An Optimization Framework for Real-Time Appearance-Based Tracking under Weak Perspective

Alex Po Leung, Shaogang Gong
Department of Computer Science
Queen Mary, University of London
London, E1 4NS, UK
{alex,sgg}@dcs.qmul.ac.uk

Abstract

In this work, we present a framework for tracking objects in changing views by finding the subwindow most likely to be the object using Haar-like features selected by AdaBoost as the representation. Probabilistic AdaBoost [14] is used to derive the objective function. In addition, the projective warping of 2D features is used to track 3D objects in non-frontal views in real time. Transformed 2D features can approximate relatively flat object structures such as the two eyes in a face. In this paper, it is shown that, under weak perspective projection, the projective warping of a rectangle feature can be approximated by a similarity transform with an additional free parameter. Since features in non-frontal views are computed on-the-fly by projective transforms under weak perspective projection, our framework requires only frontal-view training samples to track objects in multiple views.

1 Introduction

Much effort has been made to solve the problem of real-time object tracking over the years. However, tracking algorithms still suffer from fundamental problems including drifts away from targets [2] (partially due to change of viewpoint), inability to adapt to changes of object appearance, dependence on the first frame for template matching [3], instability to track objects under deformations (e.g. deformed contours), the inefficiency of Monte Carlo simulations for temporal tracking [4], and reliance on gradients by active contours [5], i.e. problems with similar intensities on the background and the object, or high gradient edges on the object itself. These problems are due to the complexity of the object dynamics. We also have to deal with difficult tracking conditions which include illumination changes, occlusions, changes of viewpoint, moving cameras and non-translational object motions like zooming and rotation.

Recent techniques use more complex and descriptive representations for tracking [6], [7], [8], [9]. A more descriptive representation may reduce the dependency on temporal information for tracking. There are a number of advantages to use a more descriptive

BMVC 2005 doi:10.5244/C.19.26

representation. It makes tracking more robust in cluttered scenes. Less constrained physical state trajectories such as those containing discontinuities may also be tracked. If the representation can encode the appearance of the object more discriminatively, it allows the tracking of objects largely relying on framewise detections without much temporal analysis, such as Viola-Jones detector-based tracking [6]. However, it is both difficult and expensive to obtain statistics to build a 3D model for object detection or tracking while 2D appearance models such as [13], [1], [7] and [9] have been very successful. When multi-views are considered, a huge amount of data is needed for each view for the training for a particular object. Such a huge dataset is impractical to create and it is also computationally expensive to train such a multi-view model.

In this paper, a technique to track non-rigid objects in changing views with only frontal-view training samples is developed. Non-frontal views are deduced from frontal-view samples by geometric transformations under weak perspective. We show that, under weak perspective projection, the projective warping of a rectangle feature can be approximated by a similarity transform with an additional free parameter. The additional degree of freedom allows the change of the aspect ratio of the rectangle feature. The approximation gives us an extremely efficient means to track a rotating object. Only the aspect ratio η, the scale s, a 2D rotation with the angle γ and the centroid location (x_l, y_l) need to be tracked. For face tracking, in difficult situations, a subwindow containing a face may be classified as a non-face because of the viewpoint, occlusions, the background or lighting. With probabilistic AdaBoost [14], the probability of that subwindow being a face is higher than the probability of a non-face subwindow. A better approach would be to find the subwindow most likely to be the object. An optimization framework is, therefore, constructed to find the subwindow. Kalman filters are adopted to track the state variables after projective warping in every frame. They are used to temporally confine the parameter space of the transform. Since we utilize a single appearance model for both detection and tracking ensuring a smooth transition from detection to tracking, the initialization of tracking is completely automatic. No assumption on color is made in our model. Our tracker is able to track non-rigid objects with a roughly flat surface such as faces or cars.

2 Projective Warping of Rectangle Features

Viola and Jones [1] make use of an intermediate representation for images called the integral image or summed-area table [10] to obtain the sum of pixel values for rectangle features with no more than four array references. The integral image is vital to computational efficiency for computing rectangle features. However, features are no longer rectangular after projective transforms. Therefore, we cannot calculate the features directly from the integral image. We propose to use a generalisation of the method to calculate the features while we can still use the integral image. The generalisation was proposed originally by Glassner [11] for texture mapping. It computes the average pixel value within a quadrilateral to an arbitrary degree of accuracy using the integral image with additional computation depending on the accuracy required. Glassner approximates a non-rectangular shape by rectangles. Two methods can be used to do this: additive and subtractive synthesis. Arbitrarily accurate features can be obtained and the integral image can still be used to retain the efficiency of the original appearance model.

An alternative way is to approximate projective transforms. This method makes the

computation much more efficient. A planar projective transformation is a transformation with eight free parameters. A search in the parameter space could be computationally very expensive. An advantage to approximate projective transforms is to reduce the dimensionality of the parameter space. High dimensionality leads to expensive computation and sparsity of data which prevents the search from finding the correct set of parameters. A common approach is to approximate projective transforms by considering weak perspective projection such as planar affine transforms. For a planar affine transform, the number of free parameters is reduced from eight to six.

2.1 Approximating Projective Transforms

We may use weak perspective projection to approximate the perspective projection of rectangle features such as Haar-like features. Let us consider a rectangle feature with corners P_i' where $i = 1$ for the top left, 2 for the top right, 3 for the bottom right and 4 for the bottom left.

$$P_i = R_o P_i', \tag{1}$$

where $R_o = R_{o_3}(\gamma)R_{o_1}(\alpha)R_{o_2}(\beta)$ is the rotation of the object and P_i are the corners after rotating the feature. α is the pitch angle and β is the yaw angle. Both of the rotations are out-of-plane rotations. $R_{o_3}(\gamma)$ is the in-plane rotation around the z-axis. The in-plane rotation is the last rotation so the order is $R_{o_3}(\gamma)R_{o_1}(\alpha)R_{o_2}(\beta)$. The corner of a rectangle feature after rotation in world coordinates is, therefore,

$$X_w = (\cos\gamma\cos\beta - \sin\gamma\sin\alpha\sin\beta)X_w' - \sin\gamma\cos\alpha Y_w', \tag{2}$$

$$Y_w = (\sin\gamma\cos\beta + \cos\gamma\sin\alpha\sin\beta)X_w' + \cos\gamma\cos\alpha Y_w', \tag{3}$$

where (X_w', Y_w') is the corner before rotation in world coordinates. Note that we rotate the object symmetrically by locating it on the x-y plane and its center to be in the origin in world coordinates so $Z_w' = 0$ and, under weak perspective,

$$\bar{Z_w} \approx 0. \tag{4}$$

A rectangle feature can be on any part of the object. Thus, $\bar{Z_w}$ is not exactly zero. Assuming there is no rotation between the world coordinates and the camera coordinates, under weak perspective projection, we have

$$x_{wp} \approx \frac{fR_1^T(T-P_w)}{R_3^T(\bar{P_w}-T)} = -\frac{f(X_w - T_X)}{\bar{Z_w} - T_Z}, \quad y_{wp} \approx \frac{fR_2^T(T-P_w)}{R_3^T(\bar{P_w}-T)} = -\frac{f(Y_w - T_Y)}{\bar{Z_w} - T_Z}.$$

in image coordinates. Using Equation 4, a corner in image coordinates under weak perspective projection is then

$$P_{i_{wp}} = \begin{bmatrix} \frac{f(X_w - T_X)}{T_Z} & \frac{f(Y_w - T_Y)}{T_Z} & f \end{bmatrix}^T. \tag{5}$$

By combining Equations 2, 3 and 5, a corner after the rotation of the object becomes

$$P_{i_{wp}} = \begin{bmatrix} \frac{f[(\cos\gamma\cos\beta - \sin\gamma\sin\alpha\sin\beta)X_w' - \sin\gamma\cos\alpha Y_w' - T_X]}{T_Z} \\ \frac{f[(\sin\gamma\cos\beta + \cos\gamma\sin\alpha\sin\beta)X_w' + \cos\gamma\cos\alpha Y_w' - T_Y]}{T_Z} \\ f \end{bmatrix}$$

under weak perspective projection in image coordinates. Let us assume that large pitch and yaw rotations do not occur at the same time. This is mostly true in natural human head rotations. Therefore, when α becomes large, $\beta \approx 0$, or when β becomes large, $\alpha \approx 0$. Hence, $\sin\alpha\sin\beta \approx 0$, $\sin\alpha\cos\beta \approx \sin\alpha$ and $\cos\alpha\sin\beta \approx \sin\beta$. Accordingly, the rotational matrix R_o is simplified to be

$$\begin{bmatrix} \cos\gamma\cos\beta & -\sin\gamma\cos\alpha & \cos\gamma\sin\beta+\sin\gamma\sin\alpha \\ \sin\gamma\cos\beta & \cos\gamma\cos\alpha & \sin\gamma\sin\beta-\cos\gamma\sin\alpha \\ -\sin\beta & \sin\alpha & \cos\alpha\cos\beta \end{bmatrix}.$$

And, $P_{i_{wp}}$ is

$$\begin{bmatrix} \dfrac{f(\cos\gamma\cos\beta X'_w - \sin\gamma\cos\alpha Y'_w - T_X)}{T_Z} & \dfrac{f(\sin\gamma\cos\beta X'_w + \cos\gamma\cos\alpha Y'_w - T_Y)}{T_Z} & f \end{bmatrix}^T.$$

Let $k = \frac{\cos\beta}{\cos\alpha}$, $\theta = -\gamma$, $t_{13} = -\frac{T_X}{\cos\alpha}$, $t_{23} = -\frac{T_Y}{\cos\alpha}$ and $t_{33} = \frac{T_Z}{f\cos\alpha}$.

$$\begin{bmatrix} \mathfrak{x}_1 \\ \mathfrak{x}_2 \\ \mathfrak{x}_3 \end{bmatrix} = \begin{bmatrix} k\cos\theta & \sin\theta & t_{13} \\ -k\sin\theta & \cos\theta & t_{23} \\ 0 & 0 & t_{33} \end{bmatrix} \begin{bmatrix} X'_w \\ Y'_w \\ 1 \end{bmatrix},$$

where $x_{wp} = \frac{\mathfrak{x}_1}{\mathfrak{x}_3}$ and $y_{wp} = \frac{\mathfrak{x}_2}{\mathfrak{x}_3}$. The transformation between (X'_w, Y'_w) and (x_{wp}, y_{wp}) is a similarity transform when $k = 1$. A similarity transform with 4 degrees of freedom preserves the angles and the ratios of lengths. Thus, for $k = 1$, the aspect ratio of the rectangle feature remains unchanged after the transform. k adds one more degree of freedom to the similarity transform. The number of degrees of freedom becomes 5 and it allows the change of the aspect ratio of the rectangle feature. A rectangle feature after 3D rotation is still approximately rectangular whereas the aspect ratio of the rectangle feature η after rotation becomes $\frac{cos\beta}{cos\alpha}\eta_0$, where $\eta_0 = (X'_{2_w} - X'_{1_w})/(Y'_{1_w} - Y'_{4_w})$ is the aspect ratio before rotation.

This shows that, under weak perspective projection, the projective warping of a rectangle feature can be approximated by a similarity transform with an additional free parameter. The additional degree of freedom allows the change of the aspect ratio of the rectangle feature. The approximation gives us an extremely efficient means to track a rotating object. Only the aspect ratio η, the scale s, a 2D rotation with the angle γ and the centroid location (x_l, y_l) need to be tracked.

3 Optimization for A Cascade of AdaBoost

A cascade of AdaBoost classifiers proposed by Viola and Jones [1] is computationally very efficient for object recognition. The cascade of AdaBoost classifiers reject most subwindows not containing the object in an early stage to reduce computation drastically. In this section, an optimization framework for the cascade is presented.

Using the projective warping of rectangle features, we can track the object in changing views as shown in Figure 1. However, the tracker is not very robust because it selects subwindows containing the object and uses the centroid of the subwindows as the location of the object. For face tracking, in difficult situations, a subwindow containing a face may

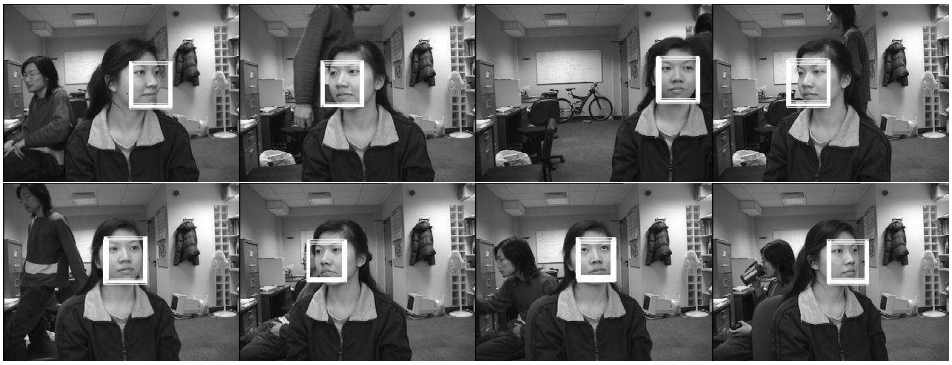

Figure 1: Experiment 1 - Tracking a non-frontal female face in real-time. The figure shows example images from an indoor sequence (Video Sequence 1).

(a) (b) (c) (d)

Figure 2: Notice that all tracking failures in our experiments are due to the fact that no subwindow is classified to be a face in several consecutive frames. Video Sequence 1 is used for Experiments (a) and (b). Video Sequence 2 is used for Experiments (c) and (d). (a) and (c) show the failure of the tracker after tracking respectively 431 frames due to the background and 499 frames due to a partial occlusion. (b) and (d) show the failure of the tracker due to view changes after respectively tracking 17 frames and 141 frames. In Experiments (b) and (d), no geometric transformation is used. The tracker is only able to track very few frames in the sequences without geometric transformations.

be classified as a non-face because of the viewpoint, occlusions, the background or lighting (see examples in Figure 2). With probabilistic AdaBoost [14], the probability of that subwindow being a face is higher than the probability of a non-face subwindow. A better approach would be to find the subwindow most likely to be the object. An optimization framework is, therefore, constructed to find the subwindow. Before optimization can be adopted to allow efficient searches over the parameter space confined by Kalman filters, a score for the cascade of AdaBoost is needed to be used as the objective function.

Let us define $\mathbb{S}$ to be the score for the cascade of probabilistic AdaBoost [14].

$$P[y=+1|x] = \frac{e^{\sum_t \alpha_t h_t}}{e^{\sum_t \alpha_t h_t} + e^{-\sum_t \alpha_t h_t}}, \quad P[y=-1|x] = 1 - P[y=+1|x] = \frac{e^{-\sum_t \alpha_t h_t}}{e^{\sum_t \alpha_t h_t} + e^{-\sum_t \alpha_t h_t}},$$

where $P[y=+1|x]$ is the probability of label +1 which, in our case, is that the subwindow is the object and $P[y=-1|x]$ is the probability that the subwindow is not the object. α_t is the weight assigned to weak hypothesis h_t. The cascade of AdaBoost proceeds to the next stage until the subwindow is rejected or the last stage is reached. Let the number of

stages classifying the subwindow be k and the number of all stages in the cascade be K. First, for two subwindows with the same number of stages evaluated k, we compare the joint probabilities of the outcomes of the evaluated stages by assuming that the outcomes are statistically independent. Let $s_y(k,x_i)$ be the logarithm of the joint probability. That is,

$$s_y(k,x_i) = \log \prod_{l=1}^{k} P_l[y|x_i]$$

where $1 \leq k \leq K$. Second, it is assumed that the probability of a subwindow to be the object is higher if the subwindow progresses further through the cascade [6]. Therefore, we define

$$\mathbb{S}_y(i,x_i) > \mathbb{S}_y(i,x_j) \qquad \forall\, s_y(i,x_i) > s_y(i,x_j) \quad \text{, and} \tag{6}$$

$$\mathbb{S}_y(i,x_i)y > \mathbb{S}_y(j,x_j)y \qquad \forall\, i > j. \tag{7}$$

To find the subwindow, which is the most likely to be the object, we can either compute $\arg\max_{x_i} \mathbb{S}_{+1}(k,x_i)$, or $\arg\min_{x_i} \mathbb{S}_{-1}(k,x_i)$. We choose to the find

$$\arg\min_{x_i} \mathbb{S}_{-1}(k,x_i).$$

Now, the rules to compare the scores of two subwindows have been defined. However, we cannot use standard optimization methods to find $\arg\min_{x_i} \mathbb{S}_{-1}(k,x_i)$ because the score defined is not a quantity. To quantify the score, we let

$$\mathbb{S}_{-1}(k,x_i) = \log \prod_{l=1}^{K} P_l[y=-1|x_i],$$

and set

$$P_l[y=-1|x_i] = 1 \qquad \forall\, l > k$$

in order to satisfy Equation 6 and Equation 7. Thus,

$$\mathbb{S}_{-1}(k,x_i) = \log \prod_{l=1}^{K} P_l[y=-1|x_i] = \log \prod_{l=1}^{k} P_l[y=-1|x_i] = s_{-1}(k,x_i) \tag{8}$$

Note that we approximate $P_l[y=-1|x_i]$ by 1 for all $i > k$. In Equation 8, for two subwindows with the same number of stages evaluated k, $\prod_{l=k+1}^{K} P_l[y=-1|x_i] = 1$. Hence, only $s_{-1}(k,x_i)$ and $s_{-1}(k,x_j)$ are compared. For two subwindows with respectively k and $k+n$ as the number of stages evaluated, when $\prod_{l=1}^{k} P_l[y=-1|x_i]$ and $\prod_{l=1}^{k} P_l[y=-1|x_j]$ are comparable, $\mathbb{S}_{-1}(k,x_i)$ is larger than $\mathbb{S}_{-1}(k+n,x_j)$ since $\prod_{l=k+1}^{k+n} P_l[y=-1|x_i] = 1$ for $\mathbb{S}_{-1}(k,x_i)$. $\prod_{l=1}^{k} P_l[y=-1|x_i]$ should not be less than $\prod_{l=1}^{k} P_l[y=-1|x_j]$ because, for Subwindow x_i, the number of stages evaluated is lower. That means Subwindow x_i is less likely to be the object. Therefore, $\prod_{l=1}^{k} P_l[y=-1|x_i]$ and $\prod_{l=1}^{k} P_l[y=-1|x_j]$ should at least be comparable.

Figure 3 shows some typical shapes of the objective function $\mathbb{S}_{-1}(k,x_i)$. Because the objective function is not necessarily unimodal, the optimization algorithm is required to find the global peak in certain cases.

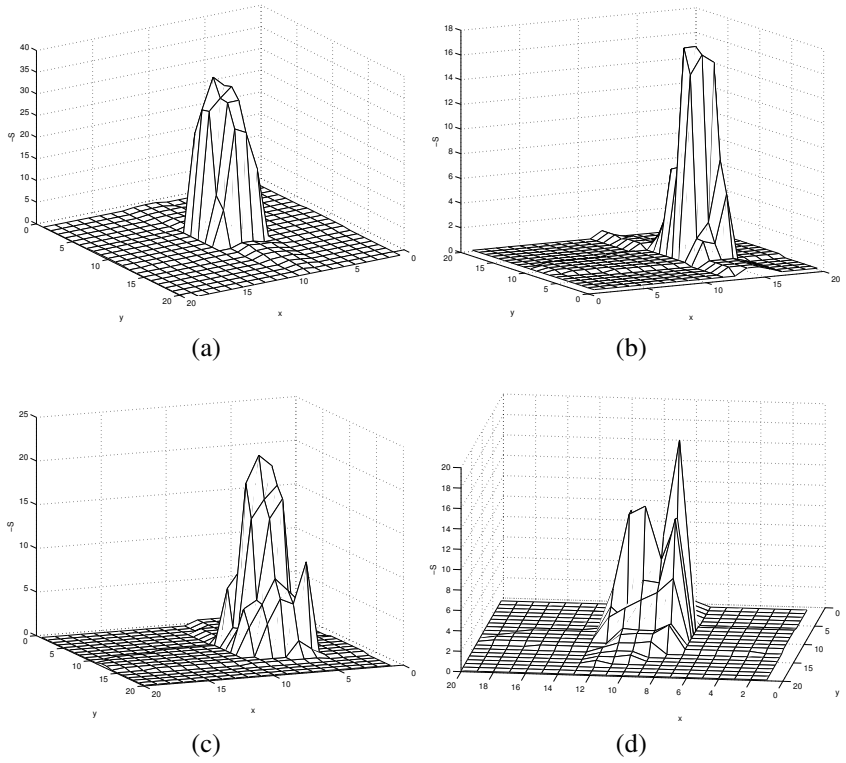

Figure 3: Some Typical Shapes of The Objective Function $\mathbb{S}_{-1}(k,x_i)$:Instead of plotting $\mathbb{S}_{-1}(k,x_i)$, -$\mathbb{S}_{-1}(k,x_i)$ is plotted against the location of the object (x_l,y_l) in the previous frame for easy visualization. In Figures (a) and (b) the objective function is unimodal. In Figures (c) and (d), the objective function is multimodal and there are two peaks in each of the figures. One of the two peaks is only a local peak. Thus, the optimization algorithm is required to find the global peak in certain cases.

4 Experimental Results

We use the MIT-CBCL face dataset [12] which consists of 6,977 cropped images (2,429 faces and 4,548 nonfaces). The resolution of the images is 19×19 and slightly lower than 24×24 used by Viola and Jones [1].

To make use of the integral image, we set the in-plane rotation angle γ to 0 so that all features are upright rectangles. That is to say we only track the out-of-plane rotations α and β. A 12-stage cascade of AdaBoost [1] is used in our experiments. There are 1,127 features in the 12 stages. The 12 stages separately compute 2, 5, 20, 50 $\times$ 2, 100 $\times$ 5 and 200 $\times$ 2 features. After the detector initializes our tracker, four Kalman filters are separately used to track the aspect ratio η, the scale s and the centroid location of the object (x_l, y_l). For the optimization, since the search space confined by the Kalman filters should be reasonably small, when compared with an exhaustive search, stochastic optimization to find the global minimum would not be very efficient. An exhaustive search

Figure 4: Experiment 2 - Tracking a non-frontal female face in real-time with the proposed objective function. The figure shows example images from an indoor sequence (Video Sequence 1).

is used to find the global minimum to avoid the local minimums of the objective function.

Experiment 2 (see Figure 4) shows a video (Video Sequence 1, the same sequence as in Figures 2a and 2b) with $|\beta| < 90^\circ$. It shows that faces with relatively large $|\beta|$ could also be tracked. It is clear that the side views share some common features with the frontal view after projective transforms. The experiment demonstrates that the tracker manages to track the face in Video Sequence 1 with 500 frames while it fails without our optimization framework as shown in Figures 2a and 2b. Experiment 3 (Figure 5) shows tracking a partially occluded non-frontal male face outdoors with a moving hand-held camera (Video Sequence 2, the same sequence as in Figures 2c and 2d). This experiment demonstrates that the tracker manages to track the face in Video Sequence 2 with 526 frames while it fails without our optimization framework as shown in Figures 2c and 2d. With Experiment 4, Figure 6 shows a 30-second long video (Video Sequence 3) with 894 frames. The distance between the person being tracked and the camera varies considerably. This experiment shows that the tracker is very robust when both the camera and the object are moving. The tracker is able to track objects in a very low resolution. The tracked face is very small in the 320×240 sequence and the scene in the background is highly complex. Experiments 2, 3 and 4 demonstrate that our tracker can track deformable objects from different viewpoints, i.e. faces with different expressions in different views in this case.

In our current experiments, the tracking frame rate is 7.5 frames per second with the frame size 320×240. The code for the interface is in Matlab. Our core code is compiled by gcc on Cygwin on an AMD Athlon 1.68GHz machine.

Figure 5: Experiment 3 - Tracking a partially occluded non-frontal male face in real-time with the proposed objective function. The figure shows example images from an outdoor sequence with a moving hand-held camera (Video Sequence 2).

5 Conclusion

We have demonstrated a system using the projective warping of 2D features to track 3D objects in non-frontal views in real time. Our framework requires only frontal-view training samples. Features in other views are computed by projective transforms under weak perspective projection on-the-fly. Furthermore, an optimization framework is constructed to find the subwindow most likely to be the object. Our method can track objects with a roughly flat surface such as faces or cars.

Future work includes pose estimation making use of the out-of-plane rotation angles α and β, and making the tracker more efficient by using noisy optimization such as implicit filtering for searches in the parameter space for projective transforms.

References

[1] P. Viola and M. Jones, "Robust real-time object detection," *IJCV*, 2002.

[2] I. Matthews, T. Ishikawa and S. Baker, "The Template Update Problem," *PAMI(26)*, No. 6, 2004, pp. 810-815.

[3] F. Jurie and M. Dhome, "Hyperplane Approximation for Template Matching," *IEEE PAMI 24(7)*, 996-1000, 2002.

[4] S. Arulampalam, S. Maskell, N. Gordon and T. Clapp, "A tutorial on particle filters for on-line non-linear/non-gaussian bayesian tracking," *Transaction of Signal Processing*, 50(2):174-188, 2002.

[5] M. Isard and A. Blake, "CONDENSATION – conditional density propagation for visual tracking," *IJCV*, 29, 1, 5–28, (1998).

[6] G. Shakhnarovich, P. A. Viola and B. Moghaddam, "A Unified Learning Framework for Real-Time Face Detection and Classification," *IEEE FG*, pp. 14-21, May 2002.

[7] S. Avidan, "Support Vector Tracking," *PAMI(26)*, No. 8, August 2004, pp. 1064-1072.

Figure 6: Experiment 4 - Tracking a non-frontal male face at varying distance in real-time with the proposed objective function. The figure shows example images from an outdoor sequence with a moving hand-held camera (Video Sequence 3).

[8] O. Williams, A. Blake and R. Cipolla, "A Sparse Probabilistic Learning Algorithm for Real-Time Tracking," *ICCV*, Nice, France, 2003.

[9] M. J. Black and A. Jepson, "EigenTracking: Robust matching and tracking of articulated objects using a view-based representation," *ICCV*, 26(1), pp. 63-84, 1998.

[10] F. Crow, "Summed-Area Tables for Texture Mapping," *SIGGRAPH*, Pages 207-212, July, 1984.

[11] A. Glassner, "Adaptive Precision in Texture Mapping," *SIGGRAPH*, 20, 4, August 1986, pp. 297-306.

[12] CBCL Face Database #1, MIT Center For Biological and Computation Learning, http://www.ai.mit.edu/projects/cbcl.

[13] H. Schneiderman and T. Kanade, "Object Detection Using the Statistics of Parts", *IJCV*, v.56 n.3, p.151-177, February-March 2004.

[14] Friedman, J., T. Hastie, and R. Tibshirani, "Additive logistic regression: A statistical view of boosting," *The Annals of Statistics*, 28(2), 337-374, 2000.