Online Multi-Target Tracking by Large Margin Structured Learning

Suna Kim, Suha Kwak, Jan Feyereisl, and Bohyung Han

Department of Computer Science and Engineering
POSTECH, Korea

Abstract. We present an online data association algorithm for multi-object tracking using structured prediction. This problem is formulated as a bipartite matching and solved by a generalized classification, specifically, Structural Support Vector Machines (S-SVM). Our structural classifier is trained based on matching results given the similarities between all pairs of objects identified in two consecutive frames, where the similarity can be defined by various features such as appearance, location, motion, etc. With an appropriate joint feature map and loss function in the S-SVM, finding the most violated constraint in training and predicting structured labels in testing are modeled by the simple and efficient Kuhn-Munkres (Hungarian) algorithm in a bipartite graph. The proposed structural classifier can be generalized effectively for many sequences without re-training. Our algorithm also provides a method to handle entering/leaving objects, short-term occlusions, and misdetections by introducing virtual agents—additional nodes in a bipartite graph. We tested our algorithm on multiple datasets and obtained comparable results to the state-of-the-art methods with great efficiency and simplicity.

1 Introduction

Multi-target tracking is a challenging but interesting problem in computer vision, and has significant potential to be applied to various high-level applications such as visual surveillance, scene understanding, event detection, and many others. Recently, tracking-by-detection framework [1–14] has been widely used due to recent advances in object detection techniques [15–17], and multi-target tracking is often formulated as a data association problem across multiple frames. The data association is relatively easy when targets are isolated and discriminative, but its complexity increases rapidly in the presence of misdetections, false alarms, occlusions, objects with similar appearance, and so on. Also, it is not straightforward to handle entering and leaving objects although object detection algorithms provide useful clues for these problems.

In tracking-by-detection techniques, tracking performance depends on detection accuracy, and challenges from imperfect detections are often dealt with systematically by global optimization such as binary quadratic programming [1], integer linear programming [2, 3], and finding min-cost flow on the cost-flow network [4, 5]. Persistent tracking has been achieved by gradually merging consistent

tracklets—trajectory fragments—into long lasting tracks through hierarchical tracklet association based on the Kuhn-Munkres (Hungarian) algorithm [6–9], Conditional Random Fields (CRF) [10, 12], and maximum weight independent set [11]. However, many of these methods are limited to *offline* batch processing [3, 5, 10, 11] although many tracking applications have *online* requirement,[1] or they assume the existence of reliable tracklets with insufficient justification [6–11]. Online learning is sometimes employed to learn scene specific models adaptively as in [8, 9, 12], but these methods do not necessarily track objects in an online manner. Only a few online multi-target tracking algorithms have been proposed within the tracking-by-detection framework [18, 19].

Finding robust associations relies on appropriate representation of the detected target objects. For this reason, various affinity measures between a pair of detections have been investigated in the past. An affinity is traditionally measured by the distances between the descriptors of detections [1–5]. For example in [11], the affinity is defined by the Mahalanobis distance between descriptors, which is updated adaptively by online metric learning using observed discriminativeness amongst targets during tracking. Affinity is often determined by jointly solving ranking and classification of associations [7], learning the discriminativeness among targets online [8], or online weight learning for weak classifiers trained offline [9]. However, the definition of affinity may not be sufficiently rigorous especially when multiple features are incorporated and the dependency between features are difficult to be identified and modeled properly. Also, affinity measures sometimes involve several parameters, and tuning such parameters with multiple features may not be straightforward.

Overview of our algorithm is illustrated in Figure 1. We pose multi-target tracking as a data association problem, which is solved by structural prediction. Our algorithm learns the best matching between two sets of targets extracted from two consecutive frames, and applies the trained model to new data. The structured prediction in a bipartite matching is performed by Structural Support Vector Machines (S-SVM) [20], which allows for robust modeling of the complex dependency between detector responses and their corresponding associations. The proposed algorithm based on the S-SVM has the following characteristics:

- We provide an online algorithm to track multiple objects through data association based on the S-SVM without the construction of tracklets.
- The data association is achieved by the structured prediction in the S-SVM framework, which is formulated and solved by the Hungarian algorithm [21]. The parameters for the affinity measures between a pair of targets are learned automatically; the dependency of multiple features is handled naturally.
- Through the structural prediction, we manage various challenges from misdetection, false alarms, occlusions, and entering/leaving objects, effectively.

[1] In this paper, *online* algorithm refers to a technique that processes inputs in a sequential manner with no (or little) requirement for future data while *offline* method requires the entire data or a significant amount of future data.

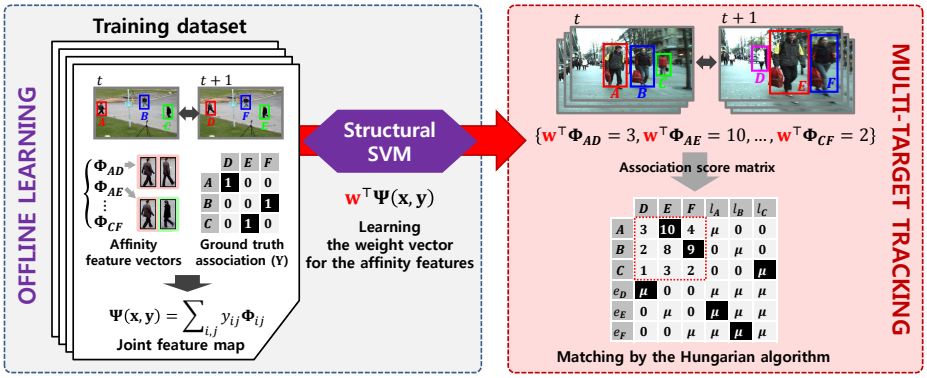


Fig. 1. Overview of our multi-target tracking framework by structural prediction. We learn the matching structure and the inference for testing is optimized by the Hungarian algorithm in a bipartite graph. Note that we can handle entering/leaving and missing/reappearing objects by introducing virtual nodes in the graph.

- Our structural classifier trained offline can generalize to many sequences without re-training. Also, the proposed technique is very fast since there is no online training and the testing procedure is simple.

For the rest of this paper, we first introduce the S-SVM and describe how our multi-target tracking algorithm is formulated by the S-SVM in Section 2 and 3, respectively. In Section 4, we evaluate our algorithm by testing it with multiple sequences and comparing it with existing multi-target tracking techniques.

2 Structural Support Vector Machines

Many existing methods in computer vision are based on binary classification, in which discrimination between two classes is sought after by learning a function $\mathcal{F} : \mathcal{X} \rightarrow \mathcal{Y}$, mapping the inputs $\mathbf{x} \in \mathbb{R}^n$ in $\mathcal{X}$ to binary outputs $y \in \{-1, +1\}$ in $\mathcal{Y}$. However, such methods do not consider the more natural but non-trivial structure of the output $\mathbf{y} \in \mathbb{R}^m$ in $\mathcal{Y}$.

Structural learning, on the other hand, can consider the structure in the output space $\mathcal{Y}$; the output variable is not a scalar any more and can be modeled by a vector $\mathbf{y}$ instead. In particular, the S-SVM [20] is a generalization of the SVM paradigm, which allows for efficient predictions of vector-valued structured outputs. The prediction becomes a task of correctly assigning the input $\mathbf{x}$ to the structured output based on a scoring function $\mathcal{F}_{\mathbf{w}}(\mathbf{x}, \mathbf{y}) = \mathbf{w}^{\mathsf{T}}\boldsymbol{\Psi}(\mathbf{x}, \mathbf{y})$, where $\boldsymbol{\Psi} : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}^r$ is a joint feature function, extracting a feature vector from a given input and output pair. Thus, we seek to find the highest scoring structure $\mathbf{y}$ amongst all possible structures in $\mathcal{Y}$ based on the trained weight vector $\mathbf{w}$ as

$$\mathcal{H}_{pred}(\mathbf{x}) = \arg\max_{\mathbf{y} \in \mathcal{Y}} \mathcal{F}_{\mathbf{w}}(\mathbf{x}, \mathbf{y}) = \arg\max_{\mathbf{y} \in \mathcal{Y}} \mathbf{w}^{\mathsf{T}}\boldsymbol{\Psi}(\mathbf{x}, \mathbf{y}). \tag{1}$$

For a set of N training examples denoted by $(\mathbf{x}_i, \mathbf{y}_i) \in \mathcal{X} \times \mathcal{Y}$ $(i = 1, \ldots, N)$, the S-SVM solves the optimization problem to find $\mathbf{w}$, which is given by

$$\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{N}\sum_{i=1}^{N}\xi_i, \tag{2}$$

$$\text{s.t.} \;\; \mathbf{w}^{\mathsf{T}}\boldsymbol{\Psi}(\mathbf{x}_i, \mathbf{y}_i) - \mathbf{w}^{\mathsf{T}}\boldsymbol{\Psi}(\mathbf{x}_i, \hat{\mathbf{y}}) \geq \Delta(\mathbf{y}_i, \hat{\mathbf{y}}) - \xi_i, \;\; \forall i, \; \hat{\mathbf{y}} \neq \mathbf{y}_i, \;\; \forall i, \; \xi_i \geq 0$$

where ξ_i is a slack variable and C is a constant. The constraint means that the score for the correct output $\mathbf{w}^{\mathsf{T}}\boldsymbol{\Psi}(\mathbf{x}_i, \mathbf{y}_i)$ should be greater than the score $\mathbf{w}^{\mathsf{T}}\boldsymbol{\Psi}(\mathbf{x}_i, \hat{\mathbf{y}})$ for all alternative outputs $\hat{\mathbf{y}}$ at least by the loss over slack variable $\Delta(\mathbf{y}_i, \hat{\mathbf{y}}) - \xi_i$.

Given the S-SVM formulation in Eq. (2), we employ the so-called *cutting-plane* algorithm to compute the optimal $\mathbf{w}$ efficiently in training[2] even when the number of elements in $\mathcal{Y}$ is very high. The algorithm is composed of two steps—solving a quadratic optimization problem and identifying the most violated constraint. The first sub-problem is typically solved efficiently, but the second one requires an efficient inference technique designed for a specific problem. Note that the identification of the most violated constraint involves *loss-augmented prediction*, which resembles structured prediction in testing phase, so we need only a single inference technique that can be shared in both training and testing.

Although the generic nature of structural SVMs allows applicability to a wide variety of problems, the following set of custom implementations are required for each unique problem; we present how each component is designed in our multi-target tracking problem in the next section.

Representation Joint feature map $\boldsymbol{\Psi}(\mathbf{x}, \mathbf{y})$,
Quality Measure Loss function $\Delta(\mathbf{y}, \hat{\mathbf{y}})$,
Optimization Efficient inference technique in training and prediction.

3 Multi-Target Tracking by Structural SVM

Denote by $D_t = \{d_1^t, \ldots, d_{n_t}^t\}$ a set of detected objects of interest, where n_t is the number of an object at frame t. In the traditional SVM for data association, a feature vector $\boldsymbol{\Phi}_{ij}$[3] is constructed between each pair of detections d_i^t and d_j^{t+1} based on a set of k similarity measures, $S = \{s_1, \ldots, s_k\}$, which is given by

$$\boldsymbol{\Phi}_{ij} = \left[\mathbf{f}_{s_1}(d_i^t, d_j^{t+1})^{\mathrm{T}} \;\; \ldots \;\; \mathbf{f}_{s_k}(d_i^t, d_j^{t+1})^{\mathrm{T}}\right]^{\mathrm{T}}, \quad (1 \leq i \leq n_t, \; 1 \leq j \leq n_{t+1}) \tag{3}$$

where $\mathbf{f}_{s_k}$ is a vector-valued similarity function based on an affinity measure s_k—for example, similarity in color, texture, motion, etc. If a binary classifier is applied to each feature vector independently as

$$y_{ij} = \mathcal{F}_{\mathbf{w}}(\boldsymbol{\Phi}_{ij}), \tag{4}$$

[2] It can find an ϵ-accurate solution by considering only $\mathcal{O}(1/\epsilon)$ constraints rather than the whole constraint space [20].

[3] The time indices are omitted for the simplicity of notation.

the prediction procedure is very simple but may incur inconsistent matching structure such as duplicate or missing associations.

To avoid such inconsistencies and achieve the global optimum between a pair of frames, we present how the S-SVM formulates the data association problem for multi-target tracking. Specifically, we describe the details about a set of custom design decisions outlined in the previous section—joint feature map, loss function, and inference technique. Also, we discuss the practical solutions to handle entering/leaving and missing/reappearing objects and occlusions.

3.1 Joint Feature Map

To learn the structure of data association between two consecutive frames, we construct a feature vector between a pair of frames instead of a pair of detections. We define an augmented feature matrix, which is a collection of the feature vectors in Eq. (3) between all pairs of detections as

$$\mathbf{X} = \begin{pmatrix} \boldsymbol{\Phi}_{11}^{\mathrm{T}} & \boldsymbol{\Phi}_{12}^{\mathrm{T}} & \cdots & \boldsymbol{\Phi}_{1n_{t+1}}^{\mathrm{T}} \\ \boldsymbol{\Phi}_{21}^{\mathrm{T}} & \boldsymbol{\Phi}_{22}^{\mathrm{T}} & \cdots & \boldsymbol{\Phi}_{2n_{t+1}}^{\mathrm{T}} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{\Phi}_{n_t1}^{\mathrm{T}} & \boldsymbol{\Phi}_{n_t2}^{\mathrm{T}} & \cdots & \boldsymbol{\Phi}_{n_tn_{t+1}}^{\mathrm{T}} \end{pmatrix}. \tag{5}$$

The corresponding augmented label matrix—structured output matrix—is a binary matrix denoted by

$$\mathbf{Y} = \begin{pmatrix} y_{11} & y_{12} & \cdots & y_{1n_{t+1}} \\ y_{21} & y_{22} & \cdots & y_{2n_{t+1}} \\ \vdots & \vdots & \ddots & \vdots \\ y_{n_t1} & y_{n_t2} & \cdots & y_{n_tn_{t+1}} \end{pmatrix}, \tag{6}$$

where

$$y_{ij} = \begin{cases} 1 & \text{if matching is positive} \\ 0 & \text{otherwise} \end{cases}, \tag{7}$$

and $\mathbf{Y}$ should have no more than one non-zero element in each row and column conceptually. To make our notation consistent with Section 2, we also use the vectorized representations of $\mathbf{X}$ and $\mathbf{Y}$ denoted by $\mathbf{x}$ and $\mathbf{y}$, respectively.

The joint feature map $\boldsymbol{\Psi}(\mathbf{x}, \mathbf{y})$ combines augmented feature and label matrices in vector forms and creates a generalized feature vector. One critical challenge in the design of the joint feature map is that the number of detections in each frame may be different and the sizes of $\mathbf{X}$ and $\mathbf{Y}$ are different consequently. Note that our joint feature vector should have a fixed dimensionality regardless of the sizes of augmented feature and label matrices to train the weight vector $\mathbf{w}$. In our setting, the joint feature map is defined by

$$\boldsymbol{\Psi}(\mathbf{x}, \mathbf{y}) = \sum \boldsymbol{\Phi}_{+} = \sum_{ij} y_{ij} \boldsymbol{\Phi}_{ij}, \tag{8}$$

where $\mathbf{\Phi}_+$ is a set of feature vectors for positive matchings only. The joint feature map characterizes the data association problem between a pair of frames using multi-dimensional (multi-aspect) similarities of positive matchings. According to the definition, the joint feature map is a linear combination of the features extracted from all pairs of detections between two frames, so its dimensionality is fixed in spite of variable number of detections in each frame. Also, it provides a space that encodes all possible combinations of correct matchings of objects encountered during training.

3.2 Loss Function

In order to be able to assess the quality of a prediction, an appropriate loss function $\Delta : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}$ is necessary. Thus, $\Delta(\mathbf{y}, \hat{\mathbf{y}})$ denotes the loss associated with a prediction $\hat{\mathbf{y}}$ when the true value is $\mathbf{y}$. The loss should decrease as the prediction $\hat{\mathbf{y}}$ approaches the ground truth $\mathbf{y}$. In our framework, we define a loss function as

$$\Delta(\mathbf{y}_i, \hat{\mathbf{y}}) = (\mathbf{1} - \mathbf{y}_i)^\mathsf{T} \hat{\mathbf{y}}, \tag{9}$$

where $\mathbf{1}$ is a vector with all elements 1. Note that this is slightly different from the Hamming distance because the penalty is given to false positive matchings only but not to true negatives. We choose this loss function because it makes the optimization procedure in training and prediction efficient; the details will be discussed in Section 3.3. However, it is still similar to the Hamming distance, a frequently used loss function, in the sense that the number of false positives is strongly correlated to the number of true negatives in matching problems.

3.3 Optimization

We require efficient inference techniques as part of the cutting-plane algorithm in training and structured prediction in testing. In our case, these are in fact identical. Inference in testing and training is presented in this order since the optimization procedures in the two stages are related and the procedure in testing is simpler.

Testing During prediction, we solve the following optimization problem,

$$\mathcal{H}_{\mathsf{pred}}(\mathbf{x}) = \arg\max_{\mathbf{y} \in \mathcal{Y}} \ \mathcal{F}_{\mathbf{w}}(\mathbf{x}, \mathbf{y}) = \arg\max_{\mathbf{y} \in \mathcal{Y}} \ \mathbf{w}^\mathsf{T} \mathbf{\Psi}(\mathbf{x}, \mathbf{y}), \tag{10}$$

which involves a simple linear combination of the joint features with a set of parameters in $\mathbf{w}$ learned from the S-SVM. However, even with the existence of a joint feature map $\mathbf{\Psi}(\mathbf{x}, \mathbf{y})$, the above inference problem can become very expensive as the number of outputs may grow exponentially with the number of tracked objects, and an exhaustive search might not always be possible. For this reason, the exploitation of the underlying structure is necessary for efficient

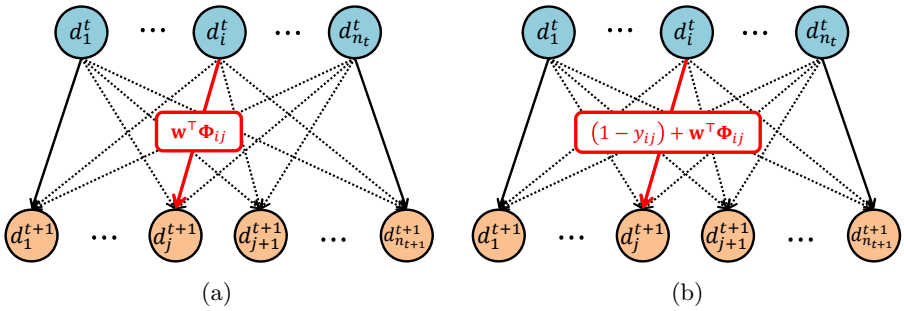


Fig. 2. Illustration of the solution for (a) Eq. (10) and (b) Eq. (13). Note that finding the most violated constraint in training and predicting the structured output in testing have the same formulation; they can be solved by the Hungarian algorithm for maximum weight matching in a bipartite graph with slightly different weights on edges. The solid and dotted edges in the above figures denote matched and unmatched ones, respectively, which are encoded in the output variable $\mathbf{y}$.

maxmization of Eq. (10). We claim that our problem is equivalent to the maximum weight matching in a bipartite graph and that the inference can be solved using the Kuhn-Munkres (Hungarian) algorithm [21], as shown in the following equations:

$$\begin{aligned} \mathcal{F}_{\mathbf{w}}(\mathbf{x}, \mathbf{y}) &= \mathbf{w}^\top \mathbf{\Psi}(\mathbf{x}, \mathbf{y}) = \mathbf{w}^\top \left(\sum_{ij} y_{ij} \mathbf{\Phi}_{ij} \right) \\ &= \sum_{ij} y_{ij} \mathbf{w}^\top \mathbf{\Phi}_{ij} \\ &= \left[\mathbf{w}^\top \mathbf{\Phi}_{11} \dots \mathbf{w}^\top \mathbf{\Phi}_{n_t n_{t+1}} \right] \mathbf{y}. \end{aligned} \tag{11}$$

Note that the function in Eq. (11) is based on the inner product of two vectors. We can interpret the first and second vector as the weight of edges in a bipartite graph and the selection of edges to maximize the sum of weights, respectively, as illustrated in Figure 2(a). Considering the original binary matrix form Eq. (6) of $\mathbf{y}$, under the constraints $\sum_i \mathbf{Y}_{ij} = 1$ and $\sum_j \mathbf{Y}_{ij} = 1$, the optimization problem in Eq. (10) is equivalent to finding the maximum weight matching that can be solved efficiently in polynomial time by the Hungarian algorithm.

Training To compute the weight vector $\mathbf{w}$ by efficiently solving the optimization problem in Eq. (2), we employ the cutting-plane algorithm in the *margin-rescaling* formulation. This algorithm first finds the most violated constraint in Eq. (2), if the violation over the slack variable ξ is larger than the threshold, the constraint is added to a working set, over which the quadratic program is solved to find the current weight vector and slack variable in the next iteration. This

is repeated until convergence when the working set remains unchanged. We find the most violated constraint by solving the following optimization step, called the *loss-augmented prediction* step,

$$\mathcal{H}_{\mathsf{train}}(\mathbf{x}) = \arg\max_{\hat{\mathbf{y}} \in \mathcal{Y}} \quad \Delta(\mathbf{y}_i, \hat{\mathbf{y}}) - \mathbf{w}^{\mathsf{T}}\boldsymbol{\Psi}(\mathbf{x}_i, \mathbf{y}_i) + \mathbf{w}^{\mathsf{T}}\boldsymbol{\Psi}(\mathbf{x}_i, \hat{\mathbf{y}}) \quad (\forall i), \tag{12}$$

where $\mathbf{w}^{\mathsf{T}}\boldsymbol{\Psi}(\mathbf{x}_i, \mathbf{y}_i)$ is a constant that can be ignored in optimization. If the loss function in Eq. (12) is replaced by Eq. (9), the loss-augmented objective function is obtained as

$$\begin{aligned} \mathcal{H}_{\mathsf{train}}(\mathbf{x}) &= \arg\max_{\hat{\mathbf{y}} \in \mathcal{Y}} \ (\mathbf{1} - \mathbf{y}_i)^{\mathsf{T}}\hat{\mathbf{y}} + \left[\mathbf{w}^{\mathsf{T}}\boldsymbol{\Phi}_{11} \ \ldots \ \mathbf{w}^{\mathsf{T}}\boldsymbol{\Phi}_{n_t n_{t+1}}\right]\hat{\mathbf{y}} \\ &= \arg\max_{\hat{\mathbf{y}} \in \mathcal{Y}} \ \left[\mathbf{w}^{\mathsf{T}}\boldsymbol{\Phi}_{11} + (1 - y_{11}) \ \ldots \ \mathbf{w}^{\mathsf{T}}\boldsymbol{\Phi}_{n_t n_{t+1}} + (1 - y_{n_t n_{t+1}})\right]\hat{\mathbf{y}}, \end{aligned} \tag{13}$$

which has the same format with Eq. (11). Thus the most violated constraint is also identified by applying the Hungarian algorithm without enumerating all constraints as illustrated in Figure 2(b).

3.4 Exception Handling

Entering/leaving and missing/reappearing objects The structured prediction based on the Hungarian algorithm described in Section 3.3 matches all possible pairs of targets between two frames. However, due to various challenges including entering/leaving and missing/reappearing objects in real video sequences, the number of objects may be different in each frame and there are sometimes unmatched objects in practice.

This problem is dealt with by a two-stage optimization process. First, we introduce virtual agents—nodes representing proxy objects to allow for association of unmatched objects by the Hungarian algorithm. It can be implemented by constructing an extended affinity matrix for a bipartite graph as

$$\mathbf{K} = \left(\begin{array}{c|c} \mathbf{M}_{n_t \times n_{t+1}} & \mathbf{V}^{l}_{n_t \times n_t} \\ \hline \mathbf{V}^{e}_{n_{t+1} \times n_{t+1}} & \cdot \end{array}\right) = \left(\begin{array}{ccc|ccc} \mathbf{w}^{\mathsf{T}}\boldsymbol{\Phi}_{11} & \cdots & \mathbf{w}^{\mathsf{T}}\boldsymbol{\Phi}_{1n_{t+1}} & \mu & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{w}^{\mathsf{T}}\boldsymbol{\Phi}_{n_t 1} & \cdots & \mathbf{w}^{\mathsf{T}}\boldsymbol{\Phi}_{n_t n_{t+1}} & 0 & \cdots & \mu \\ \hline \mu & \cdots & 0 & \mu & \cdots & \mu \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & \mu & \mu & \cdots & \mu \end{array}\right), \tag{14}$$

where μ is the threshold for choosing virtual entering or leaving nodes for matching to minimize the matching error in data association. The matrix $\mathbf{M}$ trained by the S-SVM is for matching between detections in two frames. $\mathbf{V}^e$ and $\mathbf{V}^l$ correspond to entering (or reappearing) and leaving (or missing) objects, respectively; in other words, detections in D_{t+1} that are not matched with detections in D_t are represented in $\mathbf{V}^e$ whilst detections in D_t, not matched in D_{t+1}, are handled by $\mathbf{V}^l$. In this way, unmatched detections are associated with virtual nodes in the respective matrices.

In the second stage, the unmatched detections at frame t are simply added to a missing object list, which is maintained for potential matchings in the future. Also, the unmatched detections at frame $t+1$ are double-checked with objects in the missing list maintained until frame $t-1$; we run another Hungarian algorithm using the affinity matrix between the objects in the missing list and the unmatched detections at frame $t+1$. In other words, rather than keeping all unmatched objects in one large extended matrix, we split the problem into two smaller subproblems and solve them separately.

Occlusion handling Occlusion is another critical challenge in visual tracking, and even more difficult to handle with multiple targets since there may be substantial interactions between objects. If occlusion happens, detection may fail or the appearance of detected objects may be distorted significantly due to occlusion. Thus, we store a short history of target appearances—up to 10 frames in our implementation—and use the model in the history maximizing the appearance similarity to construct the augmented feature vector matrix $\mathbf{X}$.

Rejecting False Detections False detections occur randomly, and typically have short trajectories. For this reason, objects whose trajectories are shorter than a threshold, 5 frames in our experiment, are considered false detections, and thus are removed.

4 Experiments

Our data association technique based on the S-SVM is applied to tracking multiple humans in natural videos, and its performance is evaluated with public datasets. In this section, we first describe several important issues in our experiment setting, especially the representation of the augmented feature matrix, datasets, and evaluation metrics. Then, we analyze our experimental results and present both qualitative and quantitative performance of our algorithm.

4.1 Feature Representation

The augmented feature matrix $\mathbf{X}$ in Eq. (5) is composed of the feature vectors defined between a pair of detections. To calculate a feature vector $\mathbf{\Phi}_{ij}$ based on the affinities between two detections d_i^t and d_j^{t+1}, we employ five different kinds of features—color histogram in RGB space, Local Binary Patterns (LBP) [22], motion, bounding box overlapping ratio, and Euclidean distance between object centroids.

For color histogram and LBP, we divide the object region into 3×7 blocks[4] and compute their distributions in each block. We design a 3D RGB color histogram of $8 \times 8 \times 8$ dimension to represent the color information. We also construct

[4] This is because the object aspect ratio is set to 3:7 in the pedestrian detection algorithm we used.

a motion distribution for the entire object region. For motion features, we calculate optical flows between two consecutive frames and construct a Histogram of Optical Flow (HOF) from them [23]. The affinity between a pair of distributions is computed by the Bhattacharyya coefficient for these three features. The Bhattacharyya coefficient is computed for each feature and each block separately, and the coefficients become elements in the feature vector. The bounding box overlapping ratio is obtained by

$$\frac{\text{area}\left(B(d_i^t) \cap B(d_j^{t+1})\right)}{\text{area}\left(B(d_i^t) \cup B(d_j^{t+1})\right)} \tag{15}$$

where $B(d_i^t)$ denotes the bounding box for the detection d_i^t. The final feature vector $\mathbf{\Phi}_{ij}$ is 45 dimensional—21 dimensions for each, color and LBP, and 1 dimension for each, motion, bounding box overlapping ratio and centroid Euclidean distance. Note that we need to compute the feature vectors between all pairs of detections in two consecutive frames to create the augmented feature matrix $\mathbf{X}$.

4.2 Datasets

We tested our approach in three independent datasets—ETH [24], TUD [25], and PETS [26]. These are publicly available and have been commonly used for evaluating multi-target tracking algorithms [9, 12]. Pedestrian detection results used in [9, 12] are also employed in our experiments. For unbiased comparison, we chose the "bahnnof" and "sunnyday" sequences from ETH, used in [9, 12] and the "stadtmitte" sequence from TUD, used in [9, 12, 13] as well as the "S2.L1" sequence from PETS, used in [9, 13, 14]. For all experiments, we directly adopted the pedestrian detection results of [12, 14], which can be obtained from http://iris.usc.edu/people/yangbo/downloads.html.

Each dataset is constructed in a unique environment. For example, the ETH dataset has a perspective view with respect to pedestrians and is captured by cameras installed on a moving vehicle. The TUD and PETS datasets are recorded by static cameras in side and top view, respectively. Note that the feature vectors extracted from the sequences may have non-trivially different characteristics mainly due to the variations of the environment, especially viewpoints.

4.3 Results

Evaluation method We evaluated the performance of our algorithm by 3-fold cross validation using the three datasets. For the quantitative evaluation, we used performance evaluation metrics defined in [7] and compared our results with other state-of-the-art algorithms [9, 13, 12, 14]. Also, we added F-measure—the harmonic mean of precision and recall—to examine their combined performance. An overview of all investigated performance measures is presented in Table 1.

Table 1. Data association evaluation metrics used in experiments. All, except F-measure, are taken from [7]. Favor indicates whether a large value of a measure denotes a positive (+) or a negative (−) effect of performance.

Metric	Favor	Description
Recall	+	(# of correct matchings) / (# of detections in groundtruth)
Precision	+	(# of correct matchings)) / (# of detections in tracking)
F	+	F-measure - harmonic mean of precision and recall
FAF	−	Average false alarms per frame
GT		# of trajectories in ground truth
MT	+	Ratio of mostly ($\geq 80\%$) tracked trajectories
ML	−	Ratio of mostly ($\leq 20\%$) lost trajectories
PT		Ratio of partially tracked trajectories, i.e., $1 - \text{MT} - \text{ML}$
Frag	−	Fragments, # of times that a ground truth trajectory is interrupted
IDS	−	ID switch, # of times that a tracked trajectory changes its ID

Evaluation Quantitative results obtained for the TUD, ETH and PETS sequences are illustrated in Table 2, 3, and 4, respectively. Each table contains two different results by our algorithm since two different training sets were tested. S-SVM MOT denotes our algorithm, where corresponding training sets are shown within parentheses. As illustrated in the tables, our algorithm is comparable to other algorithms in most of the metrics except Frag and IDS; our method tends to have more fragments but fewer ID switches, so can be characterized as an extended tracklet generation technique. This is natural as our algorithm is an online method that finds the best matching only between two consecutive frames without global optimization. On the contrary, all other algorithms run offline. Moreover, PRIMPT and Online CRF are based on tracklets while our algorithm does not adopt any external low-level processing techniques. Perhaps, our algorithm can be improved significantly if it is also combined with tracklets. Examples of our qualitative results are shown in Figure 3.

Discussion As discussed in Section 4.2, the datasets have different characteristics. Even in such a case, it is possible for a classifier trained one or more sequences to be naturally and effectively applied to other different sequences without re-training. The quantitative results for all three datasets confirm this fact. Furthermore, our algorithm runs at approximately 15 frames per second using an unoptimized MATLAB implementation utilizing SVM$^{\text{struct}}$ package [27] on a desktop computer with Intel i7 3.4 GHz CPU and 16 GB memory (not including processing time for human detection), which is faster or comparable to all tested algorithms.

5 Conclusion

We presented an online data association technique for multi-object tracking by structured prediction. Our problem is formulated as the maximum weight matching in a bipartite graph, whose solution is learned through a structured support

Table 2. Results in TUD sequence based on training in PETS and ETH sequences.

Method	Recall	Precision	F	FAF	GT	MT	PT	ML	Frag	IDS
Energy Min. [13]	–	–	–	–	9	60.0%	30.0%	0.0%	4	7
PRIMPT [9]	81.0%	99.5%	0.89	0.028	10	60.0%	30.0%	10.0%	0	1
Online CRF [12]	87.0%	96.7%	0.92	0.184	10	70.0%	30.0%	0.0%	1	0
S-SVM MOT (PETS)	80.0%	96.7%	0.88	0.168	10	80.0%	20.0%	0.0%	11	0
S-SVM MOT (ETH)	83.0%	95.4%	0.89	0.246	10	80.0%	20.0%	0.0%	11	0

Table 3. Results in ETH sequence based on training in PETS and TUD sequences.

Method	Recall	Precision	F	FAF	GT	MT	PT	ML	Frag	IDS
PRIMPT [9]	76.8%	86.6%	0.81	0.891	125	58.4%	33.6%	8.0%	23	11
Online CRF [12]	79.0%	90.4%	0.84	0.637	125	68.0%	24.8%	7.2%	19	11
S-SVM MOT (PETS)	78.4%	84.1%	0.81	0.977	124	62.7%	29.6%	7.7%	72	5
S-SVM MOT (TUD)	78.2%	85.0%	0.81	0.907	124	63.3%	29.0%	7.7%	78	4

Table 4. Results in PETS sequence based on training in ETH and TUD sequences.

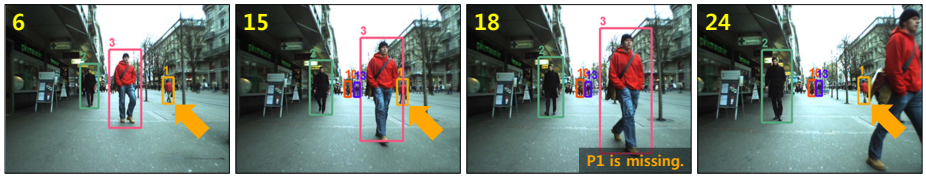
Method	Recall	Precision	F	FAF	GT	MT	PT	ML	Frag	IDS
Energy Min. [13]	–	–	–	–	23	82.6%	17.4%	0.0%	21	15
PRIMPT [9]	89.5%	99.6%	0.94	0.020	19	78.9%	21.1%	0.0%	23	1
NLMPRAM [14]	91.8%	99.0%	0.95	0.053	19	89.5%	10.5%	0.0%	9	0
S-SVM MOT (ETH)	97.2%	93.7%	0.95	0.379	19	94.7%	5.3%	0.0%	19	4
S-SVM MOT (TUD)	96.6%	93.4%	0.95	0.396	19	94.7%	5.3%	0.0%	26	5

vector machine. The proposed method associates multiple targets in two frames jointly; the similarity measure between a pair of targets is determined by the structured classifier, where dependencies between the multiple features are handled appropriately and automatically. We can deal with various challenges in multi-target tracking, such as misdetections, false alarms, occlusions, and others, effectively through the structured prediction. Our algorithm is fast and generalizes well to many other sequences without re-training.

Acknowledgement. This work was supported by MEST Basic Science Research Program through the NRF of Korea (2012-0003697), the IT R&D program of MKE/KEIT (10040246), and IT Consilience Creative Program of MKE and NIPA (C1515-1121-0003).

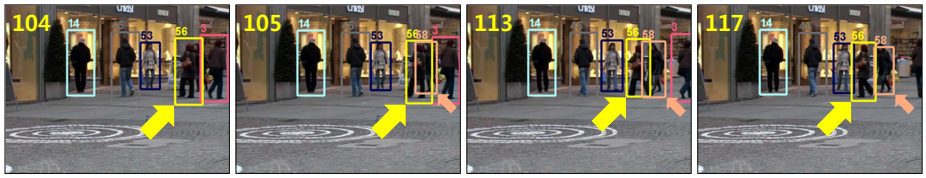
References

1. Leibe, B., Schindler, K., Van Gool, L.: Coupled detection and trajectory estimation for multi-object tracking. In: ICCV. (2007)
2. Jiang, H., Fels, S., Little, J.: A linear programming approach for multiple object tracking. In: CVPR. (2007)
3. Andriyenko, A., Schindler, K.: Globally optimal multi-target tracking on a hexagonal lattice. In: ECCV. (2010)

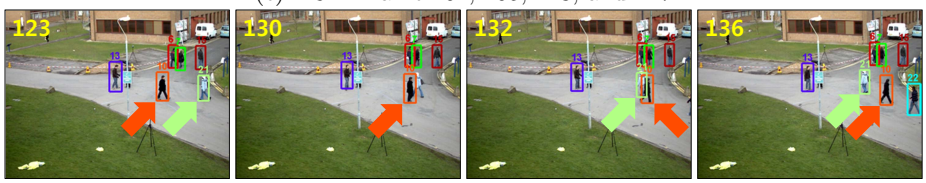


(a) ETH (bahnnof): frame 6, 15, 18, and 24

(b) ETH (sunnyday): frame 102, 108, 119, and 112



(c) TUD: frame 104, 105, 113, and 117



(d) PETS: frame 123, 130, 132, and 136

Fig. 3. Example of our results. (a) Person1 is temporarily occluded and missing for a while but its ID is recovered after occlusion. (b) Person22 is tracked with severe occlusion. The trajectories for person3 and person32 are fragmented or missing due to illumination changes. (c) Person56 and person58 are tracked correctly even with severe occlusion and interaction. (d) Person10 and person21 preserve their ID's even after crossing.

4. Zhang, L., Li, Y., Nevatia, R.: Global data association for multi-object tracking using network flows. In: CVPR. (2008)
5. Pirsiavash, H., Ramanan, D., Fowlkes, C.: Globally-optimal greedy algorithms for tracking a variable number of objects. In: CVPR. (2011)
6. Huang, C., Wu, B., Nevatia, R.: Robust object tracking by hierarchical association of detection responses. In: ECCV. (2008)
7. Li, Y., Huang, C., Nevatia, R.: Learning to associate: Hybridboosted multi-target tracker for crowded scene. In: CVPR. (2009)
8. Kuo, C.H., Huang, C., Nevatia, R.: Multi-target tracking by on-line learned discriminative appearance models. In: CVPR. (2010)

9. Kuo, C.H., Nevatia, R.: How does person identity recognition help multi-person tracking? In: CVPR. (2011)
10. Yang, B., Huang, C., Nevatia, R.: Learning affinities and dependencies for multi-target tracking using a CRF model. In: CVPR. (2011)
11. Brendel, W., Amer, M., Todorovic, S.: Multiobject tracking as maximum weight independent set. In: CVPR. (2011)
12. Yang, B., Nevatia, R.: An online learned CRF model for multi-target tracking. In: CVPR. (2012)
13. Andriyenko, A., Schindler, K.: Multi-target tracking by continuous energy minimization. In: CVPR. (2011)
14. Yang, B., Nevatia, R.: Multi-target tracking by online learning of non-linear motion patterns and robust appearance models. In: CVPR. (2012)
15. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: CVPR. (2005)
16. Leibe, B., Leonardis, A., Schiele, B.: Robust object detection with interleaved categorization and segmentation. IJCV **77** (2008) 259–289
17. Felzenszwalb, P., Girshick, R., McAllester, D., Ramanan, D.: Object detection with discriminatively trained part-based models. IEEE PAMI **32** (2010) 1627–1645
18. Wu, B., Nevatia, R.: Detection and tracking of multiple, partially occluded humans by bayesian combination of edgelet based part detectors. IJCV **75** (2007) 247–266
19. Yang, M., Lv, F., Xu, W., Gong, Y.: Detection driven adaptive multi-cue integration for multiple human tracking. In: ICCV. (2009)
20. Tsochantaridis, I., Hofmann, T., Joachims, T., Altun, Y.: Support vector machine learning for interdependent and structured output spaces. In: ICML. (2004)
21. Kuhn, H.W.: The Hungarian method for the assignment problem. Naval Research Logistics Quarterly **2** (1955) 83–97
22. Wang, X., Han, T.X., Yan, S.: An HOG-LBP human detector with partial occlusion handling. In: ICCV. (2009)
23. Laptev, I., Marszalek, M., Schmid, C., Rozenfeld, B.: Learning realistic human actions from movies. In: CVPR. (2008)
24. Ess, A., Leibe, B., Schindler, K., , van Gool, L.: A mobile vision system for robust multi-person tracking. In: CVPR. (2008)
25. Andriluka, M., Roth, S., Schiele, B.: Monocular 3d pose estimation and tracking by detection. In: CVPR. (2010)
26. PETS: IEEE International Workshop on Performance Evaluation of Tracking and Surveillance, (http://www.cvg.rdg.ac.uk/PETS2009/) (2009)
27. Vedaldi, A.: A MATLAB wrapper of SVM$^{\text{struct}}$. http://www.vlfeat.org/ vedaldi/code/svm-struct-matlab.html (2011)