Max-Margin Offline Pedestrian Tracking with Multiple Cues

Bahman Yari Saeed Khanloo[1], Ferdinand Stefanus[1], Mani Ranjbar[1],
Ze-Nian Li[1], Nicolas Saunier[2], Tarek Sayed[3], and Greg Mori[1]

[1] School of Computing Science
Simon Fraser University
{byari, fsa21, mra33, li, mori}@cs.sfu.ca

[2]Dept. of Civil, Geological and Mining Engineering
École Polytechnique de Montréal
nicolas.saunier@polymtl.ca

[3]Dept. of Civil Engineering
University of British Columbia
tsayed@civil.ubc.ca

Abstract

In this paper, we introduce MMTrack, a hybrid single pedestrian tracking algorithm that puts together the advantages of descriptive and discriminative approaches for tracking. Specifically, we combine the idea of cluster-based appearance modeling and online tracking and employ a max-margin criterion for jointly learning the relative importance of different cues to the system. We believe that the proposed framework for tracking can be of general interest since one can add or remove components or even use other trackers as features in it which can lead to more robustness against occlusion, drift and appearance change. Finally, we demonstrate the effectiveness of our method quantitatively on a real-world data set.

1 Introduction

Object tracking is an important computer vision task with numerous practical applications such as pedestrian tracking, user interfaces and traffic monitoring. Most object tracking systems amount to defining a set of features that best describe the appearance of an object, and combining the features with a motion model to track the object from frame to frame. The appearance and motion models are usually treated separately by object tracking algorithms, using independent components whose parameters are set or learned independently (e.g. Haar-like features for appearance model and constant velocity dynamics as the motion model [17], or Eigenbasis appearance model-Brownian motion model pair [10]).

Further, choosing an appearance model is itself a challenging problem due to appearance change of the target object and potential similarity in appearance with other objects. Ramanan et al. [9] consider a static approach to appearance modeling using clustering while Collins et al. [2] use online adaptive appearance modeling. A good appearance model should be able to strike a balance between resistance to drift and adaptation of the object's appearance over time. This suggests that a combination of static and continually updated cues may help a tracker achieve superior performance. It has been shown that combining different cues helps improve tracking performance, if it is done in a principled manner [13]. Many existing multi-cue tracking algorithms, however, combine the cues with either fixed weighting [4, 12, 11], or update the weights according to a heuristically-selected measure [6, 7].

Another approach that has gained popularity recently is to use some feature selection criteria to select the features that best discriminate the appearance of the object from its surroundings. In [2], a Fisher-like criterion is used to select the most discriminative features at each frame, whereas in [1], a boosting mechanism is used to select the best features from a fixed pool of features. Both methods, however, use only one type of feature (linear combination of RGB channels for [2], and Haar-like features for [1]). In our work, we try to combine different feature types, where each feature type may have different contribution relative to other features. Determining the weighting of different features in this setting is not a straightforward task, due to the interdependency between different features. It should also be noted that both [2] and [1] are only concerned with appearance modeling and do not learn a motion model.

The main contribution of this paper is to employ Structural SVM in the context of object tracking. Structural SVM is a recent development in machine learning that generalizes SVM formulation to deal with interdependent and structured variables[14, 15]. Object tracking can be formulated as a structured output prediction problem, as it tries to find the sequence of coordinates that best explain input features. This is verified by the observation that each pair of coordinates are strongly interdependent since the valid range of possible movements of the object is restricted by the motion model. In other words, an object is more likely to move to nearby locations in the next frame than to locations

that are far away. In addition, by formulating the problem as a chain-structured Markov Random Field (MRF) with emission and transition models, we will show that Structural SVM also provides an intuitive and principled way to treat the feature representation and motion model in a unified way while jointly learning the relative contributions of multiple cues.

The chain-structured MRF model and the inference scheme that we adopt is similar to the model adopted in [4]. However, our objective is not to build a multi-person tracker but rather to define a principled way to combine different cues. Francois et al. [4] combine simple cues such as ground plane occupancy and color model by treating them equally (i.e. using fixed weighting), whereas we try to find weights that represent relative contributions of different cues. In a way, the idea of combining different cues we use in our framework is closely related to the approach presented in [13]. The main difference is that instead of combining the results of multiple 'observers' (i.e. complete tracker systems each of which having its own appearance and motion model), we fuse different appearance models and a motion model. Moreover, the parameters for fusing the appearance models and motion model are learned jointly in our case whereas in [13] the observers are combined according to error distributions that are learned independently for each observer.

2 Pedestrian Tracking

In this section, we introduce our tracking system called MMTrack in the context of pedestrian tracking. The system comprises three main components: descriptive features, discriminative features and motion model. Descriptive features are used to represent the appearance of the target pedestrian, and discriminative features are used to distinguish between the tracked object from other objects. The motion model favors specific movement patterns from one frame to another. A large margin learning approach combines these three cues by learning the relative importance of the components. Finally, the learned model is used to estimate pedestrian trajectories.

The rest of this section is organized as follows. Section 2.1 describes our trajectory scoring model. Section 2.2 outlines the features that provide us with cues for locating a desired pedestrian. We explain our max-margin framework for estimating the model parameters in Section 2.3 and outline our efficient inference scheme in Section 2.4.

2.1 Trajectory Scoring Model

We are interested in offline tracking where we exploit the fact that the positions of the target in consecutive frames are interdependent and the starting location is given (e.g. by a detector). This is in contrast to online tracking that greedily picks the next best location of the object at each frame. The tracking problem in our setting can be formulated as one of finding the trajectory with the highest score given a starting location, assuming that a scoring function that can measure the compatibility between a trajectory and a video sequence exists. Further, since we are using many features, the scoring function should take into account the relative contribution of each feature in describing the trajectories.

Hence, the scoring function is a mapping in the form of $F(\mathbf{x}, \mathbf{y}; \mathbf{w}) : \mathcal{X}^T \times \mathcal{Y}^T \rightarrow \mathcal{R}$ from a sequence of frames $\mathbf{x} = \{\mathbf{x^{(1)}}, ..., \mathbf{x^{(T)}}\}$ and a trajectory $\mathbf{y} = \{\mathbf{y^{(1)}}, ..., \mathbf{y^{(T)}}\}$ to a real number where each location $\mathbf{y^{(t)}}$ is assigned to one of the image pixels and $\mathbf{w}$ is a set of weights that represent the importance of each of the features extracted from the frames. Further, we assume that if the location of the object at a particular frame is known, the locations of the object in previous and following frames will be independent from each other. Therefore, we model a trajectory as a chain-structured MRF. The scoring function of this model is decomposed into two contributions: *transition model* and *emission model*. The transition model, which is described by the motion model, defines the compatibility of the locations of the target between two consecutive frames. The emission model is a measure of compatibility of a location and the observed features at that location. Thus, we define the score of a trajectory as

$$F(\mathbf{x}, \mathbf{y}; \mathbf{w}) = \sum_{t=2}^{T} F_{\mathcal{T}}^{(t)}(\mathbf{y}^{(t-1)}, \mathbf{y}^{(t)}; \mathbf{w}_{\mathcal{T}}) + \sum_{t=2}^{T} F_{\mathcal{E}}^{(t)}(\mathbf{x}^{(1)}, \mathbf{x}^{(t-1)}, \mathbf{x}^{(t)}, \mathbf{y}^{(1)}, \mathbf{y}^{(t-1)}, \mathbf{y}^{(t)}; \mathbf{w}_{\mathcal{E}}) \quad (1)$$

where $F_{\mathcal{T}}^{(t)}(\cdot)$ and $F_{\mathcal{E}}^{(t)}(\cdot)$ are linear models describing transition and emission contributions at time t respectively. These functions are parameterized by $\mathbf{w}_{\mathcal{T}}$ and $\mathbf{w}_{\mathcal{E}}$ whose concatenation we denote by $\mathbf{w}$.

2.1.1 Emission Model

The emission model includes several features whose weighted combination votes for the presence of the target pedestrian. These features include Histogram of Oriented Gradient [3] (HOG) feature as discriminative feature and color histogram distance and two appearance templates as descriptive features. Thus, the emission model decomposes into the following contributions

$$\begin{aligned} F_{\mathcal{E}}^{(t)}(\cdot; \mathbf{w}_{\mathcal{E}}) &= {\mathbf{w}_{\mathcal{C}}}^T \Psi_{\mathcal{C}}(\mathbf{x}^{(t)}, \mathbf{y}^{(t)}) \\ &+ {\mathbf{w}_{\mathcal{H}}}^T \Psi_{\mathcal{H}}(\mathbf{x}^{(t)}, \mathbf{y}^{(t)}) \\ &+ {\mathbf{w}_{\mathcal{F}}}^T \Psi_{\mathcal{F}}(\mathbf{x}^{(1)}, \mathbf{x}^{(t)}, \mathbf{y}^{(1)}, \mathbf{y}^{(t)}) \\ &+ {\mathbf{w}_{\mathcal{P}}}^T \Psi_{\mathcal{P}}(\mathbf{x}^{(t-1)}, \mathbf{x}^{(t)}, \mathbf{y}^{(t-1)}, \mathbf{y}^{(t)}) \end{aligned} \quad (2)$$

where $\Psi_{\mathcal{C}}(\cdot)$, $\Psi_{\mathcal{H}}(\cdot)$, $\Psi_{\mathcal{F}}(\cdot)$ and $\Psi_{\mathcal{P}}(\cdot)$ denote the feature functions representing color histogram distance, HOG features and the difference between appearance templates of the first frame and the previous frame to the current frame corresponding to a bounding box around the hypothesized location $\mathbf{y}^{(t)}$ respectively. We concatenate all the emission weights to give $\mathbf{w}_{\mathcal{E}} = [\mathbf{w}_{\mathcal{C}}; \mathbf{w}_{\mathcal{H}}; \mathbf{w}_{\mathcal{F}}; \mathbf{w}_{\mathcal{P}}]^T$. Intuitively, $\mathbf{w}_{\mathcal{E}}$ weighs the emission features to give a map that ideally peaks at the body center of the target pedestrian.

2.1.2 Transition Model

Similar to the emission model, we define the transition model as

$$F_{\mathcal{T}}^{(t)}(\mathbf{y}^{(t-1)}, \mathbf{y}^{(t)}; \mathbf{w}_{\mathcal{T}}) = {\mathbf{w}_{\mathcal{T}}}^T \Psi_{\mathcal{T}}(\mathbf{y}^{(t-1)}, \mathbf{y}^{(t)}), \quad (3)$$

where $\Psi_{\mathcal{T}}(\cdot)$ is a symmetric first-order motion model. The motion model discretizes the distance travelled between two consecutive frames into several bins that represent concentric circles centered at the previous location. So, we have

$$\Psi_{\mathcal{T}}(\mathbf{y}^{(t-1)}, \mathbf{y}^{(t)}) = \text{bin}(d(\mathbf{y}^{(t-1)}, \mathbf{y}^{(t)})), \quad (4)$$

$$\text{bin}_k(d') = \mathbb{1}_{[\lfloor d' \rfloor = k]}, \quad k = 0, ..., \lfloor d_{\max} \rfloor. \quad (5)$$

Here, $d(\mathbf{y}, \mathbf{y}')$ is the Euclidean distance between the $2d$ image locations of $\mathbf{y}$ and $\mathbf{y}'$, $\mathbb{1}_{[.]}$ is the indicator function and $\text{bin}(\cdot)$ acts as a selection operator that generates a vector of length $d_{max}+1$ with all the elements set to 0 except one being 1. The upper bound d_{max} on the travelled distance from one frame to the next one is estimated using the dataset. Note that the symmetric motion model results in $\mathbf{w}_{\mathcal{T}}$ being a disk-like motion prior which is learned jointly with the emission model parameters.

2.2 Features

We use a combination of discriminative and descriptive features in our tracking framework. While descriptive features are used to describe the appearance of the object being tracked, discriminative features are intended to distinguish the tracked object from other objects. We use color histogram distance and appearance templates as the descriptive features and HOG score map as our discriminative feature.

The intuition behind using different features is that each of the features will provide the tracking system with different information. We expect the HOG feature to be of help to the system in discriminating between pedestrians and non-pedestrian objects (e.g. car, trees, etc.). Color histogram distance features are used to provide information about static appearance of the tracked pedestrian. Appearance template features, on the other hand, provide information about a particular pedestrian at a finer level than the histogram distance features. Further, other than the static histogram distance features, another type of appearance template feature is also used to provide a constantly updated appearance model of a pedestrian. With the combination of both static and continually-updated appearance model, we hope to achieve a good balance between resistance to track drift and adaptation to a pedestrian's varying appearance throughout its trajectory.

2.2.1 HOG Score

We use the output of the linear SVM classifier that operates on Histogram of Oriented Gradients [3] as a feature to help our system differentiate between pedestrians and other objects. We trained the SVM to detect pedestrians in top-down view so as to make it suitable for our particular experimental setup. The detection window size is set to 48x112 and the detection is performed on a pyramid built on the input image with its scale varying up to 130% of the original resolution. For each pixel location, we take the maximum SVM score over all image resolutions resulting in a score map where the peaks vote for presence of pedestrians. We then normalize these scores so they fall within the range [0,1] and use the final map as our feature. Figure 1 illustrates an input image along with the corresponding normalized HOG score map.

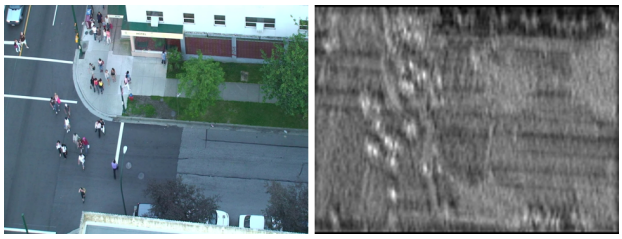

Figure 1. A frame and its corresponding HOG feature map.

2.2.2 Color Histogram Distance

Although HOG scores can help the system differentiate between pedestrians and other objects, they are not informative in distinguishing among different pedestrians, as many pedestrians will have high HOG scores. Thus, features that can uniquely respresent the appearance of a pedestrian are needed. We incorporate such features using a static color histogram model obtained from clustering as is done in [9]. The main idea here is that by clustering the histograms obtained from bounding boxes around the pedestrians throughout the video, we can gain a good insight into how the average appearance statistics of each of the people looks like. Note that the appearance model obtained in this method is a static one and this property makes it more resis-

tant againts drift that often occurs in tracking systems with dynamically-updated appearance models.

The generation of the color histogram distance features is as follows. Based on the image evidence inside the bounding box obtained from a HOG detection, nine histograms are computed over different sections of a pedestrian's body as depicted in the second column of Figure 2. Each histogram consists of 30 bins with 10 bins for each of the R, G and B channels. These nine histograms are then concatenated together to give one histogram characterizing the person's appearance. Next, we cluster all the instances of histograms of all the people in the video using the mean-shift clustering algorithm. We then represent the target pedestrian using the mean of the cluster to which it belongs. Finally, we compute one histogram distance map for each of the nine body sections by computing, at each pixel location, the χ^2-distance between the histogram built using the image observation within the corresponding section of the bounding box centered at that pixel and the mean of the cluster to which the target belongs. The resulting maps have low values in areas with similar color to the target person's and high values elsewhere. We efficiently compute the histogram distance maps using the integral histogram trick [8].

2.2.3 Appearance Templates

Besides a person's color histogram, we also use appearance templates to describe the person's appearance. We use two templates: an initial template obtained from the initial frame, and another template obtained from the previous frame. Like the histogram appearance feature, the initial template is fixed and is used to provide a fixed reference to the person's appearance. However, the initial appearance template describes the person's appearance at a finer level of detail than the histogram distance features. This template acts as a memory template which ensures that the tracker does not completely forget about the appearance of the target when it first showed up. On the other hand, the previous frame template incorporates the idea of online tracking because it is continually updated during the inference, which helps the system cope with a limited degree of object appearance changes over time.

Distance maps are computed for each of the templates by sliding each template across the current frame, and computing the sum of absolute pixel difference at each location, which can be done efficiently with a modified integral image trick [16]. These distance maps are then normalized between 0 and 1.

2.3 Large Margin Parameter Learning

As mentioned earlier, we use a scoring function parameterized by a set of weights $\mathbf{w}$ which puts together a variety of features. The learning task amounts to jointly learning the parameters that best explain the dependencies between the features and the trajectories using video-trajectory training pairs. We use a discriminative approach, namely we try to discriminate between a compatible video-trajectory pair and all the runner-ups. Hence, we find a predictor that estimates the best trajectory given an input video by learning a set of parameters that maximize the score of training set examples.

Learning the model parameters in this problem setting is challenging since we do not have negative examples. In other words, we do not know how a "bad" trajectory looks like and more importantly, how it differs from a "good" one because this information is not included in the dataset. Notice that the scoring function in equation 1 can be viewed as a $\mathbf{w}$-parameterized discriminant function. Further, the locations in a trajectory are highly interdependent and so we are dealing with a structured output problem. Hence, it is natural to adopt Structural SVM to jointly estimate the parameters.

According to the large margin criterion used by Structural SVM, we require a set of parameters that maximize the score of N given ground truth tracks while pushing away the score of all other possible trajectories from these maxima and hence the following program

$$\min_{\mathbf{w},\xi} \frac{1}{2}||\mathbf{w}^2|| + \frac{C}{N}\sum_{i=1}^{N}\xi_i, \quad s.t. \quad \forall i, \xi_i \geq 0, \tag{6}$$

$$F(\cdot, \mathbf{y}_{(i)}) - F(\cdot, \mathbf{y}) \geq \Delta(\mathbf{y}_{(i)}, \mathbf{y}) - \xi_i, \quad \forall i = 1, ..., N, \forall \mathbf{y} \in \mathcal{Y}^T \backslash \mathbf{y}_{(i)}. \tag{7}$$

The constant $C > 0$ specifies the relative importance of margin maximization and error minimization which is determined by cross validation. Note that 6 guarantees a unique solution whereas the constraints in 7 require the score of ground truth $\mathbf{y}_{(i)}$ to be at least as far away from the score of a possibly incorrect trajectory $\mathbf{y}$ as the loss $\Delta(\mathbf{y}_{(i)}, \mathbf{y})$ incurred when predicting $\mathbf{y}$. The loss function measures the total squared Euclidean distance between corresponding locations in two trajectories:

$$\Delta(\mathbf{y}_{(i)}, \mathbf{y}) = \sum_{t \in \mathcal{T}} d^2(\mathbf{y}_{(i)}{}^{(t)}, \mathbf{y}^{(t)}). \tag{8}$$

We use the SVM^{struct} framework[15] to solve this problem. In this approach, we find a subset of inequality constraints in equation 7, the most violated ones, and then solve for them such that all the constraints are violated by no more than a desired precison.

2.4 Approximate Inference

After the model parameters are learned, the inference task becomes one of finding the highest scoring trajectory

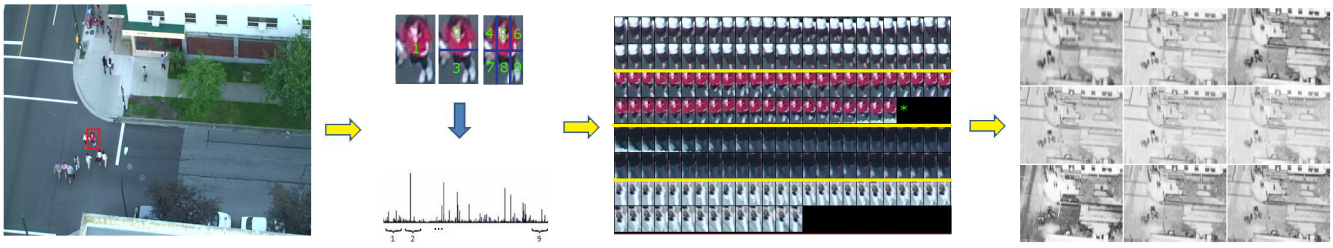

Figure 2. Color histogram distance features generation: Nine histograms are computed over nine sections of a detected person's bounding box and the resulting histograms are concatenated to give one full-histogram and all such histograms are clustered. Histogram distance maps are then generated for every frame by computing the χ^2-distance between the histogram of each of the sections to the cluster mean to which the target belongs.

given the model parameters. Exhaustive search for the highest scoring trajectory in the space of possible trajectories is computationally intractable. We efficiently solve for this problem using the Viterbi algorithm which is given by the following dynamic program

$$M^{(t)}(l_C) = \max_{l_N}\Big(M^{(t-1)}(l_N) + F_{\mathcal{T}}^{(t)}(\hat{\mathbf{y}}^{(t-1)} = l_C, \hat{\mathbf{y}}^{(t)} = l_N)\Big) + F_{\mathcal{E}}^{(t)}(., \hat{\mathbf{y}}^{(t)} = l_N),$$
$$t = 1, ..., T, \quad l_N \in \mathrm{N}(l_C). \tag{9}$$

Each element of the message vector $M^{(t)}$ corresponds to a pixel and indicates the marginal score of the highest scoring track that originates at the initial location and terminates at that pixel at time t. A traceback from the final most scoring location is done to recover the track. In our notation, l_C and l_N refer to the current and next location respectively and we just look in the neighborhood $N(l_C)$ when searching for the next possible location instead of full search. Note that this local search is valid since it complies with the nature of the movements of a pedestrian because the pedestrian is not expected to jump to a pixel which is far away from the current location. Namely, we are finding an exact solution in the space of "valid" trajectories.

However, performing the inference in high resolutions turns out to be computationally prohibitive even with local search and integral histogram optimizations. Thus, we resort to pruning strategies for inference by performing beam search i.e. just evaluate the N highest-scoring trajectories at each time step (N is set to 3 in our experiments) and discard the rest. This allows us to produce the tracking results in the original resolution while keeping the inference feasible.

Obviously, beam search will return suboptimal results because it does not explore the whole hypothesis space. However, experimental results show that our approximate inference scheme works well in practice.

3 Experimental Evaluation

We use UBC Fireworks dataset for our experiments [5]. The dataset consists of clips recorded at 1440×1080 resolution using a stationary camera installed on top of a building in downtown Vancouver (an example frame is shown in the left side of Figure 1). Hence, a top-down view of a moderately crowded scene is captured with variety of moving objects typical to an urban setting present in the image. This includes cars, bikers and pedestrians. The amount of change in illumination, scale and pose is not significant but one needs to deal with background clutter and partial occlusions. The main challenges in the dataset are the presence of occasional crowded blobs of moving pedestrians that introduces many potential distractors and significant background change that occurs when people move from sidewalk to street area and vice versa.

We use 10 manually-labeled trajectory sequences for training and 22 other manually-labeled sequences for testing with the labels being used as ground truth. Both training and test sequences contain easy, moderate and hard sequences ranging from a solitary person going through the scene to a pedestrian walking within a crowd.

3.1 Implementation Details

To reduce training time, we precompute HOG and color histogram distance features prior to training and testing. Appearance templates, however, must be generated online and we compute them efficiently using integral images. We significantly reduce the space of possible trajectories in training by running the Viterbi algorithm in steps of nine pixels in both horizontal and vertical directions. This means that the actual working resolution for Viterbi is 160×120. We define the neighborhood $N(l_C)$ to be the area within a radius of 2 pixels centered at the current location l_C.

This choice is made based upon empirical statistics of the dataset. Note also that because testing processes images at different resolution from training, the motion model obtained from training must be adapted for testing. So, a simple nearest neighbor interpolation of the motion model is performed using the same number of discretization bins as in training and the weights are used directly.

3.2 Results

We compare the results of our tracking system with the algorithms proposed in [2] and [1]. To gain insight into the importance of having a combination of all the features, we also provide the results of our algorithms when some of the features are turned off. Note that we have learned different sets of parameters for each combination of the features.

We use the same procedure to extract trajectories from all the trackers. We first initialize a tracker from a HOG detection, where the HOG detection is chosen such that it is contained in one of the 22 manually-labeled sequences that we use for testing. We then run the tracker forward and backward in time in order get a complete trajectory regardless of the initial position provided by the HOG detection. The tracker is terminated once it is within a certain number of pixels from the image borders (this scheme works in this dataset because pedestrians' entry and exit locations are located around the image borders).

Besides the usual average pixel error measure, we use two other performance measures proposed in [18]. *Correct Detected Track (CDT)* indicates the number of correct trajectories. A track is defined as a CDT if the amount of spatial and temporal overlap with the ground truth exceed thresholds T_{ov} and TR_{ov} respectively, where T_{ov} and TR_{ov} are both set to 0.5 in our experiments. This roughly means that at least half of a CDT must temporally coincide with its ground truth, its length cannot be less than half of its ground truth, and the average spatial overlap must be at least 0.5. *Closeness of Track (CT)* is defined as the average spatial overlap between a ground truth and a system track in the temporally coincident portion of the track. Its value ranges from 0 to 1, with 1 indicating that the track is exactly the same as the ground truth in the temporally coincident section of the track. More detailed explanation of the measures are provided in [18].

As can be seen from Table 1, our proposed tracker achieves a better performance than MILTracker [1] and Collins-Liu tracker [2] in this dataset. One can explain this promising performance by reasoning about our system having different cues that decribe the desired pedestrian as well as distinguishing it from background and our principled way of cue combination. More specifically, HOG feature helps the tracker eliminate areas belonging to non-pedestrian objects, static histogram distance feature provides rough description of the pedestrian and helps alleviate drift whereas appearance templates provide finer levels of a pedestrian model, with the previous frame appearance template allowing some degree of adaptability to appearance change over time.

Removing some of the features significantly reduces the performance of our tracker, indicating that the combination of HOG, histogram distance, and template appearance features is essential in achieving good performance.

An example illustrating the importance of our cue combination strategy is shown in Fig 3. As can be seen in the red inset, our tracker with only HOG and histogram distance feature drifts to a nearby pedestrian at some parts of the track, because there is a pedestrian with similar color to the tracked person nearby. On the other hand, HOG and appearance template drifts to a background area at the boundary between the pavement and the street, probably because the sum of absolute difference measure used in the appearance template is sensitive to the significant change in background pixels that occurs at this boundary. The combination of HOG, color histogram distance and appearance templates manages to track the person correctly.

4 Conclusion and Future Work

In this paper, we introduced MMTrack, a tracking system that employs a large margin learning criterion to combine different sources of information effectively. Although MMTrack is used for pedestrian tracking in this work, we believe that our framework is general and can be used to track other object categories as long as the features describe the object of interest well.

Our tracking system has its limitation in handling severe occlusion and track hijacks caused by significant change in the person's appearance or situations where the background patch is very similar to the appearance of the target. A possible future direction is to extend this framework to more complicated systems that can parameterize multiple person tracking. Moreover, learning different parameters for different locations in the image may also be of interest. This is motivated by the intuition that the relative importance of the features is likely to be affected by the statistics of back-

Tracker	# CDT	Avg CT	Avg Error
MMTrack: All	21	0.66	7.01
MMTrack: HOG+Hist	10	0.47	14.40
MMTrack: HOG+Template	14	0.52	22.24
MILTrack [1]	19	0.61	19.87
Collins-Liu [2]	14	0.54	21.24

Table 1. Tracking results on 22 test sequences.

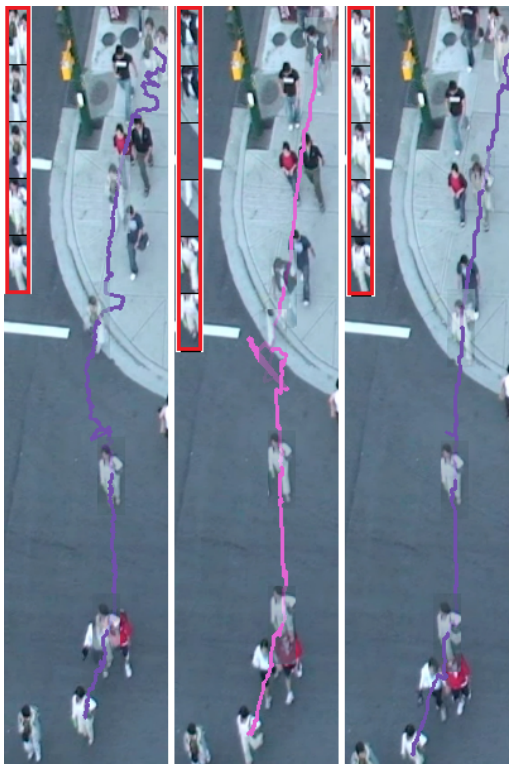

Figure 3. Trajectories obtained with HOG + histogram (left), HOG + appearance template (middle), and HOG + histogram + template features (right). The tracked objects throughout the trajectories are shown in the red insets, and also superimposed along the trajectories.

ground patches at different locations.

References

[1] B. Babenko, M.-H. Yang, and S. Belongie. Visual Tracking with Online Multiple Instance Learning. In *IEEE Conference on Computer Vision and Pattern Recognition 2009 (CVPR'09)*, pages 983–990, 2009.

[2] R. Collins, Y. Liu, and M. Leordeanu. On-line selection of discriminative tracking features. *IEEE Transactions of Pattern Analysis and Machine Intelligence*, 27(10):1631–1643, 2005.

[3] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In C. Schmid, S. Soatto, and C. Tomasi, editors, *International Conference on Computer Vision & Pattern Recognition*, volume 2, pages 886–893, INRIA Rhône-Alpes, ZIRST-655, av. de l'Europe, Montbonnot-38334, June 2005.

[4] J. B. Francois, J. Berclaz, F. Fleuret, and P. Fua. Robust people tracking with global trajectory optimization. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 744–750, 2006.

[5] K. Ismail, T. Sayed, and N. Saunier. Automated collection of pedestrian data using computer vision techniques. In *Transportation Research Board Annual Meeting Compendium of Papers*, Washington, D.C., Jan. 2009.

[6] M. Kolsch and M. Turk. Fast 2d hand tracking with flocks of features and multi-cue integration. In *CVPRW '04: Proceedings of the 2004 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'04) Volume 10*, page 158, Washington, DC, USA, 2004. IEEE Computer Society.

[7] H. Liu, L. Zhang, Z. Yu, H. Zha, and Y. Shi. Collaborative mean shift tracking based on multi-cue integration and auxiliary objects. In *IEEE International Conference of Image Processing (ICIP)*, volume 3, pages 217–220, 2007.

[8] F. Porikli. Integral histogram: A fast way to extract higtograms in cartesian spaces. *International Conference on Computer Vision & Pattern Recognition*, 1:829–836, June 2005.

[9] D. Ramanan, D. Forsyth, and K. Barnard. Building models of animals from video. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, volume 28, pages 1319–1334, 2001.

[10] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang. Incremental learning for robust visual tracking. *International Journal of Compute Vision*, 77:125–141, May 2008.

[11] L. Song, R. Zhang, Z. Liu, and X. Chen. Object tracking based on parzen particle filter using multiple cues. *Advances in Multimedia Information Processing*, 4810/2007:206–215, 2005.

[12] M. Spengler and B. Schiele. Towards robust multi-cue integration for visual tracking. In *Machine Vision and Applications*, volume 14, pages 50–58, 2003.

[13] B. Stenger, T. E. Woodley, and R. Cipolla. Learning to track with multiple observers. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[14] B. Taskar, C. Guestrin, and D. Koller. Max-margin markov networks. In *Neural Information Processing (NIPS)*, 2003.

[15] I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun. Support vector machine learning for interdependent and structured output spaces. In *IEEE International Conference on Machine Learning (ICML)*, pages 823–830, 2004.

[16] P. Viola and M. Jones. Robust real-time object detection. In *International Journal of Computer Vision (IJCV)*, 2001.

[17] J. Wang, X. Chen, and W. Gao. Online selecting discriminative tracking features using particle filter. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1037–1042, 2005.

[18] F. Yin, D. Makris, and S. A. Velastin. Performance evaluation of object tracking algorithms. *IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS2007)*, October 2007.