Robust Fragments-based Tracking using the Integral Histogram

Amit Adam and Ehud Rivlin
Dept. of Computer Science
Technion - Israel Institute of Technology
Haifa 32000, Israel
{amita,ehudr}@cs.technion.ac.il

Ilan Shimshoni
Dept. of Management Information Systems
Haifa University
Haifa 31905, Israel
{ishimshoni}@mis.haifa.ac.il

Abstract

We present a novel algorithm (which we call "FragTrack") for tracking an object in a video sequence. The template object is represented by multiple image fragments or patches. The patches are arbitrary and are not *based on an object model (in contrast with traditional use of model-based parts e.g. limbs and torso in human tracking). Every patch votes on the possible positions and scales of the object in the current frame, by comparing its histogram with the corresponding image patch histogram. We then minimize a robust statistic in order to combine the vote maps of the multiple patches.*

A key tool enabling the application of our algorithm to tracking is the integral histogram data structure [18]. Its use allows to extract histograms of multiple rectangular regions in the image in a very efficient manner.

Our algorithm overcomes several difficulties which cannot be handled by traditional histogram-based algorithms [8, 6]. First, by robustly combining multiple patch votes, we are able to handle partial occlusions or pose change. Second, the geometric relations between the template patches allow us to take into account the spatial distribution of the pixel intensities - information which is lost in traditional histogram-based algorithms. Third, as noted by [18], tracking large targets has the same computational cost as tracking small targets.

We present extensive experimental results on challenging sequences, which demonstrate the robust tracking achieved by our algorithm (even with the use of only gray-scale (non-color) information).

1. Introduction

Tracking is an important subject in computer vision with a wide range of applications - some of which are surveillance, activity analysis, classification and recognition from motion and human-computer interfaces. The three main categories into which most algorithms fall are feature-based tracking (e.g. [3]), contour-based tracking (e.g. [15]) and region-based tracking (e.g [13]). In the region-based category, modeling of the region's content by a histogram or by other non-parametric descriptions (e.g. kernel-density estimate) have become very popular in recent years. In particular, one of the most influential approaches is the mean-shift approach [8, 6].

With the experience gained by using histograms and the mean shift approach, some difficulties have been studied in recent years. One issue is the local basin of convergence that the mean shift algorithm has. Recently in [22] the authors describe a method for converging to the optimum from far-away starting points.

A second issue, inherent in the use of histograms, is the loss of spatial information. This issue has been addressed by several works. In [26] the authors introduce a new similarity measure between the template and image regions, which replaces the original Bhattacharyya metric. This measure takes into account both the intensities and their position in the window. The measure is further computed efficiently by using the Fast Gauss Transform. In [12], the spatial information is taken into account by using "oriented kernels" - this approach is additionally shown to be useful for wide baseline matching. Recently, [4] has addressed this issue by adding the spatial mean and covariance of the pixel positions who contribute to a given bin in the histogram - naming this approach as "spatiograms".

A third issue which is not specifically addressed by these previous approaches is occlusions. The template model is global in nature and hence cannot handle well partial occlusions.

In this work we address the latter two issues (spatial information and occlusion) by using parts or fragments to represent the template. The first issue is addressed by efficient exhaustive search which will be discussed later on. Given a template to be tracked, we represent it by multiple histograms of multiple rectangular sub regions (patches) of the template. By measuring histogram similarity with patches of the target frame, we obtain a vote-map describing the possible positions of each patch in the target frame. We then combine the vote-maps in a robust manner. Spatial information is not lost due to the use of spatial relationships between patches. Occlusions result in some of the patches contributing outlier vote-maps. Due to our robust method for combining the vote maps, the combined estimate of the target's position is still accurate.

The use of parts or components is a well known technique in the object recognition literature (see chapter 23 in [11]). Examples of works which use the spatial relationships between detections of object parts are [21, 17, 16, 2]. In [24] the issue of choosing informative parts which contain the most information concerning the presence of an object class is discussed. A novel application of detecting unusual events and salient features based on video and image patches has recently been described in [5].

In tracking, the use of parts has usually been in the context of human body tracking where the parts are based on a model of the human body - see [23] for example. Recently, Hager, Dewan and Stewart [14] (followed by Fan et al. [10]) analyzed the use of multiple kernels for tracking. In these works the connection between the intensity structure of the target, the possible transformations it can experience between consecutive frames, and the kernel structure used for kernel tracking was analyzed. This analysis gives insight on the limitations of single-kernel tracking, and on the advantages of multiple-kernel tracking. The parts-based tracking algorithm described in this work differs from these and other previous works in a number of important issues:

- Our algorithm is robust to partial occlusions - the works in [14, 10] cannot handle occlusions due to the non-robust nature of the objective function.

- Our algorithm allows the use of any metric for comparing two histograms, and not just analytically-tractable ones such as the Bhattacharyya or the equivalent Matusita metrics. Specifically, by using non-componentwise metrics the effects of bin-quantization are reduced (see section 2.1 and Fig. 3).

- The spatial constraints are handled automatically in our algorithm by the voting mechanism. In contrast, in [10] these constraints have to be coded in (e.g. the fixed length constraint).

- The robust nature of our algorithm and the efficient use of the integral histogram allows one to use the algorithm without giving too much thought on the choice of multiple patches/kernels. In contrast, in [14, 10] the authors carefully chose a small number of multiple kernels for each specific sequence.

- We present extensive experimental validation, on out-of-the-lab real sequences. We demonstrate good tracking performance on these challenging scenarios, obtained with the use of only gray-scale information.

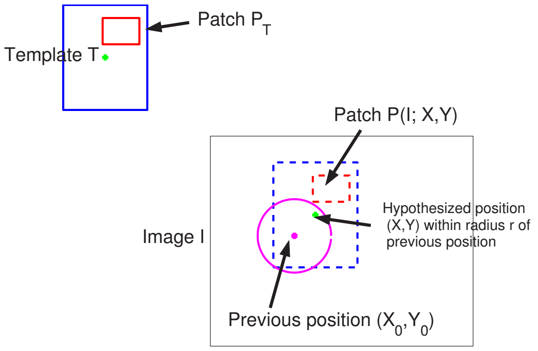

Patch P_T
Template T
Patch P(I; X,Y)
Image I
Hypothesized position (X,Y) within radius r of previous position
Previous position (X_0,Y_0)

Figure 1. Template patch P_T and the corresponding image patch $P_{I;(x,y)}$ for a hypothesized position (x, y)

Our algorithm requires the extraction of intensity or color histograms over a large number of sub-windows in the target image and in the object template. Recently Porkili [18] extended the integral image [25] data structure to an "integral histogram" data structure. Our algorithm exploits this observation - a necessary step in order to be able to apply the algorithm for real time tracking tasks. We extend the tracking application described in [18] by our use of parts, which is crucial in order to achieve robustness to occlusions.

2. Patch Tracking

Given an object O and the current frame I, we wish to locate O in the image. Usually O is represented by a template image T, and we wish to find the position and the scale of a region in I which is closest to the template T in some sense. Since we are dealing with tracking, we assume that we have a previous estimate of the position and scale, and we will search in the neighborhood of this estimate. For clarity, we will consider in the following only the search in position (x, y).

Let (x_0, y_0) be the object position estimate from the previous frame, and let r be our search radius. Let $P_T = (dx, dy, h, w)$ be a rectangular patch in the template, whose center is displaced (dx, dy) from the template center, and whose half width and height are w and h respectively. Let (x, y) be a hypothesis on the object's position in the current frame. Then the patch P_T defines a corresponding rectangular patch in the image $P_{I;(x,y)}$ whose center is at $(x + dx, y + dy)$ and whose half width and height are w and h. Figure 1 describes this correspondence.

Given the patch P_T and the corresponding image patch $P_{I;(x,y)}$, the similarity between the patches is an indication of the validity of the hypothesis that the object is indeed located at (x, y). If $d(Q, P)$ is some measure of similarity between patch Q and patch P, then we define

$$V_{P_T}(x,y) = d(P_{I;(x,y)}, P_T) \tag{1}$$

When (x,y) runs on the range of hypotheses, we get $V_{P_T}(\cdot,\cdot)$ which is the vote map corresponding to the template patch P_T.

2.1. Patch Similarity Measures

We measure similarity between patches by comparing their gray-level or color histograms. This allows more flexibility than the standard normalized correlation or SSD measures. Although for a single patch we lose spatial information by considering only the histogram, our use of multiple patches and their spatial arrangement in the template compensates for this loss.

There are a number of known methods for comparing the similarity of two histograms [9]. The simplest methods compare the histograms by comparing corresponding bins. For example, one may use the chi-square statistic or simply the norm of the difference between the two histograms when considered as two vectors.

The Kolmogorov-Smirnov statistic compares histograms by building the cumulative distribution function (that is cumulative sum) of each histogram, and comparing these two functions. The advantage over bin-wise methods is smoothing of nearby bin differences due to the quantization of measurements into bins.

A more appealing approach is the Earth Mover's Distance (EMD) between two histograms, described in [20]. In this approach the actual dissimilarity between the bins themselves is also taken into account. The idea is to compute how much probability has to move between the various bins in order to transform the first histogram into the second. In doing so, bin dissimilarity is used: for example, in gray scale it costs more to move 0.1 probability from the $[16,32)$ bin to the $[128,144)$ bin, than to move it to the $[32,47)$ bin. In the first case, the movement of probability is required because of a true difference in the distributions, and in the second case it might be due simply to quantization errors. This is exactly the transportation problem of linear programming. In this problem the bases are always triangular and therefore the problem may be solved efficiently. See [20] for more details and advantages of this approach.

We have experimented with two similarity measures. The first is the naive measure which treats the histograms as vectors and just computes the norm of their difference. The second is the EMD measure. For gray scale images, we used 16 bins. The EMD calculation is very fast and poses no problem. For color images, the number of bins is much larger (with only 8 bins per channel we get 512 bins). Therefore when using the EMD we took the $K = 10$ bins which obtained maximal counts, normalized them to unity and then used the EMD. We used the original EMD code developed by Rubner [19].

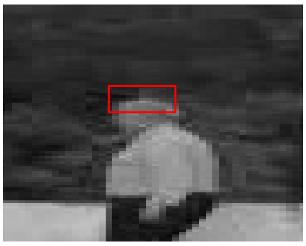
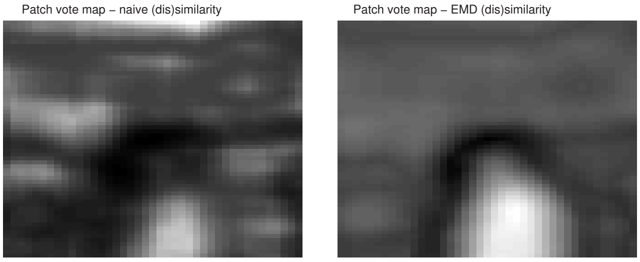


Figure 2. An example patch



(a) (b)

Figure 3. Vote maps for the example patch using the the naive measure and the EMD measure. The lower (darker) the vote - the more likely the position. Left (a) - naive measure. Right (b) - EMD. The EMD surface has a less blurred minimum, and is smoother at the same time.

Figure 2 shows an example patch (we use gray scale in this example). We computed the patch vote map for all the locations around the patch center which are up to 30 pixels above or below and up to 20 pixels to the left or right. Figure 3 shows the resulting vote maps when using the naive measure and the EMD measure. Note that in both measures the lower the value (darker in the image), the more similar the histograms. The EMD surface is smoother and has a more distinct minimum than the surface obtained when using the naive measure.

3. Combining Vote Maps

In the last section we saw how to obtain a vote map $V_{P_T}(\cdot,\cdot)$ for every template patch P_T. The vote map gives a scalar score for every possible position (x,y) of the target in the current frame I, given the information from patch P_T. We now want to combine the vote maps obtained from all template patches.

Basically we could sum the vote maps and look for the position which obtained the minimal sum (recall that our vote maps actually measure dissimilarity between patches). The drawback of this approach is that an occlusion affecting even a single patch may contribute a high value to the sum at the correct position, resulting in a wrong estimate. In other words, we would like to use a robust estimator which could handle outliers resulting from occluded patches or other reasons (e.g. partial pose change - for example a person turns his head).

One way to make the sum robust to outliers is to bound the possible contribution of an outlier

$$C(x,y) = \sum_P \left\{ \begin{array}{cc} V_P(x,y) & V_P(x,y) < T \\ T & V_P(x,y) >= T \end{array} \right. \qquad (2)$$

by some threshold T. If we adopt a probabilistic view of the measurement process - by transforming the vote map to a likelihood map (e.g. by setting $L_P(x,y) = K * \exp^{-\alpha * V_P(x,y)}$) - then this method is equivalent to adding a uniform outlier density to the true (inlier) density. Minimizing the value of $C(\cdot,\cdot)$ is then equivalent to obtaining a maximum likelihood estimate of the position, but without letting an outlier take the likelihood down to 0.

However, we found that choosing the threshold T is not very intuitive, and that the results are sensitive to this choice. A different approach is to use a LMedS-type estimator. At each point (x,y) we order the obtained values $\{V_P(x,y)|\text{ patches P}\}$ and we choose the Q'th smallest score:

$$C(x,y) = Q'\text{th value in the sorted set } \{V_P(x,y)|\text{ patches } P\} \qquad (3)$$

The parameter Q is much more intuitive: it should be the maximal number of patches that we always expect to yield inlier measurements. For example, if we think that we are guaranteed that occlusions will always leave at least a quarter of the target visible, than we will choose Q to be 25% of the number of patches (to be precise - we assume that at least a quarter of the patches will be visible).

The additional computational burden when using estimate (3) instead of (2) is not significant (the number of patches is less than 40).

4. Using the Integral Histogram

The algorithm that we have described requires multiple extractions of histograms from multiple rectangular regions. We extract histograms for each template patch, and then we compare these histograms with those extracted from multiple regions in the target image. The tool enabling this to be done in real time, as required by tracking, is the integral histogram described in [18].

The method is an extension of the integral image data structure described in [25]. The integral image holds at the point (x,y) in the image the sum of all the pixels contained in the rectangular region defined by the top-left corner of the image and the point (x,y). This image allows to compute the sum of the pixels on arbitrary rectangular regions by considering the 4 integral image values at the corners of the region - in other words in (very short) constant time independent of the size of the region.

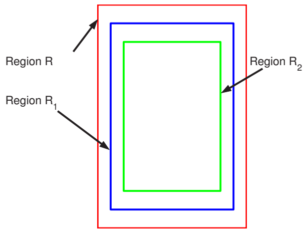


Figure 4. Giving less weight to contributions from the outer part of the region

In order to extract histograms over arbitrary rectangular regions, in the integral histogram we build for each bin of the histogram an integral image counting the cumulative number of pixels falling into that bin. Then by accessing these integral images we can immediately compute the number of pixels in a given region which fall into every bin, and hence we obtain the histogram of that rectangular region.

Once the integral histogram data structure is computed (with cost proportional to the image (or actually search region) size times the number of bins), extraction of a histogram over a region is very cheap. Therefore evaluating a hypothesis on the current object's position (and scale) is relatively cheap - basically it is the cost of comparing two histograms.

As noted previously, a tracking application of the integral histogram was suggested in [18]. We extend that example with the parts-based approach.

4.1. Weighting Pixel Contributions

An important feature in the traditional mean shift algorithm is the use of a kernel function which assigns lower weights to pixels which are further away from the target's center. These pixels are more likely to contain background information or occluding objects, and hence their contribution to the histogram is diminished. However, when using the integral histogram, it is not clear how one may include this feature.

The following discrete approximation scheme may be used instead of the more continuous kernel weighting (see Figure 4). If we want to extract a weighted histogram in the rectangular region R, we may define an inner rectangle R_1 and subtract the integral histogram counts of R_1 from those of R to obtain the counts in the ring $R - R_1$. These counts and the R_1 counts may be weighted differently and combined to give a weighted histogram on R. Of course, an additional inner rectangle R_2 may be used and so forth.

The additional cost is the access and arithmetic involved with 4 additional pixels for every added inner rectangle. For medium and large targets this cost is negligible when compared to trying to weigh the pixels in a straightforward manner.

4.2. Scale

As noted in [25, 18], an advantage of the integral image/histogram is that the computational cost for large regions is not higher than the cost for small regions. This makes our search for the proper scale of the target not harder than our search for the proper location.

Just as a hypothesis on the position (x, y) of the object defines a correspondence between a template patch P_T and an image patch $P_{I;(x,y)}$, if we add a scale s to the hypothesis, it is straightforward to find the corresponding image patch $P_{I;(x,y,s)}$: we just scale the displacement vector of the patch and its height and width by s. The cost of extracting the histogram for this larger (or smaller) image patch is the same as for the same-size patch.

We have implemented the standard approach (suggested in [8] and adopted by e.g. [4]) of enlarging and shrinking the template by 10%, and choosing the position and scale which give the lowest score in (3). The next section will present some results obtained with this approach.

We remark that as noted in [7], this method has some limitations. For example, if the object being tracked is uniform in color, then there is a tendency for the target to shrink. In the case of partial occlusions of the target, we are faced with an additional dilemma: suppose that a uniform colored target is partially occluded. We get a good score by shrinking the target and locating it around the non-occluded part. Due to our robust approach, we also get a reasonable score by keeping the target at the correct size and locating it at the correct position, which includes some occluded parts of the target. However, there is no guarantee that the correct explanation will yield a better score than the partial explanation. A full treatment of this problem is out of the scope of the current work.

5. Results

Note: The full video clips are available at the authors' websites.

We now present our experimental results. The tracker was run on gray scale images and the histograms we used contained 16 bins. Note that the integral histogram data structure requires an image for every bin in the histogram, and therefore on color images the application can become quite memory-consuming.

We used vertical and horizontal patches as shown in Figure 5. The vertical patches are of half the template height, and about one tenth of the template's width. The horizontal patches are defined in a similar manner. Over all we had around 36 patches (the number slightly varies with template size because of rounding to integer sizes). We note that this choice of patches was arbitrary - we just tried it and found it was good enough. In the discussion we return to this issue.

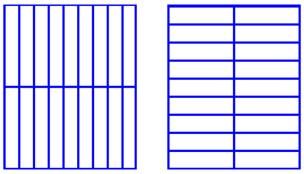


Figure 5. The patches we used

The search radius was set to 7 pixels from the previous target position. The template was fixed at the first frame and not updated during the sequence (more on this in the discussion). We used the 25'th percent quantile for the value of Q in (3).

These settings of the algorithm's parameters were fixed for all the sequences.

The first two sequences ("face" and "woman") show the robustness to occlusions. For these sequences we manually marked the ground truth (every fifth frame), and plotted the position error of our tracker and of the mean-shift tracker. In both cases our tracker was not affected by the occlusions, while the mean-shift tracker did drift away. Figures 6 and 7 show the errors with respect to the ground truth. Figure 8 shows the initial templates and a few frames from these sequences. Note the last frame of the woman sequence (second row) where one can see an example of the use of spatial information (see figure caption also).

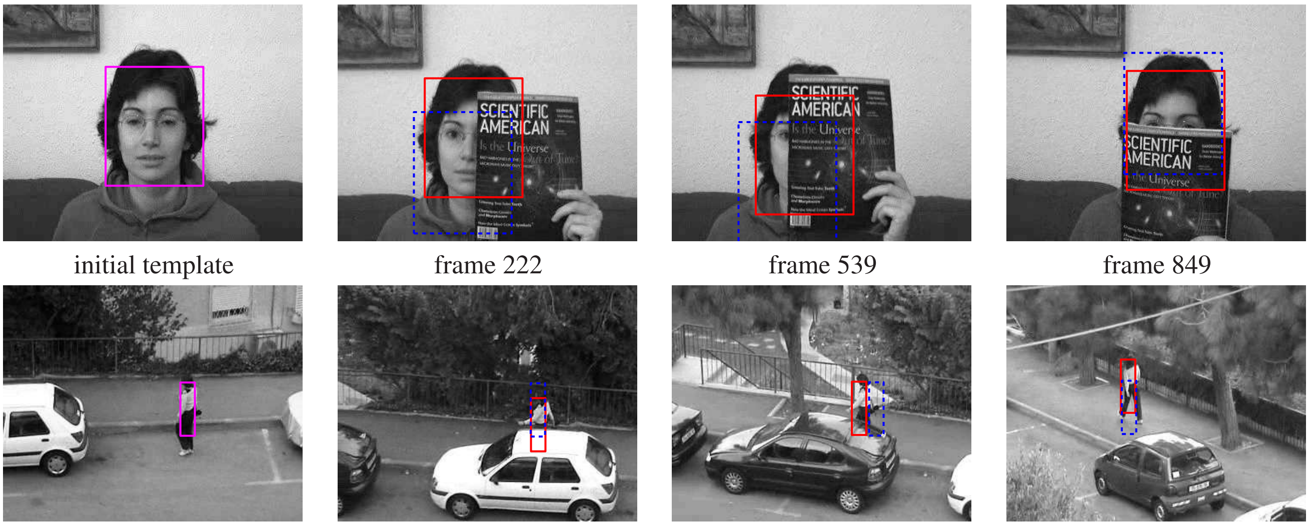
We additionally note that we ran our tracker on these examples with only a single patch containing the whole template, and it failed (this is actually the example tracker described in [18]).

The next sequence - "living room" in Figure 9 - shows performance under partial pose change. When the tracked woman turns her head the mean shift tracker drifts, and then together with an occlusion it gets lost. Our tracker is robust to these interferences.

In Figure 10 we present more samples from three more sequences. In these frames we marked only our tracker. The first two sequences are from the CAVIAR database [1]. The first is an occlusion clip and the second shows target scale changes. The third sequence is again an occlusion clip. We bring it to demonstrate how our tracker uses spatial information (which is generally lost in histogram-based methods). Both persons have globally similar histograms (half dark and half bright). Our tracker "knows" that the bright pixels should be in the upper part of the target and therefore does not drift to the left person when the two persons are close.

6. Discussion and Conclusions

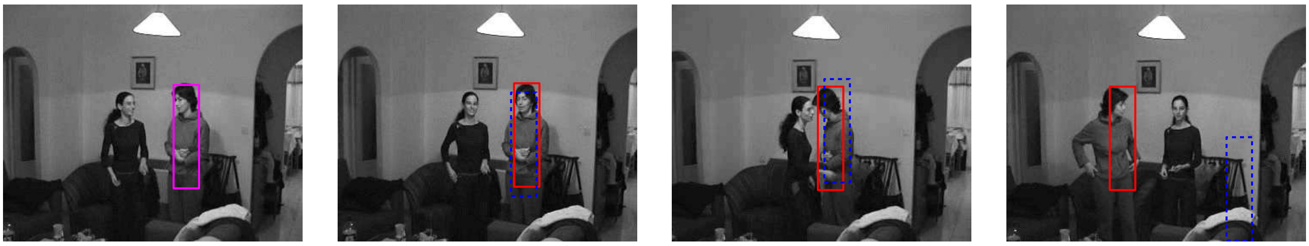
In this work we present a novel approach ("FragTrack") to tracking. Our approach combines fragments-based repre-



initial template | frame 222 | frame 539 | frame 849

initial template | frame 66 | frame 134 | frame 456

Figure 8. Occlusions - frames from “face” and “woman” sequences. Our tracker - solid red. Mean-shift tracker - dashed blue. Note in frame 456 how the spatial information - bright in the upper part, dark in the lower part - helps our tracker. The mean-shift tracker which does not have this information chooses a region with a dark upper part and a bright lower part.



initial template | frame 29 | frame 141 | frame 209

Figure 9. Pose change and occlusions - frames from “living room” sequence. Our tracker - solid red. Mean-shift tracker - dashed blue.

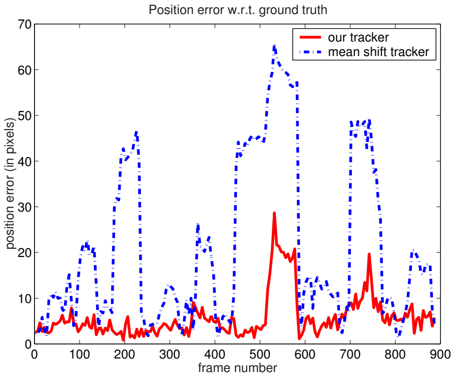


Figure 6. Face sequence - error with respect to manually marked ground truth. Our tracker - solid red. Mean shift - dashed blue. Please see videos for additional impression

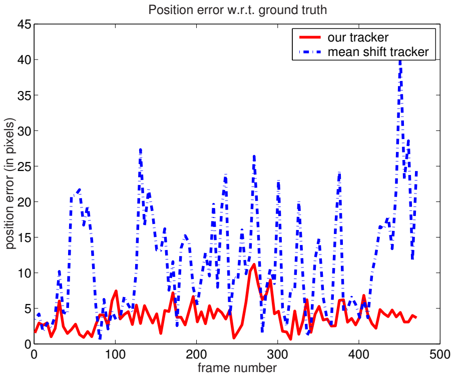
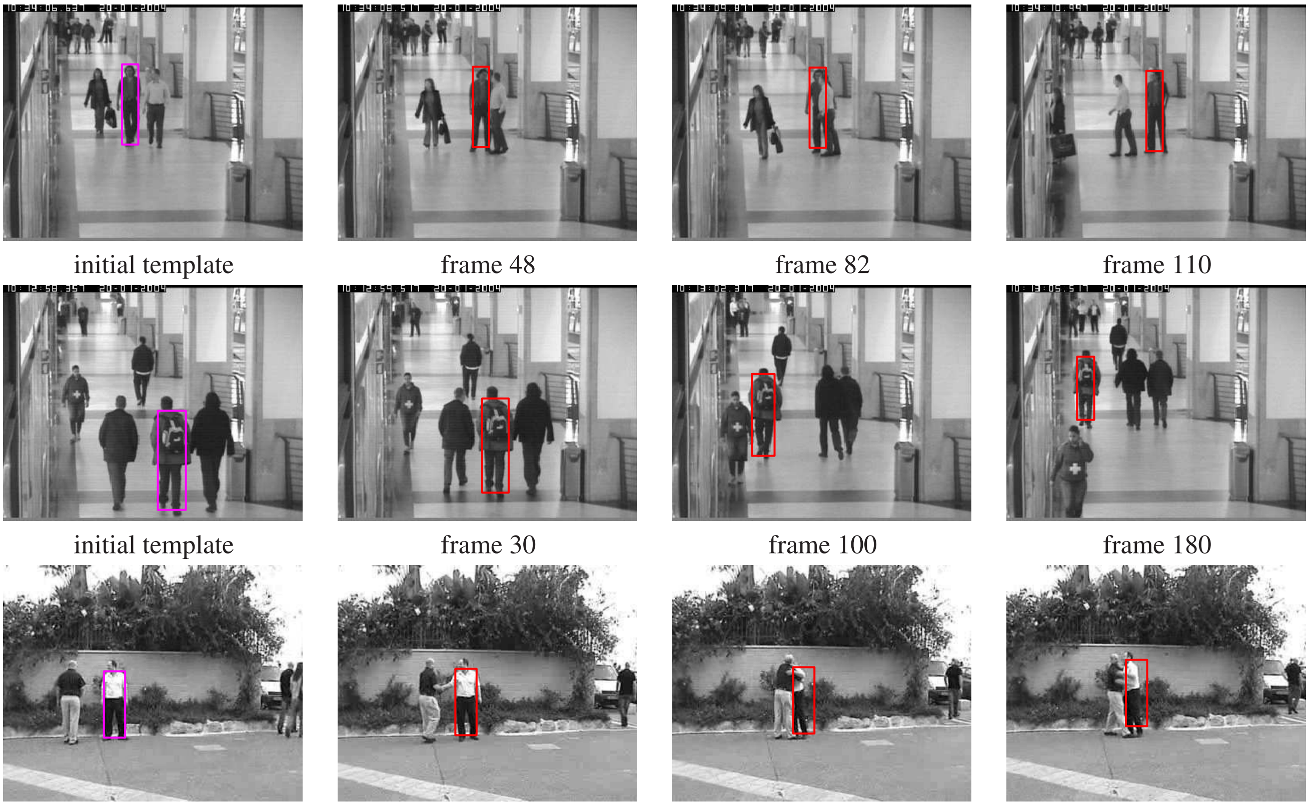


Figure 7. Woman sequence - error with respect to manually marked ground truth. Our tracker - solid red. Mean shift - dashed blue. Please see videos for additional impression

sentation and voting known from the recognition literature, with the integral histogram tool. The result is a real time tracking algorithm which is robust to partial occlusions and pose changes.

In contrast with other tracking works, our parts or fragments approach is model-free: the fragments are chosen



initial template frame 48 frame 82 frame 110

initial template frame 30 frame 100 frame 180

initial template frame 35 frame 65 frame 90

Figure 10. Additional examples. The first two rows are from the CAVIAR database. *No background subtraction/frame differencing was used.* In the last row note again the use of spatial information - both persons have the same global histogram.

arbitrarily and not by reference to a pre-determined parts-based description of the target (say limbs and torso in human tracking, or eyes and nose in face tracking).

Without the integral histogram's efficient data structure it would not have been possible to compute each fragment's votes map. On the other hand, without using a fragments-based algorithm, robustness to partial occlusions or pose changes would not have been possible.

We demonstrate the validity of our approach by accurate tracking of targets under partial occlusions and pose changes in several video clips. The tracking is achieved without any use of color information.

There are several interesting issues for current and future work. The first is the question of template updating. We want to avoid introduction of occluding objects into the template. The use of the various fragments' similarity scores may be useful towards meeting this goal.

A second issue is the partial versus full explanation dilemma described earlier and in [7] when choosing scale. This dilemma is even more significant under partial occlusions.

Lastly, we may also consider disconnected rectangular fragments. It would be interesting to find a way to choose the most informative fragments [24] with respect to the tracking task.

References

[1] Caviar datasets available at http://groups.inf.ed.ac.uk/vision/caviar/caviardata1/.

[2] S. Agarwal, A. Awan, and D. Roth. Learning to detect objects in images via a sparse, part-based representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(11):1475–1490, 2004.

[3] D. Beymer, P. McLauchlan, B. Coifman, and J. Malik. A real-time computer vision system for measuring traffic parameters. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 1997.

[4] S. Birchfield and S. Rangarajan. Spatiograms vs. histograms for region based tracking. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2005.

[5] O. Boiman and M. Irani. Detecting irregularities in images and video. In *Proc. IEEE Int. Conf. on Computer Vision (ICCV)*, 2005.

[6] G. Bradski. Real time face and object tracking as a component of a perceptual user interface. In *Proc. IEEE WACV*, pages 214–219, 1998.

[7] R. Collins. Mean shift blob tracking through scale space. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages II: 234–240, 2003.

[8] D. Comaniciu, R. Visvanathan, and P. Meer. Kernel based object tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(5):564–575, 2003.

[9] W. Conover. *Practical Nonparamteric Statistics*. Wiley, 1998.

[10] Z. Fan, Y. Wu, and M. Yang. Multiple collaborative kernel tracking. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2005.

[11] D. Forsyth and J. Ponce. *Computer Vision: A Modern Approach*. Prentice-Hall, 2001.

[12] B. Georgescu and P. Meer. Point matching under large image deformations and illumination changes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26:674–689, 2004.

[13] G. Hager and P. Belhumeur. Efficient region tracking with paramteric models of geometry and illumination. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(10):1125–1139, 1998.

[14] G. Hager, M. Dewan, and C. Stewart. Multiple kernel tracking with ssd. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2004.

[15] M. Isard and A. Blake. Condensation: Conditional density propagation for visual tracking. *Int. Journal of Computer Vision (IJCV)*, 29(1):5–28, 1998.

[16] K. Mikolajczyk, C. Schmid, and A. Zisserman. Human detection based on a probabilistic assembly of robust part detectors. In *Proc. Eurpoean Conf. on Computer Vision (ECCV)*, 2004.

[17] A. Mohan, C. Papageorgiou, and T. Poggio. Example-based object detection in images by components. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(4):349–361, 2001.

[18] F. Porkili. Integral histogram: A fast way to extract histograms in cartesian spaces. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2005.

[19] Y. Rubner. Code available at http://ai.stanford.edu/∼rubner/.

[20] Y. Rubner, C. Tomasi, and L. Guibas. The earth mover's distance as a metric for image retrieval. *Int. Journal of Computer Vision (IJCV)*, 40(2):91–121, 2000.

[21] C. Schmid and R. Mohr. Local gray-value invariants for image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1997.

[22] C. Shen, M. Brooks, and A. Hengel. Fast global kernel density mode seeking with application to localisation and tracking. In *Proc. IEEE Int. Conf. on Computer Vision (ICCV)*, 2005.

[23] L. Sigal et al. Tracking loose-limbed people. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2004.

[24] S. Ullman, E. Sali, and M. Vidal-Naquet. A fragment-based approach to object representation and classification. In *Proc. IWVF4, LNCS 2059*, pages 85–100, 2001.

[25] P. Viola and M. Jones. Robust real time object detection. In *IEEE ICCV Workshop on Statistical and Computational Theories of Vision*, 2001.

[26] C. Yang, R. Duraiswami, and L. Davis. Efficient mean-shift tracking via a new similarity measure. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2005.