Object Association Across PTZ Cameras using Logistic MIL*

Karthik Sankaranarayanan James W. Davis
Dept. of Computer Science and Engineering
Ohio State University
Columbus, OH 43210 USA
{sankaran,jwdavis}@cse.ohio-state.edu

Abstract

We propose a novel approach to associate objects across multiple PTZ cameras that can be used to perform camera handoff in wide-area surveillance scenarios. While previous approaches relied on geometric, appearance, or correlation-based information for establishing correspondences between static cameras, they each have well-known limitations and are not extendable to wide-area settings with PTZ cameras. In our approach, the slave camera only passively follows the target (by loose registration with the master) and bootstraps itself from its own incoming imagery, thus effectively circumventing the problems faced by previous approaches and avoiding the need to perform any model transfer. Towards this goal, we also propose a novel Multiple Instance Learning (MIL) formulation for the problem based on the logistic softmax function of covariance-based region features within a MAP estimation framework. We demonstrate our approach with multiple PTZ camera sequences in typical outdoor surveillance settings and show a comparison with state-of-the-art approaches.

1. Introduction

Multiple PTZ cameras are commonly used in wide-area surveillance applications. While these cameras can detect and track objects individually and independently, in order to further exploit the "networked" nature of these systems it is necessary to establish correspondences between them to enable them to function in an integrated setting. To achieve this, one needs to address problems such as camera calibration, registration (to establish overlapping zones), and object association across multiple cameras. In this paper we address the problem of associating objects across cameras in a novel manner.

There are two main approaches towards establishing object correspondences across multiple cameras: Geometry-based and Appearance-based. Geometry-based approaches [4, 6, 12] work by using 3D camera geometry and calibration information and also exploit homography constraints to establish correspondences between pixels in the different views without looking at the appearance information. Alternative approaches such as [9, 14] learn correspondences between views without explicitly performing calibration, but by instead modeling the correlations between activity levels at pixel locations across multiple views. Appearance-based approaches [8, 18] try to learn the inter-camera color calibration to effectively transfer the appearance models between different views so as to reliably handoff the object model. These are mutually exclusive approaches and a complete system would need to employ an effective combination of these techniques to achieve robust association across cameras.

While appearance-based approaches have seen some success in constrained settings (with static cameras), in case of PTZ camera systems, robustly transferring appearance information and learning brightness transfer functions across them for the whole field-of-coverage is a very challenging task. This is primarily due to two reasons (i) non-uniform appearance nature of targets - different poses of the same target can have very different appearances. (ii) two cameras looking at the same target can have widely differing viewpoints (top vs. side). Such common cases demonstrate the practical challenges with PTZ camera systems and reveal the lack of useful appearance information that can be transferred between cameras. We therefore adopt a novel approach to learn object associations across PTZ views in a master-slave configuration.

In a typical surveillance setting, an intuitive way for the security operator (controlling a PTZ camera with a joystick) to automatically initialize a target tracker would be to loosely follow the person of interest and and then have the system *automatically* learn the intended target, and then continue tracking the target. Since the system now has a robust model that has been automatically learned using the

*Appears in *IEEE Conference on Computer Vision and Pattern Recognition* (CVPR), Colorado Springs, CO, June 2011.

target's persistent appearance features, it can use this model to reliably track the target in future frames. Extending this idea to a master-slave camera configuration where the master camera actively tracks the target, the slave camera can follow the same target by using a registration mechanism (to roughly point to the same region as the master) to obtain a view of the same target, and then use these frames to learn a robust target model. Since it only passively follows the target and bootstraps itself from its own incoming imagery, this effectively circumvents the need to perform any transferring of appearance information across cameras. To passively learn the target model from this setup, we design a novel logistic-based MIL algorithm based on the softmax combining function and derive the gradient-based formulation for it. This algorithm uses covariance features to model image regions and we further develop this within a Gaussian MAP (maximum a-posteriori) estimation framework to obtain a robust, regularized estimate of the target model.

2. Related Work

Appearance-based association of objects across views have been popular since the work of [13] and [7] where they used color histograms to match people and cars, and [16, 11] who used Gaussian-based color models for matching. Improving upon these, an inter-camera color calibration approach was proposed by [18] in which a Brightness Transfer Function (BTF) is learned to transfer appearance histograms across views since different color and illumination distributions are observed across cameras (especially enhanced in outdoor scenes). However, these BTFs are typically not robust in real settings and are affected by parameters of scene geometry, exposure, focal length, and aperture size. An improvement to this approach is proposed by [8] by learning a subspace of these BTFs from training data, but this technique is still specific to world point surface material properties by assuming that the same 3D point is viewable from multiple views. For multiple reasons as explained before, this need not be the case with PTZ camera systems. In order to overcome these issues, the proposed algorithm takes a novel approach for performing camera handoff by using the slave camera to automatically learn the intended target models.

In the area of Multiple Instance Learning, since the original work of [5], there have been various algorithms proposed such as Diverse Density (DD) [15] and SVM techniques [1]. MIL has also been demonstrated for use in object tracking in [2, 17]. The state-of-the-art MIL-based tracking algorithm [2] uses an online boosting framework similar to [24] by choosing a succession of weak classifiers using Haar-like features to build an additive strong classifier. This approach requires a manual initialization and they demonstrate their tracking with indoor datasets. While Haar-like features are popularly used in typical object detection problems, in our approach we use covariance descriptors [19] to model regions, since they have been shown to be robust for outdoor pedestrian tracking scenarios typical in surveillance settings. More recently, in a comparison study of MIL algorithms, Multiple instance logistic regression [22] has been shown to be empirically superior, especially for image retrieval tasks. In this work, we propose a novel logistic-based MIL approach using the softmax combining function and derive a gradient based optimization framework.

3. MIL Framework

In this section we develop the theoretical framework used in our approach based on Multiple Instance Learning.

In our problem formulation, we model images containing the target of interest as bags, and patches within the images as instances. In order to learn the target model from this sequence of images and localize the target of interest within a new image, we wish to build a discriminative classifier which can output the probability $p(y = 1|x)$ indicating the posterior probability that the target is present $(y = 1)$ in the image patch x. In a MIL framework, the input data is obtained in the form of positive bags (B^+) and negative bags (B^-) containing instances. More formally, the input data $D = \{(X_1, y_1), (X_2, y_2), ..., (X_n, y_n)\}$ where $X_i = \{x_{i1}, x_{i2}, ..., x_{im_i}\}$ denotes bag i containing m_i instances and has a corresponding bag label $y_i \in \{0, 1\}$. Each instance x_{ij} is a feature vector calculated for an image patch j from bag i. Using the definition of [5], a bag is labeled positive if it contains *at least one* positive instance, and negative if it contains *all* negative instances.

Using a likelihood formulation, the correct bag classifier/labeler will maximize the log likelihood of labels over all the bags (given the MIL constraints)

$$\log \mathcal{L} = \sum_{i}^{n} \log p(y_i|X_i) \tag{1}$$

where $p(y_i|X_i)$ is the probability of the bag i (given its instances) having label y_i. Since the above likelihood formulation is expressed in terms of bag probabilities and what we want is to learn an instance-level classifier (for an instance/patch x), we will use a combining function to assemble instance-level probabilities into a bag probability. This is done using the *softmax* combining function as follows.

From the definition of positive and negative bags, we can formally express the notion of bag label in terms of its instance labels as $y_i = \max_j(y_{ij})$ which states that the label of a bag is the label of the instance within it which has the highest label. Here, we incorporate a probabilistic approximation of the max operator called *softmax*, in order to combine these instance probabilities in a smoother way, so as to allow all instances to contribute to the bag label. This

softmax function is generically defined as

$$softmax(a_1, ..., a_m) = \frac{\sum_{j=1}^{m} a_j \exp(\alpha a_j)}{\sum_{j=1}^{m} \exp(\alpha a_j)} \quad (2)$$

where α is a constant that controls the weighting within the *softmax* function such that *softmax* calculates the mean when α=0 and max when $\alpha \to \infty$. Since this *softmax* function is also differentiable (as opposed to the max operator), this allows us to incorporate it into a gradient-based optimization framework.

The bag-level probabilities for positive and negative bags are now defined as: p_i=$p(y_i$=1$|X_i)$ =softmax$(s_{i1}, ..., s_{im_i})$ and $p(y_i$=0$|X_i)$ =1 $- p(y_i$=1$|X_i)$, where s_{ij}=$p(y_{ij}$=1$|x_{ij})$ are the instance level probabilities being combined to obtain the bag probabilities $p(y_i|X_i)$. Thus, if one of the instances is very likely to be positive, the nature of the *softmax* combining function is such that its estimate of the bag's "positive-ness" will be very high, since it gives an exponentially higher weight to such an instance, and consequently the weighted average of all the instances will also be high. Here, α controls the proportion of instances in the bag that influence the bag label. Therefore, if one has an estimate of the proportion of positive to negative instances in the positive bags (noise-level), one can appropriately tune α to reflect this.

Next, to model these instance-level probabilities s_{ij}, we employ a logistic formulation given as

$$s_{ij} = p(y_{ij} = 1|x_{ij}) = \frac{1}{1 + \exp(-\mathbf{w} \cdot x_{ij})} \quad (3)$$

where the parameter vector $\mathbf{w}$ (to be learned) models the target of interest, so that the probability $p(y_{ij} = 1|x_{ij})$ calculated with Eqn. 3 would be high for an image patch x_{ij} that contains the target, and low for a patch that does not contain the target.

3.1. Maximum a-Posteriori Estimation

Using Eqn. 3 and the bag-level probabilities in Eqn. 1, we can express a maximum likelihood formulation (in terms of the parameter vector $\mathbf{w}$ to be learned) as

$$\hat{\mathbf{w}}_{ML} = \arg\max_{\mathbf{w}} \log p(D|\mathbf{w}) \quad (4)$$

with the log likelihood term $p(D|\mathbf{w})$ given by

$$\log p(D|\mathbf{w}) = \sum_{i \in B^+} \log \left(\frac{\sum_{j=1}^{m_i} s_{ij} t_{ij}}{\sum_{j=1}^{m_i} t_{ij}} \right) + \sum_{i \in B^-} \log \left(1 - \frac{\sum_{j=1}^{m_i} s_{ij} t_{ij}}{\sum_{j=1}^{m_i} t_{ij}} \right) \quad (5)$$

where s_{ij} and $t_{ij} = \exp(\alpha \cdot s_{ij})$ are functions of $\mathbf{w}$. By incorporating the bag labels y_i, this can be written as

$$\log p(D|\mathbf{w}) = \sum_{i=1}^{n} y_i \log p_i + (1 - y_i) \log(1 - p_i) \quad (6)$$

The maximum likelihood estimate $\hat{\mathbf{w}}_{ML}$ can be found by optimizing the above equation. However, in order to avoid overfitting of $\mathbf{w}$ and obtain a more regularized estimate, we can impose a prior on $\mathbf{w}$. We assume a Gaussian prior with zero mean $\mathbf{w} \sim \mathcal{N}(0, \mathbf{\Lambda^{-1}})$ and the covariance matrix $\mathbf{\Lambda}$ is a diagonal matrix which controls the individual weight values.

Employing this prior into the ML formulation, we can obtain the Maximum a-posteriori (MAP) estimate of $\mathbf{w}$ using Bayes rule as

$$\hat{\mathbf{w}} = \arg\max_{\mathbf{w}} (\log p(D|\mathbf{w}) + \log p(\mathbf{w})) \quad (7)$$

In terms of the likelihood and prior information, this function to be optimized can be written as

$$\mathcal{F}(\mathbf{w}) = \sum_{i=1}^{n} y_i \log p_i + (1 - y_i) \log(1 - p_i) - \frac{\mathbf{w}^T \mathbf{\Lambda} \mathbf{w}}{2} \quad (8)$$

which can be solved using gradient descent so that the weight vector update is performed as $\mathbf{w}_{new} = \mathbf{w}_{old} - \mathbf{u}(\nabla \mathcal{F}^T \mathbf{u})/(\mathbf{u}^T \mathbf{H} \mathbf{u})$, where $\mathbf{u} = -\mathbf{H}^{-1} \nabla \mathcal{F}$ is used as the search direction for line searches. The gradient for this objective is calculated as

$$\nabla \mathcal{F}(\mathbf{w}) = \sum_{i=1}^{n} \frac{\partial p_i}{\partial \mathbf{w}} \left(\frac{y_i}{p_i} + \frac{y_i - 1}{1 - p_i} \right) - \mathbf{\Lambda} \mathbf{w} \quad (9)$$

where the partial derivative term $\frac{\partial p_i}{\partial \mathbf{w}}$ is obtained as

$$\frac{\partial p_i}{\partial \mathbf{w}} = \frac{1}{(\sum_{j=1}^{m_i} t_{ij})^2} \left[\sum_{j=1}^{m_i} t_{ij} \left(\sum_{j=1}^{m_i} s_{ij} \partial(t_{ij}) + t_{ij} \partial(s_{ij}) \right) - \sum_{j=1}^{m_i} \partial(t_{ij}) \sum_{j=1}^{m_i} s_{ij} t_{ij} \right] \quad (10)$$

and the partial derivatives of s_{ij} and t_{ij} are given as (see Appendix for proofs),

$$\partial(s_{ij}) = s_{ij} \cdot (1 - s_{ij}) \cdot x_{ij} \quad (11)$$

$$\partial(t_{ij}) = \alpha \cdot t_{ij} \cdot s_{ij} \cdot (1 - s_{ij}) \cdot x_{ij} \quad (12)$$

and the Hessian $\mathbf{H}$ can be calculated by differentiating the gradient $\nabla \mathcal{F}$ w.r.t $\mathbf{w}$. However, since calculating the Hessian at every iteration and inverting it for the update step

is computationally slow, based on the work of [3], the changing Hessian can be approximated by a fixed Hessian $\tilde{\mathbf{H}} = -\frac{1}{k}\mathbf{X}\mathbf{X}^T - \mathbf{\Lambda}$ that needs to be inverted only once, and convergence is guaranteed as long as k is picked suitably so that $\tilde{\mathbf{H}} \leq \mathbf{H}$. In our experiments, we set $k = 10$.

3.2. Optimization of Prior Parameters $\mathbf{\Lambda}$

To select the optimal parameters $\mathbf{\Lambda}$ for the Gaussian prior on $\mathbf{w}$, we perform type-II maximum likelihood estimation and maximize the marginal likelihood of $p(D|\mathbf{\Lambda})$ as

$$\hat{\mathbf{\Lambda}} = \arg\max_{\Lambda} p(D|\mathbf{\Lambda}) = \arg\max_{\Lambda} \int p(D|\mathbf{w})p(\mathbf{w}|\mathbf{\Lambda})d\mathbf{w} = \arg\max_{\Lambda} \int \exp(\phi(\mathbf{w}))d\mathbf{w} \quad (13)$$

where $\phi(\mathbf{w}) = \log p(D|\mathbf{w}) + \log p(\mathbf{w}|\mathbf{\Lambda})$. Using a second-order Taylor series approximation of ϕ, we express this as

$$\phi(\mathbf{w}) \approx \phi(\hat{\mathbf{w}}) + \frac{1}{2}(\mathbf{w} - \hat{\mathbf{w}})\mathbf{H}(\hat{\mathbf{w}}, \mathbf{\Lambda})(\mathbf{w} - \hat{\mathbf{w}})^T \quad (14)$$

Using this and the Gaussian prior on $p(\mathbf{w}|\mathbf{\Lambda})$, the log of the marginal likelihood from Eqn. 13 can be written as

$$\begin{aligned} &\log p(D|\mathbf{\Lambda}) \\ &\approx \log\left(p(D|\hat{\mathbf{w}})p(\hat{\mathbf{w}}|\mathbf{\Lambda})(2\pi)^{d/2}|-\mathbf{H}^{-1}(\hat{\mathbf{w}}, \mathbf{\Lambda})|^{1/2}\right) \\ &\approx \log p(D|\hat{\mathbf{w}}) - \frac{1}{2}\left(\hat{\mathbf{w}}^T\mathbf{\Lambda}\hat{\mathbf{w}} - \log|\mathbf{\Lambda}| + \log|-\mathbf{H}(\hat{\mathbf{w}}, \mathbf{\Lambda})|\right) \end{aligned} \quad (15)$$

Now, to obtain the parameters $\mathbf{\Lambda}$ we optimize this log likelihood using gradient descent, the gradient for which is calculated as follows

$$\begin{aligned} \frac{\partial p(D|\mathbf{\Lambda})}{\partial \mathbf{\Lambda}} &= -\frac{1}{2}\left(\hat{\mathbf{w}}\hat{\mathbf{w}}^T - \mathbf{\Lambda}^{-1} + \mathbf{H}^{-1}(\hat{\mathbf{w}}, \mathbf{\Lambda})\right) \\ \frac{\partial p(D|\mathbf{\Lambda})}{\partial \lambda_\mathbf{i}} &= -\frac{1}{2}\left(\hat{w}_i^2 - {\lambda_i}^{-1} + H_{ii}^{-1}\right) \end{aligned} \quad (16)$$

where $diag(\mathbf{\Lambda}) = (\lambda_1, ..., \lambda_d)$. Setting this gradient to zero, we get an update rule for updating the parameters in the kth iteration as follows

$$\lambda_i^{k+1} = 1/({w_i^k}^2 + {H_{ii}^{-1}}^k) \quad (17)$$

This idea of using nested optimization iterations (one for updating the weights $\mathbf{w}$ and one for the parameters of $\mathbf{\Lambda}$) is inspired by the Sparse Learning work of [23]. Therefore, when presented with a set of training image bags, with patches within it forming instances, the parameter vector $\hat{\mathbf{w}}$ obtained from the final optimization algorithm represents the learned target model. Further, when presented with a new image, the probability that an image patch within it (with feature vector x) contains the learned target can be calculated from Eqn. 3 using $\hat{\mathbf{w}}$. The complete MIL based optimization framework is summarized in Algorithm 1.

Input: $\{X_i, y_i\}_1^n$ are the bags where $\{x_{ij} \in \mathbb{R}^d\}_{j=1}^{m_i}$ are the instances in bag i

---

Initialize λ_i=1 and w_i=0 for $i = 1, ..., d$.

---

repeat iterate k

  Calculate Hessian Matrix $\mathbf{H}$

  Find MAP estimate of $\mathbf{w}$ using $\mathbf{\Lambda}_k$

  ---

  repeat iterate l

    Compute gradient $\nabla\mathcal{F}(\mathbf{w})$

    Find direction of line search $\mathbf{u} = -\mathbf{H}^{-1}\nabla\mathcal{F}$

    Update $\mathbf{w} \leftarrow \mathbf{w} - \mathbf{u}(\nabla\mathcal{F}^T\mathbf{u})/(\mathbf{u}^T\mathbf{H}\mathbf{u})$

  until $\max_i |w_i^l - w_i^{l-1}| < \epsilon_1$;

  Update $\mathbf{\Lambda}$ parameters using $\lambda_i^k = 1/({w_i^k}^2 + {H_{ii}^{-1}}^k)$

until $\max_i |\log\lambda_i^k - \log\lambda_i^{k-1}| < \epsilon_2$;

Output: Weight vector $\hat{\mathbf{w}}$ representing the learned target model

---

Algorithm 1: The proposed MIL optimization framework

3.3. Covariance Features

We adopt a variation of covariance matrix features to obtain a feature vector x_{ij} for each image patch instance in our formulation (though other features could also be used). Covariance features have been shown to be robust appearance-based descriptors for modeling image regions [19]. The covariance matrix representation C_R for a given image patch R of size $W \times H$ in our framework is calculated as

$$C_R = \frac{1}{WH}\sum_{k=1}^{WH}(f_k - \mu_R)(f_k - \mu_R)^T \quad (18)$$

where f_k = $[x\ y\ r\ g\ b\ I_x\ I_y]$ is a 7 dimensional feature vector using a combination of position, color, and gradient values at each pixel location in the image patch R, and μ_R is the mean feature vector within the image patch.

We require a Euclidean distance based feature representation for Eqn. 3 whereas distances between covariance matrices are based on their eigenvalues [19]. Therefore, we use the property from [10] that eigenvalues of matrix logarithm of a covariance matrix C_R are equal to logarithms of eigenvalues of C_R. Therefore, the covariance matrix descriptor can be transformed to a feature vector representation by first calculating the matrix logarithm of C_R to obtain C_l and then stringing out the elements of the matrix C_l to obtain a vector C_v [10]. Moreover, since the matrix logarithm C_l is a symmetric matrix, it is fully specified by its bottom triangular part. Therefore, the feature vector C_v only needs to have the bottom triangular part of C_l, with the off-diagonal elements scaled by $\sqrt{2}$ to compensate for their double presence

in the matrix. In our case, the 7x7 dimensional covariance matrix reduces to a 28 dimensional feature vector. We then use these log covariance features to model the instances x_{ij} corresponding to each image patch.

4. Association Algorithm

While a master camera actively tracks a target, the slave camera uses the registration technique from [21] to point to the same area (we also introduce artificial pan-tilt jitter to demonstrate the applicability of our approach without accurate calibration). We then use these incoming frames from the slave camera to build image "bags" for MIL.

The first step in this approach is to extract image patch instances from these images and use them to construct positive and negative bags. To do this, we first detect regions of motion in each image by standard frame differencing (with the assumption that the target is moving). For each image, we then extract image patches from a reasonably large sample of the pixel locations marked as belonging to the motion region (the patch size can be predetermined or multiple sizes/aspect-ratios can be used). We construct a positive bag for this image using these instances since it is guaranteed to have at least one instance patch containing the desired target. Note that with this technique, instances corresponding to other parts of the scene in motion (trees, cars, noise pixels, etc.) would also be added, but that is acceptable since a positive bag can contain negative instances.

At the same time, we sample a similarly large number of pixel locations from the (non-moving) background and extract image patches from these locations to construct the corresponding negative bag. This method ensures that no instance in this bag will contain the target. We similarly repeat this process for each of the input frames. This way it is guaranteed that at least one instance corresponding to the desired target is present in *each* of the positive bags and at the same time, absent from *all* the negative bags, thus satisfying the Multiple Instance assumption. Once the positive and negative bags are constructed, we train the MIL classifier using Algorithm 1 to learn the target concept (represented by the weight vector $\hat{\mathbf{w}}$).

Online Update: An important aspect of the proposed learning approach is that the learned target concept can be updated in an online manner with each new incoming frame instead of having to retrain the classifier using all the bags collected from the beginning. We use the assumption that the target appearance does not change much with the new frame and hence would result in only a small change in the target model. Once we receive a new incoming frame, we create a positive and negative bag using the method described above and run Algorithm 1, but this time with the weight vector $\mathbf{w}$ initialized to the previously learned target model. Once the algorithm converges, we obtain the new weight vector reflecting the updated target model.

Figure 1. Top row - Image sequence input to MIL. Bottom row - Probability surfaces overlaid on new frames showing learned target (best viewed in color).

Multiple Targets: It could be the case that more than one target is present across all input frames. Therefore, the proposed approach tests for multiple targets by using the first learned concept to remove all corresponding target instances from the positive bags, and then retrains to learn the next strongest concept, and so on for each remaining target. More specifically, once the algorithm converges and learns the first target model, it then uses Eqn. 3 with this model for every instance from every positive bag to calculate its probability of being the target. It then classifies each instance as positive (target) if this probability lies above a fixed threshold σ. We then update every positive bag by removing from it all the instances classified as being positive. This ensures that none of the positive bags contain even a single instance corresponding to the learned target.

It is important that the threshold σ be set conservatively so as to eliminate every true positive instance, even at the cost of eliminating a few false positives if necessary (we set σ=0.75). Since all instances corresponding to the first target have now been removed, re-running the optimization algorithm learns the next strongest target concept (if such a valid target is present). This process is repeated until all valid target concepts have been learned. To test whether a valid target concept has been learned or not, we calculate the log likelihood of the learned model on the input set of positive and negative bags. An extremely low value of the combined likelihood over all the bags indicates that the learned model is degenerate and there was no target concept left to learn.

5. Experiments

In this section, we present multiple experiments to evaluate the proposed approach in different settings and compare its performance with other techniques.

5.1. Extracting a Persistent Target

We first performed an experiment to demonstrate the proposed algorithm by extracting a target that is persistent across all frames of an input sequence in a single camera scenario. This setting is similar to standard MIL-based problems such as image retrieval, where one attempts to

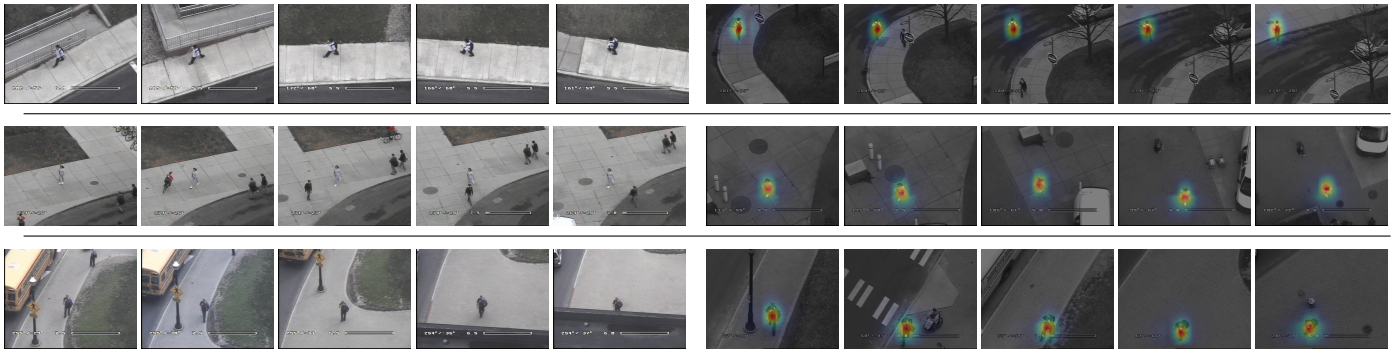

Figure 2. Probability surfaces overlaid on incoming frames showing target localized by slave camera for handoff for 3 different sequences.

learn models for objects (car, waterfall, etc.) that are common across the entire input dataset. In our case, given a sequence of input images, we first constructed a set of positive and negative bags according to the technique described in Sect. 4 with patch size of 75x25 pixels. We then ran our MIL algorithm to learn a concept corresponding to a target that was common across all positive bags and absent in each of the negative bags. The parameters of the learning algorithm were set as α=3, $\epsilon_{1,2}$=10^{-5}. The validity of the learned target concept was checked by calculating the likelihood of the learned model across all input bags.

Next, we tested additional new frames. The target model was evaluated against each new input image at every possible location. This results in a probability surface in each new frame indicating the probability of the target being present at each particular location. After this, a new pair of positive and negative bags was created for the new frame, the target model was updated using the online update technique described in Sect. 4, and the updated model then was used to evaluate the next incoming frame. This process was repeated with every new frame.

Figure 1(a) shows the sequence of input frames used to learn a target model. Figure 1(b) shows the probability "heatmap" overlaid on new frames, representing the probability surface $p(y = 1|x)$ for each patch x across the image. As seen in the figure, the target present across all training images was detected by the algorithm. Figure 1(b) shows the results of target localization using model update in the new frames. Notice also that the other person was not learned by the algorithm since that person was not present in all of the frames, thus not satisfying the MIL constraints.

5.2. Associating Targets for Handoff

The next set of experiments were performed to demonstrate and evaluate the proposed approach to perform camera handoff with multiple PTZ camera settings. Our system consisted of a master PTZ camera actively tracking a target and a slave camera (also PTZ) passively following the same target by roughly pointing to the same area. The proposed algorithm was then run on the incoming frames from the slave camera to identify and learn the target being tracked, so that the system could then handoff to the slave camera which could then take over the active tracking.

While the geo-registration of the cameras can be used to drive the slave PTZ camera to accurately orient and point to the intended target being tracked by the master, to demonstrate that our approach does not require accurate geometric calibration of the cameras, we artificially added Gaussian jitter to the pan-tilt motor control of the slave camera so that the target could be present anywhere in the scene (not necessarily at the image center). The results with different jitter levels for three different sequences are shown in Fig. 2. In each case, it was observed that the target model was successfully learned from the input frames thus providing the active tracker the ability to perform handoff to the slave camera.

The online update capability of the algorithm allows the system to continue updating all the target models (if there are multiple targets present in the scene satisfying the MIL criterion), and continue this process until a *unique* target is detected in the scene. This is possible because the objective function evaluation from the optimization algorithm can be used to identify the number of target models learned. Thus, we can exploit this feature to continuously update multiple target models until only the single most persistent target remains in the scene and then use that model for active tracking. We demonstrate this capability in an experiment with the next sequence.

As seen in Fig. 4(top row), there were 2 targets present in the scene satisfying the MIL criterion. Therefore, the algorithm learned 2 corresponding target models and these were then used for localization with the incoming frames from the slave camera as shown in Fig. 4(bottom) (note that a probability surface corresponding to each learned model is obtained separately, but they are shown here overlaid on the same image). Further, with each new incoming frame, the target models were updated online and eventually, when incorporating a frame where one of the targets is no longer

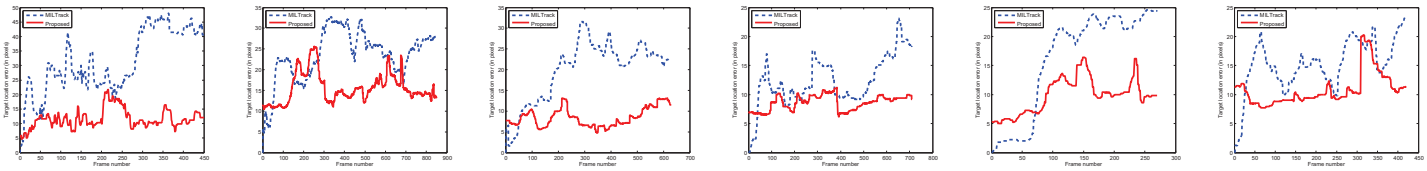

Figure 3. Comparison of target tracking accuracy between the proposed approach and MILTrack for 6 different sequences.

Figure 4. Target localization with multiple targets in the scene.

present, the MIL criterion is violated and consequently the only single remaining target model was updated, as shown in the last two frames in Fig. 4. Once this happens, the slave camera can take over control and use the learned model to bootstrap its active tracking. Note that even if the other target later re-enters the scene, they will not be localized, as the MIL criterion requires the target object to be persistent across all frames from the beginning (or is manually reset).

5.3. Quantitative Evaluation

We next evaluated the proposed approach quantitatively based on accuracy of target localization and compared its approach with MILTrack [2] on 6 PTZ tracking sequences. We also experimented with other trackers such as IVT [20], but it was difficult to compare performance directly since they require parameter tuning for every video sequence as noted in [2].

For MILTrack, we use a search radius s of 100 pixels and extract positives instances from the slave camera frames using a positive radius r of size 5 pixels. Note that while MILTrack requires a manual initialization of the target, the proposed approach learns this automatically from an initial set input frames. Therefore, in all the experiments, we fixed the number frames used to initialize and learn the target model at 25. The frame immediately following this is considered frame 0 and also the one used to initialize MILTrack manually. The target localization results are compared using all the following frames.

A comparison of the accuracy of the tracking for 6 different sequences is shown in Fig. 3. As noted before, MILTrack has mostly been evaluated with indoor sequences so its performance in typical surveillance videos is not well known. Note that MILTrack always starts with a tracking error of 0 in the first frame due to manual initialization, which

Table 1. Comparison of tracking accuracy in ($\mu \pm \sigma$) pixel error

Sequence	Proposed	MILTrack
1	11.6 ± 3.2	31.1 ± 10.8
2	15.5 ± 3.6	23.5 ± 6.1
3	8.4 ± 2.2	19.9 ± 7.3
4	8.6 ± 1.2	12.4 ± 4.3
5	9.8 ± 3.0	16.1 ± 8.6
6	10.8 ± 2.8	15.2 ± 4.4

is not the case with the proposed approach. As seen from the overall results, the tracking results from the proposed approach performs better overall compared to the state-of-the-art MIL-based approach. Table 1 shows a comparison the mean and standard deviation values for the tracking error between the two approaches. We observed that the proposed approached performed approximately $30 - 40\%$ better than MILTrack. We also observed that in most of the sequences, the proposed approach maintains a fairly constant tracking accuracy (low values of σ) and does not seem to deteriorate with the number of frames, in contrast with MILTrack.

Bootstrapping performance: The proposed approach works by using an initial set of frames to bootstrap and learn the target model, and then uses an online strategy to update the model with every new frame. Since the quality of model learned would depend on the training set of frames, we next performed a set of experiments to evaluate the tracking performance with respect to the number of initial frames used for bootstrapping the approach.

For each of our sequences, we ran the proposed approach multiple times, each time with an increasing number of initial frames. Once the target model was learned, we then used the online update and localized the target in each new frame. The tracking errors were noted and compared for different number of initial frames used for bootstrapping. The results are shown in Fig. 5. For the sake of clarity, we show the results for 4 of the 6 sequences which spanned the range of errors observed. The results for the other 2 were within this error range. As expected, we observed that for each of the sequences, on average the tracking error reduces with the number of initial frames used to learn the target model, although not monotonically (remains constant in some cases). Also as seen in Fig. 5, it was observed that the tracking error flattens out for $\geq$50 initial frames (without

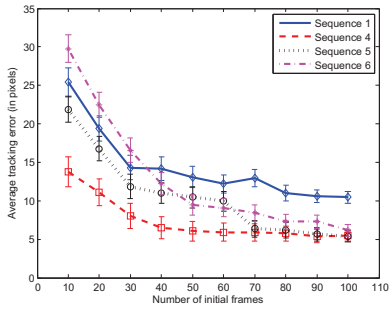

Figure 5. Comparison of tracking performance with increasing number of frames used for bootstrapping.

a significant improvement in performance). This analysis would be useful to tune the proposed approach for accuracy requirements in varying tracking scenarios.

6. Summary and Future Work

We proposed a novel approach to the problem of establishing object correspondences and target modeling for common tasks such as camera handoff in wide-area surveillance settings. In our approach, the slave PTZ camera passively follows the target and bootstraps itself from its incoming imagery. This problem is set in a Multiple Instance Learning framework where the input image frames are used to build positive and negative bags and the intended target model is learned by exploiting the idea of "commonality" across all input frames. To achieve this, we developed a novel MIL-based formulation based on the logistic model and used the softmax function to combine instance probabilities to bag probabilities. We employed covariance-based region features to model instances and learned target models using a MAP estimation framework. We evaluated our approach with multiple PTZ camera sequences in outdoor surveillance settings and showed promising results in comparison with the state-of-the-art MIL-based approach. In future work, we plan to study trajectory analysis and also explore other interesting applications for our theoretical work.

Acknowledgement: We gratefully acknowledge the support of the U.S. Department of Energy through the LANL/LDRD Program under LDRD-DR project 'RADIUS' for this work. We also thank Matt Nedrich for all the valuable discussions.

Appendix

$$\begin{aligned}\frac{\partial s_{ij}}{\partial \mathbf{w}} &= \frac{\partial}{\partial \mathbf{w}}\left(\frac{1}{1+\exp(-\mathbf{w}\cdot x_{ij})}\right)\\ &= \frac{\exp(-\mathbf{w}\cdot x_{ij})}{(1+\exp(-\mathbf{w}\cdot x_{ij}))^2}\cdot x_{ij}\\ &= s_{ij}\cdot(1-s_{ij})\cdot x_{ij}\end{aligned}$$

$$\begin{aligned}\frac{\partial t_{ij}}{\partial \mathbf{w}} &= \frac{\partial}{\partial \mathbf{w}}\left(\exp(\alpha\cdot s_{ij})\right)\\ &= \alpha\cdot\exp(\alpha\cdot s_{ij})\cdot\partial s_{ij}\\ &= \alpha\cdot\exp(\alpha\cdot s_{ij})\cdot s_{ij}\cdot(1-s_{ij})\cdot x_{ij}\\ &= \alpha\cdot t_{ij}\cdot s_{ij}\cdot(1-s_{ij})\cdot x_{ij}\end{aligned}$$

References

[1] S. Andrews and T. Hofmann. Support vector machines for multiple instance learning. In *Proc. NIPS*, 2003. 3434

[2] B. Babenko, M. Yang, and S. Belongie. Visual tracking with online multiple instance learning. In *CVPR*, 2009. 3434, 3439

[3] D. Bohning. The lower bound method in probit regression. In *Computational Statistics and Data Analysis*, 1999. 3436

[4] Q. Cai and J. Aggarwal. Tracking human motion in structured environments using a distributed camera system. In *TPAMI*, 1999. 3433

[5] T. Dietterich and T. Lozano-Perez. Solving the multiple-instance problem with axis-parallel rectangles. In *Artificial Intelligence*, 1997. 3434

[6] W. Hu, M. Hu, X. Zhou, T. Tan, J. Lou, and S. Maybank. Principal axis-based correspondence between multiple cameras for people tracking. In *TPAMI*, 2006. 3433

[7] T. Huang and S. Russell. Object identification in a bayesian context. In *Proc. IJCAI*, 1997. 3434

[8] O. Javed, K. Shafique, and M. Shah. Appearance modeling for tracking in multiple non-overlapping cameras. In *CVPR*, 2005. 3433, 3434

[9] Y. Jo and J. Han. A new approach to camera hand-off without camera calibration for the general scene with non-planar ground. In *ACM Multimendia Wkshp. on VSSN*, 2006. 3433

[10] J. Jost. Riemannian geometry and geometric analysis,. In *Berlin: Springer-Verlag*, 2002. 3436

[11] J. Kang, I. Cohen, and G. Medioni. Continuous tracking within and across cameras. In *CVPR*, 2003. 3434

[12] S. Khan and M. Shah. A multiview approach to tracking people in crowded scene using a planar homography constraint. In *Proc. IEEE ECCV*, 2006. 3433

[13] J. Krumm, S. Harris, and S. Shafer. Multi-camera multi-person tracking for easyliving. In *Intl. Wkshp on Visual Surveillance*, 2000. 3434

[14] D. Makris, T. Ellis, and J. Black. Bridging the gaps between cameras. In *CVPR*, 2004. 3433

[15] O. Maron and T. Lozano-Perez. A framework for multiple instance learning. In *Proc. NIPS*, 1998. 3434

[16] A. Mittal and L. Davis. M2tracker: A multi-view approach to tracking people in a clustter scene using region-based stereo. In *Proc. IEEE ECCV*, 2002. 3434

[17] J. Mu, L. Kwok and L. Bao-Liang. Online multiple instance learning with no regret. In *CVPR*, 2010. 3434

[18] F. Porikli. Inter-camera color calibration using crosscorrelation model function. 2003. 3433, 3434

[19] F. Porikli, O. Tuzel, and P. Meer. Covariance tracking using model update based on means on Riemannian manifolds. In *CVPR*, 2006. 3434, 3436

[20] D. Ross, J. Lim, and M. Yang. Incremental learning for robust visual tracking. In *IJCV*, 2008. 3439

[21] K. Sankaranarayanan and J. Davis. A fast linear registration framework for multi-camera gis coordination. In *AVSS*, 2008. 3437

[22] B. Settles, M. Craven, and S. Ray. Multiple-instance active learning. In *Proc. NIPS*, 2007. 3434

[23] M. Tipping. Sparse bayesian learning and the relevance vector machine. *JMLR*, 1:211–244, 2001. 3436

[24] P. Viola, J. Platt, and C. Zhang. Multiple instance boosting for object detection. In *Proc. NIPS*, 2005. 3434