removed by constraints, Eq. (21) or Eq. (22), the second one becomes the first historical appearance. The pair, \( (c_{\text{rcnt}}^i, a_{\text{rcnt}}^i) \) is appended on a historical appearance cue only when \( c_{\text{rcnt}}^i \), Eq. (16) is bigger than \( \tau_{\text{hist}} \). Otherwise, the pair is discarded if the track found a new matching observation. Note that \( c_{\text{rk}}^i \) are not utilized in DeepTAMA because LSTM cells substitute the behavior of them.

### G. Track initialization and termination

For initialization, we constructed a hypothesis tree using hierarchical matching. With assumption that geometric state of same target doesn’t change a lot in consecutive frames, we first check Intersection-Over-Union (IOU) between non-associated observations (\( \tilde{z} \)) and hypotheses (\( h_{n,k}^{i,t} \)) as

\[ J = \{ j \mid \text{IOU}(\tilde{z}_j^t, h_{n,k}^{i,t}) > \tau_{\text{ion}} \} \]

(25)

\[ h_{n-1}^k \in H_{n-1}^t, (k = 1 \cdots N(H_{n-1}^t)) \]

\[ H_t^i = \{ \tilde{z}_j^t | j \in J \}, J \neq \phi \]

(26)

\[ \phi, \text{ else} \]

where \( H_t^i \) indicates \( n \)-th hypothesis tree in frame \( t \). \( h_{n,k}^{i,t} \) is a \( k \)-th node of \( H_t^i \). If there’s no matching \( \tilde{z} \), track hypothesis tree is removed. When depth-5 hypothesis tree is created, it starts a new track with 5 bounding boxes stored in hypothesis tree. But there’s a limitation of IOU based matching. Strict IOU based matching can miss true-matching observations in complex scene condition like low FPS, variant camera perspective or camera movement. So, instead of removing hypothesis tree, we use a less strict matching measure if there’s no IOU matching.

\[ \text{dist}_p = \| \text{pos}(h_{n,k}^{i,t}) - \text{pos}(\tilde{z}_j^t) \| \]

(27)

\[ shp_s = \min \left( \frac{\text{height}(\tilde{z}_j^t)}{\text{height}(h_{n,k}^{i,t})}, \frac{\text{height}(h_{n,k}^{i,t})}{\text{height}(\tilde{z}_j^t)} \right) \]

\[ H_t^i = \begin{cases} \{ \tilde{z}_j^t \}, & \text{dist}_p < \beta_{\text{dist}} \cdot \text{width}(\tilde{z}_j^t) \cap shp_s > \tau_{\text{shp}} \\ \phi, & \text{else} \end{cases} \]  

(28)

where \( \text{pos}(\cdot) \) and \( \text{height}(\cdot) \) denote the center coordinate and the height of bounding-box. \( \beta_{\text{dist}} \) and \( \tau_{\text{shp}} \) are heuristically selected control parameter and threshold respectively. Distance based matching is relatively weak constraint than IOU since it separately measures position \((x, y)\) difference and shape similarity between non-associated observation and hypothesis. So, it is able to find matching with little bit distant observation with similar shape. Observations, which couldn’t find matching tracks and existing hypothesis, become root node of a new tree, \( H_{n+1}^i \). \( H^n \) with empty \( H_t^i \) is regarded as a false-positive hypothesis and removed.

Track termination can be simply implemented than initialization. We removed tracks which failed to find matching observation up to \( \tau_{\text{term}} = \text{FPS} \cdot \beta_{\text{term}} \) frames. We assume here that avoiding occlusion takes longer time in lower FPS video and vice versa. For this reason, track termination threshold, \( \tau_{\text{term}} \), is related to FPS. \( \beta_{\text{term}} \) was decided as 2 heuristically.

### H. Affinity matrix construction

We have defined the elements of cost matrix as Eq. (3). The appearance likelihood \( p_a(\tilde{z}_j^t | x_t^i) \) is implemented as proposed in the previous sections. For geometric state likeness, Kalman filter is used, where the motion and shape are considered simultaneously. \( [A \mid B] \) projected motion and shape into a single matrix whereas several works \([6, 15]\) constructed two independent matrices and modeled shape and motion likelihood separately. Then the multiplied of two likelihood to get the final geometric likelihood is implemented \( p_{\text{geo}}(\tilde{z}_j^t | x_t^i) = p_{\text{geo}}(\tilde{z}_j^t | x_t^i) p_s(\tilde{z}_j^t | x_t^i) \). In our paper, except DeepTAMA which includes the shape information in appearance similarity inference, the motion and the shape are modeled by the second method, i.e., separated motion and shape modeling. Since the dimension of the motion state \((x, y)\) and the shape state \((\text{width}, \text{height})\) are equally 2, we also applied Kalman filter for shape modeling. The likelihood of geometric states
Algorithm 1 Multi-target tracking process

1: \( X_t = \{i\}, H_t = Z_t, N(A) = \text{number of element in } A \)
2: for \( t = 1 \) to \( N(T) \) do \( \triangleright \) loop until end of video
3: \( G = \{(i,j) | i = 1 \ldots N(X_t), j = 1 \ldots N(Z_t)\} \)
4: for \( (i,j) \) in \( G \) do \( \triangleright \) construct a cost matrix
5: \( p_{geo}(z_i^t | x_i^t) = \text{Eq. } 29 \)
6: if \( p_{geo}(z_i^t | x_i^t) > \tau_{\text{match}} \) then \( \triangleright \) geometric gating
7: \( p_a(z_i^t | x_i^t) = \text{Eq. } 14 \text{ Eq. } 20 \)
8: \( p^j(z_i^t | x_i^t) = p_{geo}(z_i^t | x_i^t) * p_a(z_i^t | x_i^t) \)
9: else
10: \( p^j(z_i^t | x_i^t) = 0 \)
11: Cost\((i,j) = -p(z_i^t | x_i^t), j \neq N(Z_t) + 1 \)
12: Cost\((i,N(Z_t) + 1) = -\tau_{\text{match}} \)
13: \( M = \text{hungarian}(\text{Cost}) \) \( \triangleright \) 1-to-1 assignment
14: for \( (i,j) \) in \( M \) do
15: if \( j == N(Z_t) + 1 \) then
16: miss\((x_i^t) = \text{miss}(z_i^t) + 1 \)
17: if \( \text{miss}(z_i^t) \geq \tau_{\text{term}} \) then \( \triangleright \) termination
18: \( X_t = X_t - x_i^t \)
19: remove \( (i,j) \) from \( M \)
20: else
21: historical appearance cue update (Eq. 24)
22: \( \hat{Z}_t = \{z_i^t \} \ni (x_i^t) \notin M \) \( \triangleright \) for all possible indices
23: update \( H_t \) using \( \hat{Z}_t \) (Eq. 25-28)
24: for \( H_t^n \) in \( H_t \) do
25: if depth\((H_t^n) \geq 4 \) \( \triangleright \) initialization
26: \( x_{t+1}^n = H_t^n, \text{conn} \)
27: for all \( \text{conn} : \text{connected indices from leaf to root} \)
28: \( X_t = X_t + x_{t+1}^n, H_t^n = \emptyset \)
29: for \( (i,j) \) in \( M \) do \( \triangleright \) Kalman filtering
30: update state \( x_{t+1}^i \) from \( x_{t+1}^i \) and \( z_i^t \)
31: predict state \( x_{t+1}^i \) for all \( x_{t+1}^i \) in \( X_t \rightarrow X_t+1 \)

between track and observation are calculated as,
\[
p_m(z|x) = \exp(-\xi(\text{pos}(z) - \text{pos}(x)^T)\Sigma(\text{pos}(z) - \text{pos}(x))),
\]
\[
p_s(z|x) = \exp\left(-\epsilon\left\{\frac{\Delta_{\text{height}}}{H_{\text{height}}} + \frac{\Delta_{\text{width}}}{W_{\text{width}}}\right\}\right),
\]
\[
\Delta_{\text{height}}(x,z) = |\text{height}(x) - \text{height}(z)|,
\]
\[
\Delta_{\text{width}}(x,z) = |\text{width}(x) - \text{width}(z)|,
\]
\[
\tau_{\text{height}}(x,z) = \text{height}(x) + \text{height}(z),
\]
\[
\tau_{\text{width}}(x,z) = \text{width}(x) + \text{width}(z),
\]
(29)

where we omitted \( i, j \) and \( t \) for simplicity. \( \Sigma \) is originally an inverse of covariance matrix in Mahalanobis distance. Due to the failure during camera movement or occlusion, we use a matrix with fixed values that work well on most environments. The hungarian algorithm [17] solves the constructed cost matrix subjects to one-to-one matching. Only the matching results having higher likelihood than \( \tau_{\text{match}} \) are regarded as a valid matching. Again, note that Deep-TAMA uses only \( p_m(z|x) \) as \( p_{\text{geo}}(z|x) \) because the shape \( \text{height and width} \) is exploited as an appearance information, not a part of the state. For better understanding, we summarize the whole

V. EXPERIMENTS

In this section, experimental results are delivered to show effectiveness of our proposed methods. This section consists of three main parts. Training setting and implementation detail are provided first. Then ablation studies including parameter tuning experiments will be detailed. Finally, our trackers will be compared with state-of-the-art trackers on popular MOT benchmark.

A. Implementation detail

We implemented the whole framework using Matlab and MatConvnet [35]. To accelerate the computation, Titan X with 12GB memory has been used when training and testing siamese and JI-Net. However, since LSTM implementation of MatConvnet doesn’t support CUDA, LSTM computation is a time bottleneck of our framework. We are sure that this is not a problem if re-implemented using Tensorflow or other frameworks. To help re-implementation for readers, we provide implementation details below.

Dataset preparation: We used training sets of 2DMOT2015 [36], MOT16 [37] and CVPR19 challenge [38] for training and validation. As described in Fig. 9, we split the whole

https://motchallenge.net/data/CVPR_2019_Tracking_Challenge/
sequences into training and validation sets. Training set and validation set1 contain both static and dynamic scenes. Validation set2 consists of newly published CVPR19 challenge dataset [38] which represents extremely crowded environment. Training set is used for training our proposed networks and baseline siamese network. Validation set1 is used for parameter tuning and baseline comparison. Finally, baseline comparison is conducted once more on validation set2 to strengthen the generality. Test sets of MOT16 and MOT17 are used for benchmark result comparison in Section V-C.

Neural-Net setting and training: We designed the JI-Net and siamese network structures as Fig. 10. It basically follows the siamese network structure, described in [15]. Additionally, batch normalization layers [39] were adopted to prevent divergence and overfitting. To train the siamese network or JI-Net, we chose the anchor and corresponding positive, negative samples randomly. To be specific about JI-Net training, 1000 positive samples and 1000 negative samples were inserted per epoch with batch size 32. Those samples were augmented by adding noise during cropping, random noise in center coordinate and bounding box-size, and random brightness change, from 0.8 to 1.2. It took 200 epochs to converge. Next, we selected LSTM structure with the weights, $W \in \mathbb{R}^{(128 \times (128+152))}$ following [16]. This looks small but is proven to work fine without redundancy in Section V-B.

To train the LSTM for Deep-TAMA, we artificially generated positive and negative tracks. Each artificial track consists of the maximum 14 pedestrian patches and 1 anchor image. These 15 track patches were randomly sampled from continuous 40-frames trajectory of the same pedestrian. This 40-frames trajectory is also randomly sampled from whole trajectory of the pedestrian. Different from [21] which randomly added a false image in the artificial track, we didn’t put any false images. In real tracking situation, thanks to geometric constraint, tracks are rarely matched to the false target unexpectedly. So, we assume that random noise during bounding-box cropping can sufficiently reflect the real tracking situations. An identical positive-negative sample ratio is used for training of JI-Net and siamese network. We used the stochastic gradient descent (SGD) to optimize weights of both feature extractor and LSTM. The training starts from learning rate 0.001 with learning rate decay 0.97 per every epoch.

Heuristic hyperparameters: We specify every hyperparameters, mentioned in Section IV, in a single table below.

| $\beta_{age}$ | $\beta_{int}$ | $\tau_{hit}$ | $\tau_{comm}$ | $\beta_{hit}$ | $\beta_{temp}$ | $\tau_{match}$ |
|---------------|---------------|--------------|--------------|--------------|---------------|--------------|
| 2             | 0.2           | 0.6          | 0.5          | 0.8          | 0.8           | 2            | 0.4          |

The critical hyperparameter, $\tau_{cue}$, is determined by experiments, conducted in following ablation studies.

B. Ablation studies

In this section, we validate our proposed methods and LSTM structure of Deep-TAMA. It is hard to evaluate tracking performance of every LSTM network trained in different settings. So, we compare their training graphs and find the best one which shows the lowest converged loss value. For tracking score comparison, MOTA [40] and IDF1 [41] are considered simultaneously. Except baseline comparison 2, validation set1 has been used for experiments.

LSTM training graph comparison: To confirm an effectiveness of the relative width and height, $rd_{j}^{k} = \frac{d_{j}^{k}}{(width_j, height_j)}$ in Eq. (17), we have conducted several experiments. We provide converging graphs of cross-entropy loss value in various settings at Fig. 11. We picked 1000 positive pairs and 1000 negative pairs from validation set1 before start training and averaged the binary cross-entropy loss calculated on those samples. Note that these samples were samey used for all settings. In Fig. 11a we compared training graph in three settings, only $f_{stim}$, $\hat{f}_{stim}$ with raw $d_{j}^{k}$ and $f_{stim}$ with $rd_{j}^{k}$. Since the scale of $d_{j}^{k}$ is larger than the scale of JI-Net output feature values, it aggravates the training process. $rd_{j}^{k}$ critically improved the performance. In

Fig. 11: Ablation studies by comparing LSTM training graph. (a) Comparison of training setting, with and without relative shape difference information. (b) Comparison with simple normalization methods. (c) Comparison by changing the size of intermediate feature and hidden state.

Fig. 12: Comparison of tracking performance in various $\tau_{cue}$.
Red circled x-axis number indicates selected $\tau_{cue}$.

Figures 11 and 12 present the results of ablation studies and tracking performance comparisons, respectively. These figures demonstrate the effectiveness of the relative width and height ($rd_{j}^{k}$) and the improvement in training efficiency and tracking performance through proper hyperparameter tuning.
Fig. 13: MOTA and IDF1 scores according to change of $\lambda_f$ or $\lambda_c$. Red circled x-axes indicate a selected $\lambda$.

Fig. 14: Performance improvement with hierarchical matching based initialization.

Fig. 11b, validity of $rd^b_j$ with respect to $d^b_j/100$ and $d^b_j/1000$ is confirmed. $rd^b_j$ shows a lowest converged loss value. In Fig. 11c, additional experiments have been performed to relieve redundancy of Deep-TAMA network. We designed a larger JI-Net, originated from [36], with the higher dimension of intermediate feature, 512, and varied the size of LSTM hidden state from 128 to 256. $f_{itm} \in \mathbb{R}^{512}$ leads to a higher loss value than $f_{itm} \in \mathbb{R}^{256}$. Since [36] concatenated additional optical flow information to the input, the structure would have been redundant for our pure RGB image based input. Lastly, though [21] set its LSTM hidden state size as 512, in our experiment, the hidden state with size 256 did not make the significant improvement than the state with size 128. Thus, we determined the JI-Net as described in Fig. 10 and the size of the LSTM hidden state as 128.

The maximum length of historical appearance cue: As we mentioned in Section IV-F, $\tau_{cue}$, in Eq. (21), is one of the important hyperparameters because $\tau_{cue}$ is directly related to the maximum capacity of the LSTM. So, we performed experiments to select the best $\tau_{cue}$. Due to the training method with artificially generated tracks of random length ranging [1,15] for the LSTM cells of the Deep-TAMA, it is able to handle various length of historical appearance cue. We varied $\tau_{cue}$ from 1 to 14 with interval 1 and compared MOTA and IDF1. Comparison results are represented in Fig. 12. It showed the best performance in average, 34.9 MOTA and 26.3 IDF1, when $\tau_{cue}$ is 8. So, $\tau_{cue}$ is fixed into 8 in further experiments.

Control parameter variation: There exist control parameters in Eq. (5) and (14), $\lambda_f$ and $\lambda_c$ respectively. For fair comparison, the best performing control parameter for each appearance model should be selected. So, we measured MOTA and IDF1 by varying lambda value as depicted in Fig. 13. After analysis, {2, 4, 3} are selected for $\lambda_f$ of color histogram, triplet-siamese and $\lambda_c$ of C-TAMA respectively. MOTA and IDF1 scores, outputted from selected $\lambda$, are directly used for baseline comparisons.

Hierarchical initialization: To prove a benefit of our initialization method, we’ve conducted simple comparison. Hierarchical initialization consists of two sub-parts, IOU based strict matching and distance based weak matching. We compare hierarchical method with these two baseline methods. In Fig. 14 we compared on three metrics, MOTA, FP and FN. Since initialization method is in charge of whole tracks, FP and FN are necessary metrics to be compared. Strict IOU based initialization made a lot of False-Negatives. In contrast, weak distance based initialization made a lot of False-Positives. Hierarchical initialization method shows well balanced number of False-Positives and False-Negatives. So it results the best performance in terms of MOTA.

Comparison with baselines: Five different methods have been evaluated on the validation set1. Table I, and set2, Table II. Except that Deep-TAMA removed the shape similarity from similarity calculation, every other minor conditions are shared equally. Color histogram and triplet-siamese take Eq. (5) and JI-Net take Eq. (6) with $\tau_a = 0.6$ for appearance modeling. We additionally included Multi-Object Tracking Precision(MOTP), Mostly Tracked(MT) and Mostly Lost(ML) metrics for detailed comparison.

Table I shows the results on validation set1. RGB-HSV color histogram shows the lowest performance without argument. Deep appearance model, triplet-siamese and JI-Net, outperform the color histogram. JI-Net without TAMA performs slightly better than triplet-siamese. This shows the ability of JI-Net which outputs reliable likelihood in ambiguous comparison. Further improvement can be achieved with temporal appearance modeling. C-TAMA showed improved performance than JI-Net in MOTA and IDF1, 0.4 and 1.2 respectively. Note that it is hard to improve large portion of MOTA and IDF1 by changing only appearance model because geometric gating is performed before the appearance likelihood calcula-

| Method          | MOTA↑ | MOTP↑ | IDF1↑ | MT↑ | ML↑ |
|-----------------|-------|-------|-------|-----|-----|
| Color Hist      | 33.0  | 73.0  | 24.3  | 62  | 144 |
| Triplet_Siamese | 33.8  | 73.5  | 25.2  | 75  | 136 |
| JI-Net          | 34.0  | 73.5  | 25.1  | 76  | 134 |
| C-TAMA          | 34.4  | 73.6  | 27.2  | 76  | 134 |
| Deep-TAMA       | 34.9  | 73.5  | 26.3  | 78  | 121 |

TABLE I: Comparison with baseline methods on validation set1. Red indicates the best score.

| Method          | MOTA↑ | MOTP↑ | IDF1↑ | MT↑ | ML↑ |
|-----------------|-------|-------|-------|-----|-----|
| Color Hist      | 59.5  | 86.1  | 53.3  | 773 | 383 |
| Triplet_Siamese | 60.1  | 86.2  | 53.9  | 786 | 369 |
| JI-Net          | 60.3  | 86.5  | 53.5  | 780 | 374 |
| C-TAMA          | 60.4  | 86.1  | 53.5  | 809 | 365 |
| Deep-TAMA       | 61.2  | 85.9  | 56.9  | 824 | 366 |

TABLE II: Comparison with baseline methods on validation set2. Red indicates the best score.
In this section, we provide quantitative results on two MOT-Challenge benchmark tables (MOT16 and MOT17). We chose state-of-the-art trackers in MOT16 and MOT17 benchmark for comparison. For better visibility, name of trackers, which are state-of-the-art trackers in MOT16 and MOT17 benchmark tables (MOT16 and MOT17). We chose C-TAMA outperforms the baseline method (JI-Net) on actual tracking scene. In perspective of MT and ML, Deep-TAMA reduced nearly 10% of lost tracks compared to other methods. We assume that this improvement comes from successful detachment of shape constraint from Eq. [3].

Table III, performance improvement on main metrics, MOTA and IDF1, are relatively small. In contrast, Deep-TAMA consistently shows visible improvement in every metrics except MOTP. From the results so far, we can summarize that the enhancement from C-TAMA may depend on scene condition and data-driven weights of Deep-TAMA removed this concern effectively.

In Fig. 15, we visually support how our method (Deep-TAMA) outperforms the baseline method (JI-Net) on actual tracking scene.

C. Benchmark results

In this section, we provide quantitative results on two MOT-Challenge benchmark tables (MOT16 and MOT17). We chose state-of-the-art trackers in MOT16 and MOT17 benchmark for comparison. For better visibility, name of trackers, which are necessary to be compared with ours, are highlighted by the bold text.

MOT16 : In MOT16 benchmark, Table III our tracker achieved 46.2 MOTA and 49.4 IDF1. As the top performer tracker in the table scored 48.8% in MOTA, ours can be regarded as one of the state-of-the-art trackers. Offline trackers shows higher performance than online trackers in average. [8] performs the best in MOTA. It proves the great performance of JI-Net in offline tracking framework. [23], [24] share a similar motivation with ours. They used pre-trained feature map, not trained in purpose of pedestrian discrimination, and weighted parts of the feature map in which pedestrian related features are expected to exist. Our tracker shows slightly better MOTA than those two trackers. In contrary, they perform better in a few measures, IDF1, MT and ML. We guess that the reason for this is due to the visual object tracking (VOT) part adopted in those trackers. As a trade-off, VOT produced a lot of false-positives and aggravated MOTA score of them. Also, our appearance similarity network is much simpler to implement than theirs. [27] used ImageNet pretrained feature map with PCA without weighting. Although it is an offline tracker, ours shows better MOTA and IDF1. [15], [49] used triplet siamese and [22] adopted quadruplet loss, modified version of triplet loss. It is clear that ours shows far higher performance in aspect of MOTA and IDF1. [16] exploited LSTM network in motion, appearance and structural similarity inference. We didn’t exploit the 3rd cue like structural similarity and simply modeled motion of track using kalman filter. Since ours shows competitive MOTA and IDF1 to [16], it is sufficient to prove

| Tracker | MOTA↑ | IDF1↑ | MOTP↑ | MT↑ | ML↓ | IDs↓ | FM↓ | FP↓ | FN↓ | Hz↑ |
|---------|--------|--------|--------|------|-----|------|-----|-----|-----|-----|
| offline |        |        |        |      |     |      |     |     |     |     |
| LMP [8] | 48.8 % | 51.3 % | 79.0 % | 18.2 % | 40.1 % | 481 | 595 | 6654 | 86245 | 0.5 fps |
| TLMHT [42] | 48.7 % | 55.3 % | 76.4 % | 15.7 % | 44.5 % | 413 | 642 | 6632 | 86504 | 4.8 fps |
| GCRA [43] | 48.2 % | 48.6 % | 77.5 % | 12.9 % | 41.1 % | 821 | 1117 | 5104 | 88586 | 2.8 fps |
| FWT [44] | 47.8 % | 44.3 % | 75.5 % | 19.2 % | 38.2 % | 852 | 1534 | 8886 | 85487 | 0.6 fps |
| NLLMPa [45] | 47.6 % | 47.3 % | 78.5 % | 17.0 % | 40.4 % | 629 | 768 | 5844 | 89093 | 8.3 fps |
| EAGS16 [42] | 47.4 % | 50.1 % | 75.9 % | 17.3 % | 42.7 % | 575 | 913 | 8369 | 86931 | 197.3 fps |
| MHT DAM [27] | 45.8 % | 46.1 % | 76.3 % | 16.2 % | 43.2 % | 590 | 781 | 6412 | 91758 | 0.8 fps |
| INTERA_MOT [46] | 45.4 % | 47.7 % | 74.4 % | 18.1 % | 38.7 % | 600 | 930 | 13407 | 85547 | 4.3 fps |
| QuadMOT16 [22] | 44.1 % | 38.3 % | 76.4 % | 14.6 % | 44.9 % | 745 | 1096 | 6388 | 94775 | 1.8 fps |
| online |        |        |        |      |     |      |     |     |     |     |
| MOTDT [34] | 47.6 % | 50.9 % | 74.8 % | 15.2 % | 38.3 % | 792 | 1858 | 9253 | 85431 | 20.6 fps |
| AMIR [16] | 47.2 % | 46.3 % | 75.8 % | 14.0 % | 41.6 % | 774 | 1675 | 2681 | 92856 | 1.0 fps |
| DMMOT [24] | 46.1 % | 54.8 % | 73.8 % | 17.4 % | 42.7 % | 532 | 1616 | 7909 | 89874 | 0.3 fps |
| STAM16 [23] | 46.0 % | 50.0 % | 74.9 % | 14.6 % | 43.6 % | 473 | 1422 | 6895 | 91117 | 0.2 fps |
| RAR16pub [28] | 45.9 % | 48.8 % | 74.8 % | 13.2 % | 41.9 % | 648 | 1992 | 6871 | 91173 | 0.9 fps |
| MTF17 [47] | 46.2 % | 45.7 % | 72.6 % | 14.1 % | 36.4 % | 1987 | 3377 | 12018 | 84970 | 0.5 fps |
| CDA_DDALv2 [15] | 43.9 % | 45.1 % | 74.7 % | 10.7 % | 44.4 % | 676 | 1795 | 6450 | 95175 | 0.5 fps |
| PHD_GSDL16 [48] | 41.0 % | 43.1 % | 75.9 % | 11.3 % | 41.5 % | 1810 | 3650 | 6498 | 99257 | 8.3 fps |
| AM_ADM [49] | 40.1 % | 43.8 % | 75.4 % | 7.1 % | 46.2 % | 789 | 1736 | 8503 | 99891 | 5.8 fps |
| Ours(Deep-TAMA) | 46.2 % | 49.4 % | 75.4 % | 14.1 % | 44.0 % | 598 | 1127 | 5126 | 92367 | 2.0 fps |

TABLE III: Tracking performance comparison on MOT16 benchmark table. Bolded texts indicate, red : best performance among offline trackers, blue : best performance among online trackers, respectively. Best viewed in color. Accessed at 2019.03.10.
that our temporal appearance matching association model works fine. \cite{53} used re-id and Mask-RCNN \cite{53} feature to re-score candidates. We assume that state-of-the-art semantic segmentation network contributed to filter out potential false-positives. This gap becomes smaller in following MOT17 benchmark.

**MOT17**: In MOT17 benchmark, Table \ref{table:v} Deep-TAMA shows better performance, way more closer to the top performing tracker. We selected \cite{53} as the minimum performance boundary. It simply used IOU to expand tracks. Many trackers are duplicates of those in Table \ref{table:iii}. Ours shows better MOTA and lower IDF1 than \cite{24} similar to Table \ref{table:iii} \cite{21} appended bi-LSTM to the multiple hypothesis tracking framework to get reliable appearance and motion similarity. In most measures, our tracker outperforms. Benefits of Deep-TAMA compared to siamese network is proved again here. Ours shows coherently better performance than \cite{9} in previous comparison. Same as MOT16 benchmark, the best performing online tracker is \cite{34}. However, it is clear to see that performance gap between ours and \cite{24} became negligible compared to MOT16 benchmark. Ours shows even better results in IDF1, MT and IDs and FM. We guess the reason is that MOT17 contains two near state-of-the-art detections and so effect of mask-RCNN module in \cite{24} diminished. As a result, detection fine-tuning issue got relieved. So, we carefully estimate that MOT17 better reflects the actual performance of tracker than 2DMOT2015 or MOT16 benchmarks.

We also have published a tracking result on MOT17 benchmark from our conference paper \cite{9}. It shares the similar appearance model, Eq. \ref{eq:14}, \ref{eq:15}, with C-TAMA. To highlight the improvement from it, exclusive comparison is provided in Table \ref{table:v}. There are two points, worth to be focused on. First, balance of FP and FN has been improved. This is because \cite{9} adopted IOU matching based initialization instead of hierarchical initialization. Second, overall tracking performance has been improved except IDs. The LSTM network found better weights for association and this contributed on tracking performance.

### VI. Conclusion

In this paper, we consider a problem of conventional feature extraction method for multiple pedestrian tracking. We tried to solve the problem using Joint-Inference network. We employ a Joint-Inference network as a backbone and improved it with the proposed temporal appearance matching association methods in two different perspectives, hand-crafted and data-driven approaches. So we proposed temporal appearance matching association methods in two different perspectives: hand-crafted and data-driven methods. Particularly in data-driven method, the width and height of the bounding box are proposed as part of appearance information. Our tracker showed state-of-the-art performance in public benchmark tables. Current limitation of our appearance model is restricted in the small bounding box areas. As context propagation has improved the performance of recent detectors, our future work will be directed to extraction of feature map that having context of the scene instead of the raw images.

| Tracker        | MOTA↑ | IDF1↑ | MOTP↑ | MT↑ | ML↓ | IDs↓ | FM↓ | FP↓ | FN↓ | Hz↑ |
|----------------|-------|-------|-------|-----|-----|------|-----|-----|-----|-----|
| Ours(Deep-TAMA)| 50.3  | 53.5  | 76.7  | 19.2 | 37.5 | 2192 | 3978 | 25479| 252996| 1.5 fps |
| HAM_SADF17 [9] | 48.3  | 51.1  | 77.2  | 17.1 | 35.6 | 3998 | 8886 | 23199| 265954| 6.7 fps |
| Ours(Deep-TAMA)| 50.3  | 53.5  | 76.7  | 19.2 | 37.5 | 2192 | 3978 | 25479| 252996| 1.5 fps |

**TABLE V**: Tracking performance comparison with our conference paper on MOT17 benchmark.

**TABLE IV**: Tracking performance comparison on MOT17 benchmark table. Bolded texts indicate, **black**: trackers which focused on deep appearance modeling, **red**: best performance among offline trackers, **blue**: best performance among online trackers, respectively. Accessed at 2019.03.10.
(a) Occlusion when two targets are overlapped. Two targets have similar bounding-box size and located on same y-axis. We suppose target-A and B as a person moving right to left and a static person wearing white coat respectively. (Top, JI-Net) : Target-A is originally assigned a number 11. During occlusion, bounding-box contains appearance information of both Target-A and B (frame 143). At frame 148, number 11 is wrongly assigned to Target-B. Target-A is initialized to a new number 12. (Bottom, Deep-TAMA) : Similarly, bounding-box includes both Target-A and B at frame 143. But, different from JI-Net, it successfully tracked Target-A with number 9 and re-matched Target-B to number 10, its original number before occlusion.

(b) The target is occluded by a scene obstacle like street light. Target is almost fully occluded by a signboard on street light at frame 24 and 35. (Top, JI-Net) : The target is originally assigned a number 2. But it is missed when occluded and assigned a new number 6 at frame 45. Number 2 is wrongly assigned to bounding box, located on a signboard. This result strongly proves that JI-Net itself lacks a smoothing capability. (Bottom, Deep-TAMA) : The target is originally assigned a number 3. It is tracked successfully at both frame 24 and 35. Although number 3 bounding-box includes appearance of signboard, in contrast to JI-Net, it is successfully assigned to a correct target at frame 45.

(c) Two targets, number 2 (right side) and 9 (left side), suffer occlusions multiple times. Although two targets are very small so are fully obscured during each occlusion, both targets are reliably tracked until frame 151. We can confirm the power of our long-term association model here.

Fig. 15: Qualitative examples showing good performance in ID-consistency. (a,b) : Tracking quality comparison in challenging situation between two different similarity models (JI-Net, Deep-TAMA). (c) : Successful long-term target tracking with Deep-TAMA.
ACKNOWLEDGMENT

This research was financially supported in part by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (Development of global multi-target tracking and event prediction techniques based on real-time large-scale video analysis) Grant No. B0101-15-0525 and also by the Guangwu Institute of Science and Technology (GIST) (Autonomous Vehicle project). Du Yong Kim was supported by Vice-Chancellor’s Research Fellowship, RMIT University.

REFERENCES

[1] R. E. Kalman, “A new approach to linear filtering and prediction problems,” Journal of Basic Engineering, vol. 82, 1960.

[2] P. Del Moral, “Non linear filtering: Interacting particle solution,” Markov Processes and Related Fields, vol. 2, no. 4, 1996.

[3] R. Mahler, “Multitarget bayes filtering via first-order multitarget moments,” IEEE Transactions on Aerospace and Electronic Systems, Nov., vol. 39, no. 4, 2003.

[4] V. Takala and M. Pietikinen, “Multi-object tracking using color, texture and motion,” in CVPR, 2007.

[5] B. Wang, G. Wang, K. L. Chan, and L. Wang, “Tracklet association by online target-specific metric learning and coherent dynamics estimation,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 7, 2017.

[6] J. H. Yoon, C.-R. Lee, M.-H. Yang, and K.-J. Yoon, “Structural constraint data association for online multi-object tracking,” International Journal of Computer Vision, 2019.

[7] L. Leal-Taixe, C. Canton-Ferrer, and K. Schindler, “Learning by tracking: Siamese cnn for robust target association,” in DeepVision workshop in conjunction with CVPR, 2016.

[8] S. Tang, M. Andriluka, B. Andres, and B. Schiele, “Multiple people tracking by lifted multicut and person re-identification,” CVPR, pp. 3701–3710, 2017.

[9] Y.-C. Yoon, A. Boragule, Y. min Song, K. Yoon, and M. Jeon, “Online multi-object tracking with historical appearance matching and scene adaptive detection filtering,” in AVSS, 2018.

[10] J. H. Yoon, M.-H. Yang, J. Lim, and K.-J. Yoon, “Bayesian multi-object tracking using motion context from multiple objects,” in WACV, 2015.

[11] A. Bewley, L. Ott, F. Ramos, and B. Upcroft, “ALEXTRAC: Affinity Learning by Exploring Temporal Reincarnation within Association Chains,” in ICRA, 2016.

[12] A. Milan, S. Rezatofighi, A. Dick, I. Reid, and K. Schindler, “Online multi-target tracking using recurrent neural networks,” in AAAI, 2017.

[13] A. Milan, S. H. Rezatofighi, R. Garg, A. Dick, and I. Reid, “Data-driven approximations to NP-hard problems,” in AAAI, February 2017.

[14] H. Rezatofighi, A. Milan, Z. Zhang, Q. Shi, A. Dick, and I. Reid, “Joint probabilistic data association revisited,” in ICCV, 2015.

[15] S.-H. Bae and K.-J. Yoon, “Confidence-based data association and discriminative deep appearance learning for robust online multi-object tracking,” IEEE transactions on pattern analysis and machine intelligence, vol. 40, no. 3, 2018.

[16] A. Sadeghian, A. Alahi, and S. Savarese, “Tracking the untrackable: Learning to track multiple cues with long-term dependencies,” in ICCV, 2017.

[17] H. W. Khun, “The hungarian method for the assignment problem,” in Naval Research Logistics, 1955.

[18] W. Choi, “Near-online multi-target tracking with aggregated local flow descriptor,” in ICCV, 2015.

[19] L. Wang, L. Xu, M. Y. Kim, L. Rigazico, and M.-H. Yang, “Online multiple object tracking via flow and convolutional features,” in ICP, 2017.

[20] F. Yang, W. Choi, and Y. Lin, “Exploit all the layers: Fast and accurate cnn object detector with scale dependent pooling and cascaded rejection classifiers,” in CVPR, 2016.

[21] C. Kim, F. Li, and J. Rehg, “Multi-object Tracking with Neural Gating Using Bilinear LSTM,” in ECCV, 2018.

[22] J. Son, M. Baek, M. Cho, and B. Han, “Multi-object tracking with interacting tracklets,” in CVPR, 2017.

[23] Q. Chu, W. Ouyang, H. Li, X. Wang, B. Liu, and N. Yu, “Online multi-object tracking using cnn-based single object tracker with spatial-temporal attention mechanism,” in ICCV, 2017.

[24] J. Zhu, H. Yang, N. Liu, M. Kim, W. Zhang, and M.-H. Yang, “Online Multi-Object Tracking with Dual Matching Attention Networks,” in ECCV, 2018.

[25] Z. He, J. Li, D. Liu, H. He, and D. Barber, “Tracking by animation: Unsupervised learning of multi-object attentive trackers,” in CVPR, 2019.

[26] B.-N. Vo and W.-K. Ma, “The gaussian mixture probability hypothesis density filter,” IEEE Transactions on signal processing, vol. 54, no. 11, 2006.

[27] C. Kim, F. Li, A. Ciptadi, and J. Rehg., “Multiple Hypothesis Tracking Revisited,” in ICCV, 2015.

[28] K. Fang, Y. Xiang, X. Li, and S. Savarese, “Recurrent autoregressive networks for online multi-object tracking,” in WACV, 2018.

[29] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” Neural Computation, vol. 9, no. 8, 1997.

[30] Y. Lu, C. Lu, and C.-K. Tang, “Online video object detection using association lstm,” in ICCV, 2017.

[31] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W. kin Wong, and W. Chun Woo, “Convolutional lstm network: a machine learning approach for precipitation nowcasting,” in NIPS, 2015.

[32] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” in Advances in Neural Information Processing Systems (NIPS), 2015.

[33] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and D. Saghir, “Sod: Single shot多obj detector,” in ECCV, 2016.

[34] C. Long, A. Haizhou, Z. Zijie, and S. Chong, “Real-time multi-person tracking people tracking with deeply learned candidate selection and person re-identification,” in ICME, 2018.

[35] A. Vedaldi and K. Lenc, “Matconvnet - convolutional neural networks for matlab,” in arXiv:1412.2335.

[36] L. Leal-Taixe, A. Milan, I. Reid, S. Roth, and K. Schindler, “Motchallenge 2015: Towards a benchmark for multi-target tracking,” in arXiv:1504.01942.

[37] A. Milan, L. Leal-Taixe, I. Reid, S. Roth, and K. Schindler, “Mot16: A benchmark for multi-object tracking,” in arXiv:1603.00831.

[38] P. Dendorfer, H. Rezatofighi, A. Milan, J. Shi, D. Cremers, I. Reid, S. Roth, K. Schindler, and L. Leal-Taixe, “Cvpr2017 and tracking and detection challenge: How crowded can it get?”, in arXiv:1906.04567.

[39] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” in ICML, 2015.

[40] R. Stiefelhagen, K. Bernardin, R. Bowers, J. S. Garofolo, D. Mostefa, and P. Soundararajan, “The clear 2006 evaluation,” in CLEAR, 2006.

[41] E. Ristani, F. Solera, R. Zou, B. Cucchiara, and C. Tomasi, “Performance measures and a data set for multi-target, multi-camera tracking,” in BMTT workshop in conjunction with ECCV, 2016.

[42] H. Sheng, J. Chen, Y. Zhang, W. Ke, Z. Xiong, and J. Yu, “Iterative multiple hypothesis tracking with tracklet-level association,” IEEE Transactions on Circuits and Systems for Video Technology, 2018.

[43] C. Ma, C. Yang, F. Yang, Y. Zhuang, Z. Zhang, H. Jia, and X. Xie, “Trajectory factory: Representing and re-connection by deep siamese bi-gru for multiple object tracking,” in ICME, 2018.

[44] R. Henschel, L. Leal-Taix, D. Cremers, and B. Rosenhahn, “Fusion of Head and Full-Body Detectors for Multi-Object Tracking,” in CVPRW, 2018.

[45] E. Levinkov, S. Tang, E. Insafutdinov, and B. Andres, “Joint graph decomposition and node labelling by local search,” in CVPR, 2017.

[46] L. Lan, X. Wang, S. Zhang, D. Tao, W. Gao, and T. S. Huang, “Interacting tracklets for multi-object tracking,” IEEE Transactions on Image Processing, 2018.

[47] Z. Fu, F. Angelini, J. Chambers, and D. M. Naqvi, “Multi-level cooperative fusion of gm-phi filters for online multiple human tracking,” in IEEE Transactions on Multimedia, 2019.

[48] Z. Fu, P. Feng, F. Angelini, J. A. Chambers, and S. M. Naqvi, “Particle phi filter based multiple human tracking using online group-structured dictionary learning,” in IEEE Access, vol. 6, 2018.

[49] S.-H. Lee, M.-Y. Kim, and S.-H. Bae, “Learning discriminative appearance models for online multi-object tracking with appearance discriminability measures,” in IEEE Access, 2018.

[50] H. Sheng, Y. Zhang, J. Chen, and Z. Xiong, “Heterogeneous association graph fusion for target association in multiple object tracking,” IEEE Transactions on Circuits and Systems for Video Technology, 2018.

[51] M. Keuper, S. Tang, B. Andres, T. Brox, and B. Schiele, “Motion segmentation & multiple object tracking by correlation co-clustering,” in IEEE Transactions on Pattern Analysis and Machine Intelligence, 2018.

[52] J. Chen, H. Sheng, Y. Zhang, and Z. Xiong, “Enhancing detection model for multiple hypothesis tracking,” in BMTT-Workshop in conjunction with CVPR, 2018.
[53] E. Bochinski, V. Eiselein, and T. Sikora, “High-speed tracking-by-detection without using image information,” in *International Workshop on Traffic and Street Surveillance for Safety and Security at IEEE AVSS*, 2017.

[54] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask-rcnn,” in *ICCV*, 2017.