2010 Fourth Pacific-Rim Symposium on Image and Video Technology

Object Tracking based on Covariance Descriptors and On-Line Naive Bayes Nearest Neighbor Classifier

Pedro Cortez Cargill*, Domingo Mery Quiroz* and Luis Enrique Sucar†
**Departamento de Ciencias de la Computación, Pontificia Universidad Católica de Chile*
Av.Vicuña Mackenna 4860 (143), Santiago, Chile
Email: pmcortez@uc.cl, dmery@ing.puc.cl
†*Instituto Nacional de Astrofísica, Óptica y Electrónica de México*
Luis Enrique Erro #1, Tonantzintla, Puebla, México
Email: esucar@inaoep

***Abstract*—Object tracking in video sequences has been extensively studied in computer vision. Although promising results have been achieved, often the proposed solutions are tailored for particular objects, structured to specific conditions or constrained by tight guidelines. In real cases it is difficult to recognize these situations automatically because a large number of parameters must be tuned. Factors such as these make it necessary to develop a method robust to various environments, situations and occlusions. This paper proposes a new simple appearance model, with only one parameter, which is robust to prolonged partial occlusions and drastic appearance changes. The proposed strategy is based on covariance descriptors (which represent the tracked object) and an on-line nearest neighbor classifier (to track the object in the sequence). The proposed method performs exceptionally well and reduces the average error (in pixels) by 47% compared with tracking methods based on on-line boosting.**

***Keywords*-Object Tracking; Naive Bayes; Covariance Descriptors; On-line Model**

I. Introduction

Object tracking of pedestrians, faces and generic objects has been extensively studied in computer vision. High performance has been achieved [1], [2] but proposed algorithms are highly structured and often restrictive. Generally they are directly related to the type of object being tracked (training off-line [3]) or the environment in which the object is located (background subtraction [4]). Thus, object tracking under practical conditions is still an unsolved problem.

In general terms, a tracking system has three components: *i)* image representation, *ii)* appearance model and *iii)* motion model. Currently a variety of methods exist to create an appearance model and many use statistical information which is either defined manually or trained during an initial input [5]. These methods are simple but they often encounter difficulty when the tracked object's appearance changes significantly within the video sequence. This led to the development of specific methods that were able to adapt the model to appearance changes, they are known as *Adaptive Appearance Models* [6], [7]. In another method, the appearance model is subject to weak classifiers trained on-line which detect the followed object and/or the background. This method, known as *Tracking by Detection* [8], [9], [10], must balance between the tracking's stability and classifiers' learning speed.

The most challenging problem in updating appearance models is selecting new positive and negative samples. Babenko et al. proposed an optimal sampling methodology using classifiers based on an appearance model [8]. This model uses an on-line boosting method to select the best features to track in the object. Although this system achieves high performance, tracking may fail when a prolonged partial occlusion occurs because the balance between stability and learning in this case is still an unsolved problem.

This investigation mainly focuses on the appearance model and image representation. We propose a new simple appearance model which is robust to prolonged partial occlusions and drastic appearance changes. The proposed strategy is based on covariance descriptors representing the tracked object [11] as well as an on-line nearest neighbor classifier to track the object in the sequence.

II. Proposed Tracking System

The proposed tracking system includes the three components described above, image representation, appearance model and motion model. The image representation and motion model are constructed utilizing small variations of existing methods. However, the proposed appearance model is a new method called On-Line Naive Bayes Nearest Neighbor, which is robust to prolonged partial occlusions and drastic appearance changes.

A. Image Representation

Recently Tuzel *et al.* proposed a straightforward solution that integrates multiple features which are simple and quick to calculate such as gradient, color, position and intensity. The method can even integrate features from infrared or thermal cameras [11]. Compared to other descriptors such as Haar [12] or histogram gradients, this descriptor provides more information and is not restricted to the search window size like the others. The covariance features have been used in a variety of applications and several improvements have been proposed. Tuzel *et al.* and Yao *et al.* propose using covariance features with a LogiBoost classifier for pedestrian detection [13], [14]; Hu *et al.* proposed utilizing a modified particle filter, where weights are proportional to a specific measure applied to the covariance features [15]; Poriki *et al.* proposed an algorithm to track objects using the covariance features and Lie algebra to create an *Adaptive Appearance Model* [7] .

The covariance descriptor proposed by Tuzel *et al.* in [11], is formally defined as:

$$F(x, y, i) = \phi_i(I, x, y), \tag{1}$$

where I is an image (which can be RGB, black and white, infrared, etc.), F is a $W \times H \times d$ matrix, where W is the image width, H is the image height, d is the number of features used, and ϕ_i is a function that relates the image with the i-th feature, *i.e.*, the function that obtains the i-th feature from the image I. It is important to note that the features are obtained at pixel level.

The use of the covariance matrix as a descriptor has multiple advantages: 1) it unifies the target's spatial and statistical information; 2) it provides an elegant solution to merge different features and modalities; 3) it has a low dimensionality; 4) it is capable of comparing regions without fixed window size restraints because region size is irrelevant, the descriptor size is $d \times d$; and 5) the covariance matrix can be easily calculated for any region or sub-region.

The proposed method uses a 12 characteristic tensor F , which is defined by:

$$F(x, y, i) = [x \quad y \quad R \quad G \quad B \quad |I_x| \quad |I_y| \quad \dots \\ \sqrt{|I_x|^2 + |I_y|^2} \quad |I_{xx}| \quad |I_{yy}| \quad \tan^{-1}(\tfrac{I_x}{I_y}) \quad S]^\mathsf{T}, \tag{2}$$

where x and y represent the coordinates of each pixel; R, G, B represent the red, green and blue values of each pixel; $|I_x|, |I_y|, |I_{xx}|, |I_{yy}|$ represent the image intensity's first and second derivative in x and y; $\sqrt{|I_x|^2 + |I_y|^2}$ represents the second derivative's magnitude; $\tan^{-1}(\frac{I_x}{I_y})$ represents the derivative orientation and S represents a *Saliency Map* [16].

Regardless of the benefits a region's representation as a covariance matrix brings, the calculation for any window or area of an image is computationally prohibitive if using conventional methods. Tuzel *et al.* proposed a method to calculate the covariance matrix for any rectangular window or region of an image based on the integral image [11], [12].

Note that the proposed method only calculates the tensor of patches of interest as opposed to the whole image.

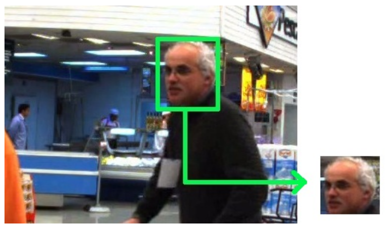

Figure 1. Normalization patches method. Note that this creates a new image while allowing the use of windows search with affine transformations.

Thus, we can use search windows with rotation or affine transformations. The process involves extracting the desired patch (in order to calculate the descriptor) from the original image, creating a new image, and transforming it into a rectangle of fixed size. As a corollary, another level of scale changes normalization was included (Figure 1).

Finally, the covariance descriptor is not an element of Euclidean space because it is a semi positive defined matrix (SPD^+). Therefore, the classical algorithms of machine intelligence such as neural networks, PCA, LDA, etc., cannot be used. SPD^+ matrices are included in Lie algebra or *Riemannian Manifolds* geometry [17] so in order to compare two covariance descriptors, we use the metric for semi positive definite matrices *Log-Euclidiana* [18], which is defined as:

$$\rho(\mathbf{X}, \mathbf{Y}) = \|log(\mathbf{X}) - log(\mathbf{Y})\| \tag{3}$$

where $log(\mathbf{X})$ is the logarithmic map of the covariance matrix, which is defined by the singular value decomposition of matrix $\mathbf{X}$. Let $SVD(\mathbf{X}) = \mathbf{U}\Sigma\mathbf{U}^\mathsf{T}$ be the singular value decomposition of $\mathbf{X}$, where $\mathbf{U}$ is an orthonormal matrix and $\Sigma = \text{diag}(\lambda_1, ..., \lambda_n)$ is a diagonal matrix of eigenvalues. Therefore, the logarithmic map is defined as:

$$log(\mathbf{X}) = \mathbf{U}[\text{diag}(log(\lambda_1), ..., log(\lambda_n))]\mathbf{U}^\mathsf{T} \tag{4}$$

This paper proposes a joint appearance model, mixing both adaptive and detection models, that uses a *Naive Bayes Nearest Neighbor* [19] classifier and covariance features which we call *On-Line Naive Bayes Nearest Neighbor* .

On-Line Naive Bayes Nearest Neighbor (ONBNN)

State of the art image classification methods require an intensive learning or training stage (using SVM, Boosting, etc.). In contrast, non-parametric Nearest-Neighbor (NN) based image classifiers require no training time and have a variety of favorable properties. However, the large performance gap between these two approaches rendered NN-based image classifiers useless. In defense of the NN algorithm Boiman *et al.* [19] proposes a trivial NN-based classifier with a performance that ranks among the top learning based image classifiers. This method employs

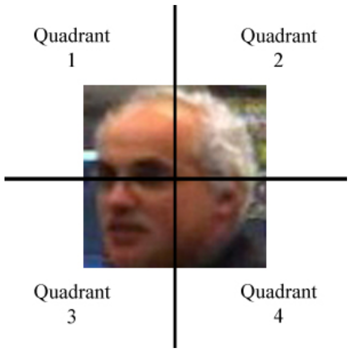

Figure 2. Distribution of the four quadrants for which the covariance descriptor is calculated.

NN-distances in the space of the local image descriptors (and not in the space of images). NBNN computes direct "Image to Class" distances without descriptor quantization.

The resulting NBNN classifier can be summarized by algorithm 1, where C represents the classes of the classifier, $\hat{C}$ is the estimated class and $NN_C(d_i)$ represents the descriptor in the specific class C, closer to d_i, according to nearest neighbors (NN). Note that the proposed tracking system has two cases, the tracked object and everything else.

Algorithm 1 NBNN [19]

```
1: Compute descriptors $d_1, \dots, d_n$ of the query image
2: $\forall d_i$ $\forall C$ compute the NN of $d_i$ in C: $NN_C(d_i)$
3: $\hat{C} = \arg\min_c \sum_{i=1}^{n} \|d_i - NN_C(d_i)\|^2$
```

Based on the idea of Boiman *et al.*, we designed an appearance model that classifies the patch of each new sample with a NBNN classifier. In the proposed method the NBNN classifier is updated in each iteration with new positive samples and each sample is represented by the patch's covariance features. This way, we build a "database" of the tracked object, which is necessary in the nearest neighbor approach used by NBNN.

Each sample is represented by four covariance features, where each descriptor represents one of the four image quadrants (Figure 2). Thus, the appearance model is defined as $M(f, s_f)$; where $f \in \{1, 2, 3, 4\}$ represents one of four quadrants, $s_f \in \{1, \dots, N\}$ is the number of samples for the specific f quadrant and $M(f, s_f)$ is the f quadrant covariance descriptor of the sample s_f. Note that the samples are specific to each quadrant, but they must be less than or equal to N.

Next we define $P(f)$ as a sample; where once again $f \in \{1, 2, 3, 4\}$ represents one of the four quadrants and $P(f)$ is the f quadrant's covariance descriptor. Therefore, the distance of a sample P to the model M is defined as:

$$d(M,P) = \sum_{k \in f} \min_{i \in s_f} \rho(M(k,i), P(k)) + \lambda \|\overline{M(k,i)} - \overline{P(k)}\| \quad (5)$$

where we use the metric defined in (3), since $M(f, s_f)$ and $P(f)$ are covariance matrices. Additionally, we add the Euclidean distance relative to the original image, between the quadrants centers. This way spatial information, which was lost by using the patch instead of the full picture, can be added. $\overline{M(k,i)}$ and $\overline{P(k)}$ are the patch's center position (x, y), with respect to the original image, which represents $M(k,i)$ and $P(k)$ respectively and λ is the relevant factor of the Euclidean distance in the model.

We defined a protocol to update the model, where the quadrant with the lowest average distance to the model M of the last tracked patch $P^B(f)$ is added. The average distance between the quadrant and the quadrant's model samples, is defined as:

$$\omega(M,P,f) = \frac{1}{|s_f|} \sum_{i \in s_f} \rho(M(f,i), P(f)) \quad (6)$$

where f is the quadrant to which the average distance is being calculated and $|s_f|$ is the number of samples for quadrant f. We also define $R(f, s_f)$ which maintains the average distance to the model when adding a new sample to the model (10).

Finally, the model is updated as follows:

$$f = \arg\min_{i \in f} \omega(M, P^B, i) \quad (7)$$

$$j_f = \begin{cases} |s_f| + 1 & \text{if } |s_f| < N \\ \arg\max_{i \in s_f} R(f,i) & \text{if } |s_f| = N \end{cases} \quad (8)$$

$$M(f, j_f) = P^B(f) \quad (9)$$

$$R(f, j_f) = \omega(M, P^B, f) \quad (10)$$

Specifically, equation (7) seeks the quadrant of P^B with the lowest average distance to the model; equation (8) describes how the index, where the sample (of the f quadrant) will be added to the model, is selected and equation (10) saves the sample's average distance to the model. This information is used to define the protocol update (8).

Because the model must be on-line, it is not possible to add an infinite number of samples so the number N is defined as the model's memory. In order to grant some flexibility when updating a quadrant, with N sample in memory, the sample with greater average distance is removed and the new sample is added. The distance that makes up the samples ranking is calculated when the new sample is added to the model. Thus, avoiding confusion from the prolonged partial occlusions in the model, we give a "threshold of learning" for stability.

The moment the model is initialized, four quadrants of the initial patch are added with an average distance to the model equal to zero. For more details refer to Algorithm 2. It receives a set of parameters describing a set of search windows, where (x_i, y_i) represents the center, (w_i, h_i) represents the width and height and θ_i represents the window's rotation. In our case, the algorithm delivers a parameter's configuration that gets less distance to the model, meaning any algorithm can be used (particle filter, Kalman filter, etc.) to estimate the next window.

Algorithm 2 On-line NBNN

```
Input: Dataset $\{x_i, y_i, w_i, h_i, \theta_i\}_{i=1}^N$, the model $M$ and the
    average distance stored $R$
 1: for $k = 1$ to N do
 2:     $I$ = getImage$(x_k, y_k, w_k, h_k, \theta_k)$ //Get warped image
 3:     $P_k$ = getCovarianceDescriptors$(I)$
 4:     $d_k = d(M, P_k)$ //from (5)
 5: end for
 6: $k^* = \arg\min_k d_k$
 7: $P^B = P_{k^*}$
 8: $f = \arg\min_{i \in f} \omega(M, P^B, i)$
 9: if $s_f < N$ then
10:     $j_f = |s_f + 1|$
11: else
12:     $j_f = \arg\max_{i \in s_f} R(f, i)$
13: end if
14: $M(f, j_f) = P^B(f)$
15: $R(f, j_f) = \omega(M, P^B, f)$
Output: $x_{k^*}, y_{k^*}, w_{k^*}, h_{k^*}, \theta_{k^*}$
```

B. Motion Model

To highlight the appearance model's real contribution, initially we use a very simple motion model to perform an exhaustive search at multiple scales (currently unused patches with affine transformations), in a smaller area, inside the last tracked patch.

Formally, let $O^{t-1}_{x,y,w,h}$ be the tracked patch at time $t-1$, with center x, y and dimensions w, h. We define the search area for the time t as $S^t_{x,y,w,h}$; where:

$$S^t_{x,y,w,h} = O^{t-1}_{x,y,\frac{w}{\alpha},\frac{h}{\alpha}} \tag{11}$$

In the area $S^t_{x,y,w,h}$ we search exhaustively from the patch $T^t_{x,y,w,h}$, where the $T^t_{x,y,w,h}$ centers are the coordinates of each pixel, within the search area $S^t_{x,y,w,h}$ and β_n is a list of scalars (Figure 3).

$$T^t_{x,yw,h} = S^t_{x_p,y_p w\beta_n, h\beta_n} \tag{12}$$

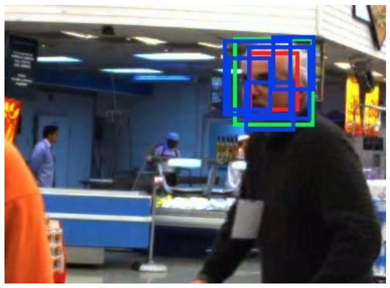

Figure 3. Motion model description. The green rectangle represents the last tracked patch $O^{t-1}_{x,y,w,h}$, the red rectangle represents the search area $S^t_{x,y,w,h}$ and the blue rectangles represent samples $T^t_{x,y,w,h}$.

Finally, at moment t, the patch $T^t_{x_n,y_n,w_n,h_n}$ with smaller distance to the appearance model is selected. Note that the motion model is completely independent of the appearance model.

III. Experiments

We tested our NBNN algorithm on a set of challenging public and private video sequences. For comparison, four different algorithms were used:

Online-AdaBoost (OAB): algorithm described in [9] and implemented in [8], with search radius set to 35 pixels.

SemiBoost tracker: algorithm described in [20], this method uses label information from the first frame only, and then updates the appearance model via on-line semi-supervised learning in subsequent frames. This makes it particularly robust to scenarios where the object leaves the scene completely. However, the model relies strongly on the prior classifier (trained using the first frame).

FragTrack: described in [5], this algorithm uses a static appearance model based on integral histograms, which have been shown to be very efficient. The appearance model is partially based on this algorithm which makes it robust to occlusions.

Online-MILBoost (OMB): algorithm described in [8], this method is a state of the art on-line boosting tracker with the learning rate for the weak classifiers set to 0.85 and the scalar for sampling negative examples set to 50. In our experiment we also used five variations of the algorithm OMB, where we set the labeling threshold for each sample r in 1, 2, 3, 4 and 5.

ONBNN: for our proposed method (ONBNN), the values of the parameters for *all experiments* were: $\lambda = 0.05$, $\alpha = 2.5$ y $\beta_n = \{0.8, 1, 1.1, 1.3\}$.

Video Clip	OAB	OMB 2	SemiBoost	Frag	ONBNN
Occluded Face	44	32.4	41	**6**	11.3
Occluded Face 2	21	22.8	43	45	**12.9**
Girl	48	33.3	52	27	**13.9**

Table I
AVERAGE CENTER LOCATION ERRORS (PIXELS). ALGORITHMS COMPARED ARE ONLINE-ADABOOST TRACKER [9] WITH $r = 1$, FRAGTRACK [5], SEMIBOOST TRACKER [20], OMB [8] WITH $r = 2$ AND ONBNN. BOLD RESULTS INDICATE BEST PERFORMANCE.

Video Clip	OMB 1	OMB 2	OMB 3	OMB 4	OMB 5	ONBNN
Surfer	4.9	7.7	13.4	5,5	14.6	**4.5**
Occluded Face	18.4	32.4	19.6	31.6	34.4	**11.3**
Occluded Face 2	30.7	22.8	16.7	14.3	16.5	**12.9**
Girl	31.6	33.2	33.8	34.9	26.5	**13.9**

Table II
AVERAGE CENTER LOCATION ERRORS (PIXELS). ALGORITHMS COMPARED ARE OMB [8] WITH $r = 1$, $r = 2$, $r = 3$, $r = 4$, $r = 5$ AND ONBNN. BOLD RESULTS INDICATE BEST PERFORMANCE.

Initially, we performed an experiment on four public video sequences (*Surfer, Occluded Face, Occluded Face 2 and Girl*), against five different configurations of the OMB algorithm. Where [8] provided a label of the object's ground truth center every five frames. Then we performed a second experiment on three video sequences (*Occluded Face, Occluded Face 2 and Girl*), against the three algorithms described above and the OMB best average configuration (obtained from the first experiment). In addition we also performed a third experiment on sequences of private videos, without comparison to the other four algorithms. This set of videos includes pedestrian tracking and a complex background.

Finally, to measure performance, we used the average center location errors (pixels), which calculates the distance between the centers of the real position and estimated position of the tracked object.

IV. RESULTS

In almost all experiments our ONBNN algorithm outperforms the other four algorithms. In the *Occluded Face* video, FragTrack performed the best because it is specifically designed to handle occlusions via a part-based model. However, on a similar, but more challenging video, *Occluded Face 2*, FragTrack performs poorly because it cannot handle appearance changes well. This highlights the advantages of using an adaptive appearance model. Moreover, the on-line boosting and semi-boosting models have problems with prolonged partial occlusion (table I).

Compared to a state of the art on-line boosting method like Babenko's *et al.* [8] proposed method (OMB), our method produces better results (see table II). In summary, our method managed to decrease by 47% the average error (in pixels) compared to OMB's best average setting.

Finally, our results were satisfactory. The ONBNN model obtained a good performance in videos with high appearance changes (Figure 4), quick object movement (Figure 4) and partial occlusion. However, we obtained poor results in cases of complete occlusion and erratic movements (Figure 5), mainly caused by the simple motion model used.

For more resulting videos, go to http://www.youtube.com/user/pcortez.

V. CONCLUSION

The results obtained from the on-line NBNN algorithm are very encouraging, as shown in table I and II, our method can outperform the four algorithms in almost all cases. The superior performance is due to the use of covariance features, which give much more information than other features (like Haar or Histogram), and the optimal way the appearance model is updated by using a learning threshold.

Furthermore, the SemiBoost algorithm discards a lot of useful information by leaving all extracted images unlabeled, except for the first frame. This leads to poor performance in the presence of significant appearance changes. The OMB algorithm is particularly good at dealing with fast partial occlusions but not with prolonged partial occlusions.

The proposed method is very simple and only needs a configuration parameter in addition to the fact that its value is generally very stable despite the different scenes ($\lambda = 0.05$).

There are various ways to continue investigation in this area. First, the motion model we used here is fairly simple and could be replaced with something more sophisticated, such as a particle filter. It is also possible to increase pedestrian tracking in complex background situations by optimizing initial window selection.

Finally, it would be interesting to combine our method with the OMB algorithm. Rather than labeling all samples within an area as positive with the OMB algorithm, our method could provide a probability proportional to the distance between the ONBNN model and the sample.

ACKNOWLEDGMENT

We thank Fernando Betteley Cencosud for providing the Santa Isabel's Super Market facilities for video acquisition and the Instituto Nacional de Astrofísica, óptica y Electrónica de México. This research was funded in part by LACCIR (Latin American and Caribbean Collaborative ICT Research), project #S1009LAC006.

REFERENCES

[1] M. Isard and J. MacCormick, "BraMBLe: A Bayesian multiple-blob tracker," in *Eighth IEEE International Conference on Computer Vision, 2001. ICCV 2001. Proceedings*, vol. 2, 2001, pp. 34–41.

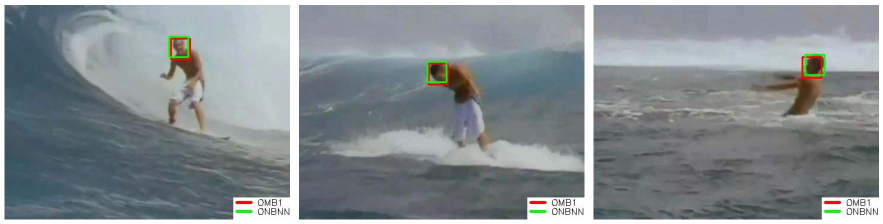

Figure 4. Tracking video with fast movements and changes in appearance. The green rectangle is the proposed On-line NBNN (ONBNN) and red is the tracking from Babenko *et al.* (OMB) [8].

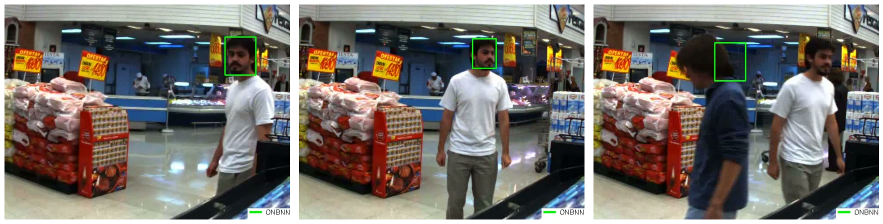

Figure 5. Tracking video with complete occlusion. The green rectangle is the proposed On-line NBNN (ONBNN).

[2] S. Birchfield, “Elliptical head tracking using intensity gradients and color histograms,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. pages. Citeseer, 1998, pp. 232–237.

[3] S. Stalder, H. Grabner, and K. U. Leuven, “Beyond Semi-Supervised Tracking : Tracking Should Be as Simple as Detection , but not Simpler than Recognition,” in *ICCV*, 2009.

[4] H. Palaio and J. Batista, “A region covariance embedded in a particle filter for multi-objects tracking,” 2008.

[5] A. Adam, E. Rivlin, and I. Shimshoni, “Robust fragments-based tracking using the integral histogram,” in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, 2006, pp. 798–805.

[6] Y. Wu, J. Cheng, J. Wang, and H. Lu, “Real-time Visual Tracking via Incremental Covariance Tensor Learning,” *nlpr.labs.gov.cn*, no. Iccv, pp. 1631–1638, 2009.

[7] F. Porikli, O. Tuzel, and P. Meer, “Covariance Tracking using Model Update Based on Lie Algebra,” *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 1 (CVPR’06)*, pp. 728–735, 2006.

[8] B. Babenko, M. Yang, and S. Belongie, “Visual tracking with online multiple instance learning,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, 2009. CVPR Workshops 2009*, 2009, pp. 983–990.

[9] H. Grabner, M. Grabner, and H. Bischof, “Real-time tracking via on-line boosting,” in *Proc. BMVC*, vol. 1. Citeseer, 2006, pp. 47–56.

[10] S. Avidan, “Ensemble tracking.” *IEEE transactions on pattern analysis and machine intelligence*, vol. 29, no. 2, pp. 261–71, Feb. 2007.

[11] O. Tuzel, F. Porikli, and P. Meer, “Region Covariance : A Fast Descriptor for Detection and Classificatio,” *Lecture Notes in Computer Science*, vol. 3952, p. 589, 2006.

[12] P. Viola and M. Jones, “Rapid Object Detection using a Boosted Cascade of Simple,” in *Proc. IEEE CVPR 2001*, vol. 00, no. C, 2001.

[13] O. Tuzel, F. Porikli, and P. Meer, “Pedestrian detection via classification on Riemannian manifolds.” *IEEE transactions on pattern analysis and machine intelligence*, vol. 30, no. 10, pp. 1713–27, octubre 2008.

[14] J. Yao and J. Odobez, “Fast human detection from videos using covariance features,” *Learning*, 2008.

[15] H. Hu and J. Qin, “Region covariance based probabilistic tracking,” *2008 7th World Congress on Intelligent Control and Automation*, pp. 575–580, junio 2008.

[16] S. Montabone and A. Soto, “Human detection using a mobile platform and novel features derived from a visual saliency mechanism,” *Image and Vision Computing*, vol. 28, no. 3, pp. 391–402, marzo 2010.

[17] W. Rossmann, *Lie Groups: An Introduction Through Linear Groups*. Oxford Press, 2002.

[18] A. Ayache, P. Fillard, X. Pennec, and A. Nicholas, “Geometric means in a novel vector space structure on symmetric positive-definite matrices,” *Society*, vol. 29, no. 1, pp. 328–347, 2007.

[19] O. Boiman, E. Shechtman, and M. Irani, “In defense of Nearest-Neighbor based image classification,” *2008 IEEE Conference on Computer Vision and Pattern Recognition*, no. i, pp. 1–8, junio 2008.

[20] H. Grabner, C. Leistner, and H. Bischof, “Semi-supervised on-line boosting for robust tracking,” in *Proc. ECCV*, vol. 2, no. 6. Springer, 2008, pp. 234–247.