Efficient Visual Object Tracking with Online Nearest Neighbor Classifier

Steve Gu and Ying Zheng and Carlo Tomasi

Department of Computer Science, Duke University

Abstract. A tracking-by-detection framework is proposed that combines nearest-neighbor classification of bags of features, efficient subwindow search, and a novel feature selection and pruning method to achieve stability and plasticity in tracking targets of changing appearance. Experiments show that near-frame-rate performance is achieved (*sans* feature detection), and that the state of the art is improved in terms of handling occlusions, clutter, changes of scale, and of appearance. A theoretical analysis shows why nearest neighbor works better than more sophisticated classifiers in the context of tracking.

1 Introduction

Visual object tracking is crucial to visual understanding in general, and to many computer vision applications ranging from surveillance and robotics to gesture and motion recognition. The state of this art has advanced significantly in the past 30 years [1–8]. Recently, advances in apparently unrelated areas have given tracking a fresh impulse: Specifically, progress in the definition of features invariant to various imaging transformations [9, 10], online learning [11, 12], and object detection [13–16] have spawned the approach of *tracking by detection* [17–21], in which a target object identified by the user in the first frame is described by a set of features. A separate set of features describes the background, and a binary classifier separates target from background in successive frames. To handle appearance changes, the classifier is updated incrementally over time. Motion constraints restrict the space of boxes to be searched for the target.

In a recent example of this approach, Babenko *et al.* [20] adapt Multiple Instance Learning (MIL) [12, 11] by building an evolving boosting classifier that tracks bags of image patches, and report excellent tracking results on challenging video sequences. The main advantages of tracking by detection come from the flexibility and resilience of its underlying representation of appearance. Several parametric learning techniques such as Support Vector Machines (SVM, [22]), boosting [20], generative models [23], and fragments [24] have been used successfully in tracking by detection. More recently, Santner *et al.* propose a sophisticated tracking system called PROST [21] that achieves top performance with a smart combination of three trackers: template matching based on normalized cross correlation, mean shift optical flow [25], and online random forests [26] to predict the target location.

However, since computation occurs at frame rate, efficiency in both appearance learning and target/background classification is a paramount consideration for practical systems. In addition, image boxes that might contain the target must be enumerated quickly. Finally, and perhaps most fundamentally, the so-called *stability-plasticity dilemma* [21] must be addressed: a *stable* description of target appearance, based only on the first frame, can handle occlusions well, but fails to track an object whose appearance changes over time. A more *plastic* description can be obtained by updating features from observations in subsequent frames, but at the cost of potential confusion between target and foreground when incorrectly classified features contaminate the training sets.

To address these issues, we propose to use Nearest Neighbor (NN) as the underlying, non-parametric classifier; Efficient Subwindow Search (ESS, [15]) to hypothesize target locations; and a novel feature updating and pruning method to achieve a proper balance between plasticity and stability. Despite the simplicity of the NN classifier, Boiman *et al.* [27] demonstrate its state-of-art performance for object categorization. We analyze NN geometrically to suggest why NN captures appearance change better than competing methods. In addition, NN requires no training other than data collection, and is efficient when the size of the data sample is small, as is the case in tracking by detection. Further efficiency can be achieved by the use of KD trees. The use of ESS leads to improved handling of scale changes over existing state-of-the-art trackers [20, 21], which cannot search variable-size windows. For features, we use SIFT [9], although other, more recent methods such as SURF [10] or Self-Similarity [28] could be used without further modification. Together with our feature update and pruning method, this combination leads to a unified tracking-by-detection framework that handles appearance changes, occlusion, background clutter, and scale changes in a principled and effective way, as our experiments demonstrate.

2 Tracking with an Online Nearest-Neighbor Classifier

For ease of exposition, we describe our appearance and motion model separately first. We then show how to integrate them seamlessly via online NN classification into a simple and efficient algorithm. Finally, we discuss why our tracking framework can handle significant background clutter, scale changes, occlusion and appearance change.

2.1 Appearance Model

Let $V(I) = \{(\mathbf{x}_1, \mathbf{v}_1), \cdots, (\mathbf{x}_n, \mathbf{v}_n)\}$ be the set of (SIFT) key points of an image I where $\mathbf{x}_i \in \mathbb{R}^2$ is the 2D coordinate and $\mathbf{v}_i \in \mathbb{R}^d$ is the d-dimensional descriptor vector of the i^{th} feature. We use $\Theta(W; I)$ to represent the set of key point descriptors of I within the window W: $\Theta(W; I) \triangleq \left\{\mathbf{v} \in \mathbb{R}^d \mid (\mathbf{x}, \mathbf{v}) \in V(I), \mathbf{x} \in W\right\}$. Given an image sequence $(I_0, W_0), (I_1, W_1), \cdots, (I_k, W_k)$ where W_i is the tracked window in image I_i, we describe how to compute the features belonging to the object and the background respectively. Let $\mathcal{B} \subset \mathbb{R}^d$ be the static background

model and let $\mathcal{O}_k \subset \mathbb{R}^d$ be the dynamic object model updated up to the frame index k. Initially we set $\mathcal{O}_0 = \Theta(W_0; I_0)$ and put the rest of the key point descriptors into the background model $\mathcal{B}$. Given $\mathcal{O}_{k-1}$ and W_k, we compute:

$$\mathcal{O}_k \leftarrow \mathcal{O}_{k-1} \bigcup \mathcal{F}_\lambda \left[\Theta(W_k; I_k), \mathcal{O}_{k-1}, \mathcal{B}\right] \tag{1}$$

where $\mathcal{F}_\lambda : 2^{\mathbb{R}^d} \times 2^{\mathbb{R}^d} \times 2^{\mathbb{R}^d} \rightarrow 2^{\mathbb{R}^d}$ is the *filter* operator on the feature set:

$$\mathcal{F}_\lambda \left[A, B, C\right] \triangleq \{\mathbf{v} \in A \mid \|\mathbf{v} - NN_B(\mathbf{v})\| < \lambda \|\mathbf{v} - NN_C(\mathbf{v})\|\} \tag{2}$$

where $NN_B(\mathbf{v}) \triangleq \arg\min_{\mathbf{u} \in B} \|\mathbf{u} - \mathbf{v}\|$ is the nearest neighbor of $\mathbf{v}$ in the set B.

The idea behind the design of $\mathcal{F}_\lambda$ is simple: for each feature $a \in \Theta(W_k; I_k)$, we apply the ratio test in (2) to see if a is close enough to the target set $\mathcal{O}_{k-1}$ when compared to its distance to the background set $\mathcal{B}$, and we only preserve those features in $\Theta(W_k; I_k)$ that pass the ratio test. Here λ is the selection criterion and is fixed to $2/3$, analogously to the familiar matching criterion used in SIFT[9]. We thus have an model updating scheme that keeps adjusting the appearance change while avoiding confusion between object and background features.

2.2 Motion Model

Given $\mathcal{O}_{k-1}, \mathcal{B}, W_{k-1}$ and I_k, the motion model aims to locate an optimal window W_k that encloses the current object. To this end, we need to evaluate a window based on its appearance. We propose to use the following score function:

$$\mu(W; \mathcal{O}_{k-1}, \mathcal{B}, W_{k-1}, I_k) = \underbrace{\sum_{\mathbf{v} \in \Theta(W; I_k)} \operatorname{sign}\left(\mathcal{F}_\lambda\left[\{\mathbf{v}\}, \mathcal{O}_{k-1}, \mathcal{B}\right]\right)}_{\text{appearance similarity}} - \underbrace{\kappa(W, W_{k-1})}_{\text{motion penalty}} \tag{3}$$

where $\operatorname{sign}(A) = 1$ if $A \neq \emptyset$ and -1 otherwise. In the implementation, we can also promote matched features by assigning scores greater than 1. The penalty function $\kappa(W, W_{k-1})$ measures the difference between the window W and the window W_{k-1} in terms of position drift and shape change. We set:

$$\kappa(W_1, W_2) = \gamma \left(\underbrace{\|O_1 - O_2\|}_{\text{position drift}} + \underbrace{|h_1 - h_2|}_{\text{height}} + \underbrace{|w_1 - w_2|}_{\text{width}} + s(W_1, W_2) \right) \tag{4}$$

where $\|O_1 - O_2\|$ is the distance between the centroids O_1 and O_2 of the windows W_1 and W_2, and $(w_1, h_1), (w_2, h_2)$ are the width and height of W_1 and W_2. The term $s(W_1, W_2) \triangleq \max\left\{|\frac{h1}{w1} - \frac{h2}{w_2}|, |\frac{w_1}{h_1} - \frac{w_2}{h_2}|\right\}$ then penalizes changes in the aspect ratio although other more sophisticated penalties are appropriate here as well. Finally, γ measures the relative importance of the penalty function in the score $\mu(W; \mathcal{O}_{k-1}, \mathcal{B}, W_{k-1}, I_k)$. The optimal window W_k in I_k is then

$$W_k = \arg\max_W \mu(W; \mathcal{O}_{k-1}, \mathcal{B}, W_{k-1}, I_k) \tag{5}$$

where W can be an arbitrary window within the domain of the image I_k. The presence of the penalty function $\kappa(W, W_{k-1})$ ensures that the current tracked window cannot drift or change its shape arbitrarily relative to the previous tracked window. Notice that the motion model does not take advantage of any motion continuity or locality. In practice, as is typical to many trackers, we could restrict the search region to a local image patch instead of the entire image domain. Although we believe that the motion model can be improved by introducing advanced filtering techniques such as a Kalman filter or a particle filter, we do not include those techniques in order to show that our tracker already yields excellent tracking results even without them.

2.3 Algorithm and Implementation

The algorithm for tracking is very simple:

Input: the object model $\mathcal{O}_{k-1}$, previous window W_{k-1} and the image I_k.
Output: the updated object model $\mathcal{O}_k$ and the current tracked window W_k.
Step.1: $W_k \leftarrow \arg\max_W \mu(W; \mathcal{O}_{k-1}, \mathcal{B}, W_{k-1}, I_k)$
Step.2: $\mathcal{F}_\lambda \leftarrow \left\{\mathbf{v} \in \Theta(W_k; I_k) \mid \|\mathbf{v} - NN_{\mathcal{O}_{k-1}}(\mathbf{v})\| < \lambda \|\mathbf{v} - NN_{\mathcal{B}}(\mathbf{v})\|\right\}$
Step.3: $\mathcal{O}_k \leftarrow \mathcal{O}_{k-1} \bigcup \mathcal{F}_\lambda$

The nearest neighbor search can be computed quite efficiently with KD trees if approximation is allowed. We use published software [29] to compute both the SIFT descriptors and the nearest neighbor query, with two modifications for efficiency: First, at any time we keep only features from the most recent τ frames, so that the total number of features in $\mathcal{O}_k$ ranges only from hundreds to thousands, enabling real time performance. Second, we do not explicitly modify the KD tree data structure during the update. Instead, we always maintain $1 + \tau$ KD trees, each corresponding to a frame, and whenever a frame is added or deleted, we add or remove the entire tree associated to that frame. The KD tree associated to the first frame is never deleted. The background model is assumed to be static in our implementation. However, we could easily update the background model as well, similarly to what we do with the object model.

Fig. 1. The method of efficient subwindow search ensures that our tracker can handle significant scale change efficiently. The video sequence is downloaded from YouTube.

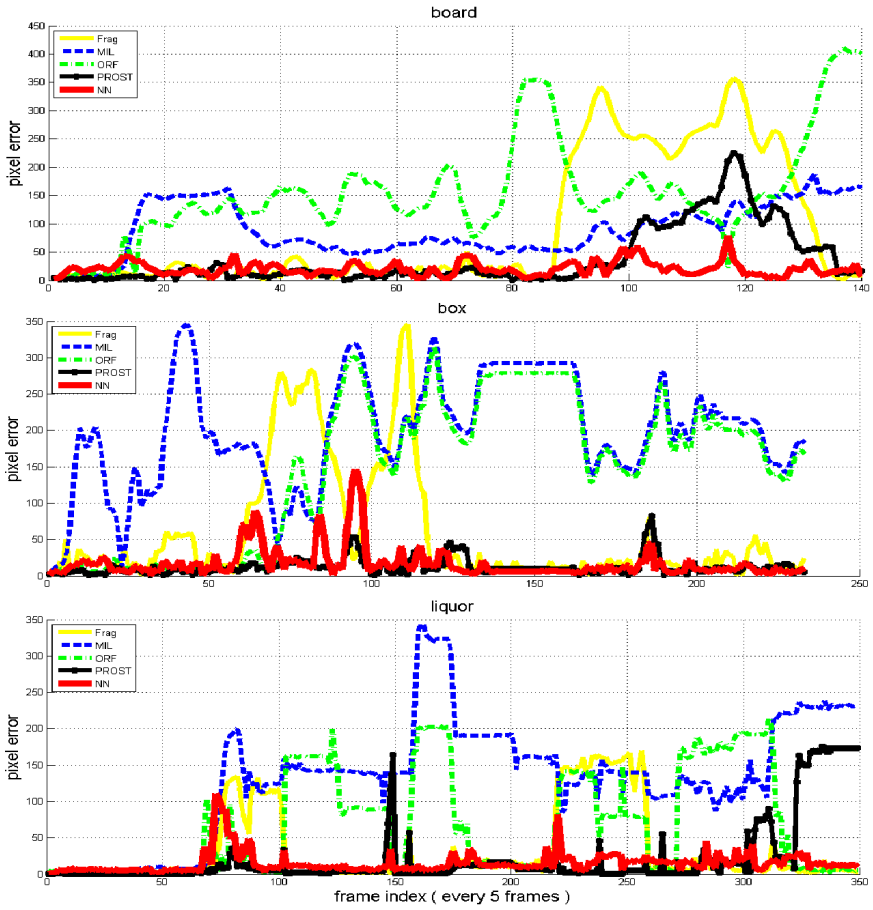

Fig. 2. Mean distance error for the selected sequences frame by frame

Exhaustive subwindow search is needed in Step.1. Although the worst running time is $O(n^2)$ where n is the number of pixels in the search area, the actual performance of the optimal window search is often close to sublinear time thanks to branch and bound [15]. We adapt this Efficient Subwindow Search (ESS) method by applying a penalty on position drift and shape change. Specifically, let $\mathcal{R}$ be the range space containing all possible windows and W_{k-1} the tracked window in the previous frame. The quality function $f^{\text{new}}(\mathcal{R})$ is modified from the quality function $f^{\text{old}}(\mathcal{R})$ by Lampert et al.[15] so that

$$f^{\text{new}}(\mathcal{R}) = f^{\text{old}}(\mathcal{R}) - \min_{W \in \mathcal{R}} \kappa(W, W_{k-1}) \tag{6}$$

It is not difficult to prove that this quality function satisfies the two constraints for branch and bound: (1) $f^{\text{new}}(\mathcal{R}) \geq \mu(W; \mathcal{O}_{k-1}, \mathcal{B}, W_{k-1}, I_k)$ for each window

$W \in \mathcal{R}$; and (2)$f^{\text{new}}(\mathcal{R})$ converges to the true score when there is a unique window in $\mathcal{R}$. The quality function $f^{\text{new}}(\mathcal{R})$ can be evaluated in constant time by pre-computing the *integral image* in linear time.

The running time of our algorithm, excluding the time for computing SIFT descriptors, is close to frame rate in our MATLAB implementation on a single-core laptop. Computing the key point descriptors is relatively slow (several frames per second) compared to tracking time. However, we compute the SIFT descriptors for the entire image for the ease of experiment, and this is not necessary. One speed up is that we only compute features in a local image area. Another speed up comes from other choices of key point descriptors. Finally, we could port the program to C/C++ and run it on a more advanced computer.

3 Analysis

We explain why our tracking framework can handle background clutter, occlusion, scale and appearance change in a principled way.

Background Clutter. Our tracking framework avoids the confusion between target and background features by applying the filter operator $\mathcal{F}_\lambda$ on the feature set within the tracked window. Therefore, only those features that resemble the features in the current object model are updated.

Scale changes. The tracking model by equation (5) ensures that we search windows with all possible locations and shapes, and this covers even significant scale changes. The other contributing factor comes from the use of SIFT descriptors that have been demonstrated to handle partial scale changes well. Notice that our tracking framework does not depend on a specific key point descriptor, and should benefit from any improvement in this area. In Figure 1 we show that our tracker can handle significant scale change.

Occlusion. We follow the bag-of-features approach which naturally handles occlusion. The score of a window is high if there are enough matched features inside the window. If only few matched features are present, there is a strong indication that the object is occluded. The current window W_k stays stable relative to the previous window W_{k-1} when occluded because of the penalty enforced by $\kappa(W_k, W_{k-1})$. We exhibit different cases of occlusion in Figure 4.

Appearance Changes. The most challenging part of tracking-by-detection systems is perhaps how to adapt to the appearance change incrementally. We show in the left column of Figure 6 that our tracker can adjust to appearance change fairly well. Interestingly we can directly estimate the "shape" of the *object feature space*, the set $\mathcal{F}_\lambda(\mathbb{R}^d, O, B)$ where O and B are the set of object and background features. As a result, we can predict the set of features that is allowed in the object model O. We recall from classical geometry that for $\lambda < 1$, the inequality $\|\mathbf{x} - \mathbf{a}\| < \lambda\|\mathbf{x} - \mathbf{b}\|$ defines a d-dimensional hyper

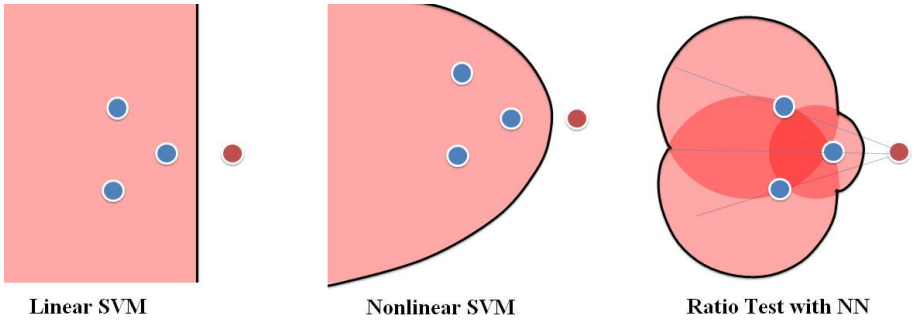

Fig. 3. The two dimensional object feature space (the pink shaded area) generated by linear (left) and nonlinear (middle) support vector machines and the nearest neighbor classifier (right) where the blue and red dots are the features belonging to the object and the background. The ratio test with the NN classifier produces an object feature space that is bounded by the union of a set of two dimensional disks.

ball: $\mathfrak{B}_\lambda(\mathbf{a},\mathbf{b}) \triangleq \left\{\mathbf{x}\in\mathbb{R}^d \mid \|\mathbf{x}-\frac{\mathbf{a}-\lambda^2\mathbf{b}}{1-\lambda^2}\| < \frac{\lambda}{1-\lambda^2}\|\mathbf{a}-\mathbf{b}\|\right\}$ with centroid located at $\frac{\mathbf{a}-\lambda^2\mathbf{b}}{1-\lambda^2}$ and radius equal to $\frac{\lambda}{1-\lambda^2}\|\mathbf{a}-\mathbf{b}\|$. We therefore can quantify the volume of the object feature space $\mathcal{F}_\lambda(\mathbb{R}^d,O,B)$ using the intersection of Voronoi regions. Let $\mathrm{Vor}(\mathbf{v};S) \triangleq \left\{\mathbf{x}\in\mathbb{R}^d \mid NN_S(\mathbf{x})=\mathbf{v}\right\}$ be the Voronoi region of $\mathbf{v}$ spanned by S and let $\mathcal{I}(S_1,S_2) = \{(\mathbf{a},\mathbf{b})\in S_1\times S_2 \mid \mathrm{Vor}(\mathbf{a};S_1)\bigcap\mathrm{Vor}(\mathbf{b};S_2)\neq\emptyset\}$ be the intersection of the Voronoi regions of the set S_1 and S_2 respectively. We have:

Theorem 1 (Ball Cover). *The object feature space is bounded by the union of a set of d-dimensional hyper balls:* $\mathcal{F}_\lambda(\mathbb{R}^d,O,B)\subseteq\bigcup_{(\mathbf{a},\mathbf{b})\in\mathcal{I}(O,B)}\mathfrak{B}_\lambda(\mathbf{a},\mathbf{b})$

Proof. Since the set of nearest neighbors to the feature $\mathbf{v}\in O$ is expressed by $\mathbf{v}$'s Voronoi region spanned by O, we can rewrite the object feature space:

$$\mathcal{F}_\lambda(\mathbb{R}^d,O,B) = \bigcup_{\mathbf{a}\in O}\bigcup_{\mathbf{b}\in B}\left[\mathrm{Vor}(\mathbf{a};O)\bigcap\mathrm{Vor}(\mathbf{v};B)\bigcap\mathfrak{B}_\lambda(\mathbf{a},\mathbf{b})\right] \tag{7}$$

$$= \bigcup_{(\mathbf{a},\mathbf{b})\in\mathcal{I}(O,B)}\left[\Big(\mathrm{Vor}(\mathbf{a};O)\bigcap\mathrm{Vor}(\mathbf{b};B)\Big)\bigcap\mathfrak{B}_\lambda(\mathbf{a},\mathbf{b})\right] \tag{8}$$

$$\subseteq \bigcup_{(\mathbf{a},\mathbf{b})\in\mathcal{I}(O,B)}\mathfrak{B}_\lambda(\mathbf{a},\mathbf{b})\ . \tag{9}$$

The ball cover theorem has many appealing properties: First, it shows that the object feature space is well bounded by our selection criterion. This is in contrast to the typical use of classifiers such as linear or non-linear SVM where the object feature space is unbounded. We illustrate this difference in the 2D cartoon in Figure 3. Second, for $\mathbf{a},\mathbf{b}\in O$ with $\|\mathbf{a}-NN_B(\mathbf{a})\| > \|\mathbf{b}-NN_B(\mathbf{b})\|$, $\mathbf{a}$ contributes more than $\mathbf{b}$ to the object feature space. In other words, the more discriminative (between target and background) the feature is, the larger the feature space it generates (*i.e.*, the larger the radius of the hyper ball).

4 Experiments

We test our new NN-based tracker on 8 video sequences from multiple data sets [30, 31, 5, 20, 21] and compare it to the state-of-art tracking methods: AdaBoost [18], Online Random Forests [26], Fragments [5], Multiple Instance Learning [20] and PROST [21]. In the comparison, we directly quote the results from [21] where the best results from each method were reported. This comparison is summarized in Table 1. In Figure 4 and Figure 6, we show the actual performance of the different trackers. The selected error plots are shown in Figure 2. For more results and comparisons, please see the supplementary material. In the experiment, we fix the parameters once for all (i.e. $\lambda = \frac{2}{3}, \gamma = 0.1$) and use the default SIFT parameters in [29]. The evaluation criterion is the same as what is used in [20, 21] except that we only compute the mean distance error e:

$$e = \frac{1}{n} \sum_{i=1}^{n} \|O_i - O_i^g\| \tag{10}$$

where n is the number of frames and $\|O_i - O_i^g\|$ is the Euclidean distance between the tracked window centroid O_i and the ground truth window centroid O_i^g.

Table 1 shows that despite the simplicity of the proposed method, our tracker yields excellent tracking results often close to or even better than the state-of-art methods, which rely on more sophisticated online learning methods. More specifically, our tracker achieves the best result in the sequences Girl, Faceocc2, Board and Liquor, and the second best result in the sequences David and Box. These results verify that our tracking framework can handle significant occlusion, background clutter and appearance change. Our tracker can successfully and closely follow the object in 7 out of the 8 video sequences and wins in the highest number of trials. PROST achieves excellent results as well, followed by the MIL and Frag tracker.

Table 1. Mean distance error to the ground truth. Bold: best. Underlined: second best.

Sequences	# Frames	AdaBoost	ORF	FragTrack	MILTrack	PROST	NN
Girl [30]	502	43.3	–	26.5	31.6	<u>19.0</u>	**18.0**
David [31]	462	51.0	–	46.0	<u>15.6</u>	**15.3**	<u>15.6</u>
Faceocc1 [5]	886	49.0	–	**6.5**	18.4	<u>7.0</u>	10.0
Faceocc2 [20]	812	19.6	–	45.1	<u>14.3</u>	17.2	**12.9**
Board [21]	698	–	154.5	90.1	51.2	<u>37.0</u>	**20.0**
Box [21]	1161	–	145.4	57.4	104.6	**12.1**	<u>16.9</u>
Lemming [21]	1336	–	166.3	82.8	**14.9**	<u>25.4</u>	79.1
Liquor [21]	1741	–	67.3	30.7	165.1	<u>21.6</u>	**15.0**

4.1 Failure Modes

Our tracker is not without its limitations. For instance, it failed to follow the target closely in the Lemming sequence (Figure 5). There are three major limiting factors: First, experiments show that the SIFT descriptor cannot handle motion blur well, because no features are found in regions of uniform texture. For the same reason, very few features can be reliably found on the body of the lemming even when the object is static. Therefore, the current NN tracker would benefit from descriptors that capture uniform regions and motion blur better. Second, our tracker does not utilize any advanced motion models. Consequently, if the feature matching step fails completely, i.e. generates uniform matching scores at all the pixels, our tracker stays still due to the presence of a motion penalty. How to use advanced filtering technique to cope with matching failure under heavy background clutter is left for future work. Third, the current tracker cannot localize objects very precisely when the object's shape deforms. This is because a rectangle that is axis-parallel to the image boundaries is used to bound the target. How to localize regions of varying shape and orientations in an efficient manner is an interesting challenge.

5 Conclusions and Future Work

The combination of nearest-neighbor classification of bags of features, efficient subwindow search, and our novel feature selection and pruning method yields a simple and efficient tracking-by-detection algorithm that handles occlusions, clutter, and significant changes of scale and appearance. Computation occurs at near-frame-rate (without counting feature detection) even in a relatively naive Matlab implementation. Performance quality in terms of stability and plasticity is competitive and often better than the previous state of the art. Our theoretical analysis suggests some of the reasons why nearest neighbor works better than more sophisticated classifiers in this context. Immediate future work entails improvements of implementation, as suggested earlier, for true real-time performance, and the use of more recent feature detection schemes. Longer term questions concern the incorporation of more detailed motion models, backup search strategies for lost objects, tracking complex objects by parts, the exploitation of *a priori* appearance models for certain categories of targets (e.g., people, cars), and how to select target without user intervention.

Acknowledgement: This material is based upon work supported by the NSF under Grant IIS-1017017 and by the ARO under Grant W911NF-10-1-0387.

References

1. Lucas, B., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: IJCAI. (1981) 674–679
2. Shi, J., Tomasi, C.: Good features to track. In: IEEE CVPR. (1994) 593 – 600
3. Isard, M., Blake, A.: A smoothing filter for condensation. In: ECCV. (1998) 767–781

Fig. 4. From top to bottom: Girl, Faceocc1, Faceocc2 and David. Faceocc1 and Faceocc2 have significant occlusion. David experiences appearance change. Girl has both occlusion and appearance change. For ease of visualization, we show only the comparison between the NN-based tracker (solid red) and the MIL tracker (dashed blue). NN follows the object more closely and handles occlusion better than MIL.

4. Comaniciu, D., Ramesh, V., Meer, P.: Real-time tracking of non-rigid objects using mean shift. In: IEEE CVPR. (2000) 2142–
5. Adam, A., Rivlin, E., Shimshoni, I.: Robust fragments-based tracking using the integral histogram. In: IEEE CVPR. (2006) 798–805
6. Avidan, S.: Ensemble tracking. IEEE PAMI **29** (2007) 261–271
7. Li, Y., Ai, H., Yamashita, T., Lao, S., Kawade, M.: Tracking in low frame rate video: A cascade particle filter with discriminative observers of different life spans. IEEE PAMI **30** (2008) 1728–1740
8. Özuysal, M., Calonder, M., Lepetit, V., Fua, P.: Fast keypoint recognition using random ferns. IEEE Trans. Pattern Anal. Mach. Intell. **32** (2010) 448–461
9. Lowe, D.: Object recognition from local scale-invariant features. In: ICCV. (1999) 1150–1157
10. Bay, H., Tuytelaars, T., Gool, L.: Surf: Speeded up robust features. In: ECCV. (2006) 404–417
11. Viola, P., Platt, J., Zhang, C.: Multiple instance boosting for object detection. In: NIPS. (2005)
12. Dietterich, T., Lathrop, R., Lozano-Pérez, T.: Solving the multiple instance problem with axis-parallel rectangles. Artif. Intell. **89** (1997) 31–71

Fig. 5. Our tracker failed in the Lemming sequence for the reason that the object contains significant motion blur which is not well captured by the SIFT descriptors

13. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: IEEE CVPR. (2005) 886–893
14. Felzenszwalb, P., McAllester, D., Ramanan, D.: A discriminatively trained, multiscale, deformable part model. In: IEEE CVPR. (2008)
15. Lampert, C., Blaschko, M., Hofmann, T.: Efficient subwindow search: A branch and bound framework for object localization. IEEE PAMI **31** (2009) 2129–2142
16. Everingham, M., Gool, L., Williams, C., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes Challenge 2009 (VOC2009) Results. http://www.pascal-network.org/challenges/VOC/voc2009/workshop/index.html (2009)
17. Tomasi, C., Petrov, S., Sastry, A.: 3d tracking = classification + interpolation. In: ICCV. (2003) 1441–1448
18. Grabner, H., Bischof, H.: On-line boosting and vision. In: IEEE CVPR. (2006) 260–267
19. Grabner, H., Leistner, C., Bischof, H.: Semi-supervised on-line boosting for robust tracking. In: ECCV. (2008) 234–247
20. Babenko, B., Yang, M., Belongie, S.: Visual tracking with online multiple instance learning. In: IEEE CVPR. (2009) 983–990
21. Santner, J., Leistner, C., Saffari, A., Pock, T., Bischof, H.: PROST Parallel Robust Online Simple Tracking. In: IEEE CVPR. (2010)
22. Tian, M., Zhang, W., Liu, F.: On-line ensemble svm for robust object tracking. In: ACCV. (2007) 355–364
23. Zhao, X., Liu, Y.: Generative estimation of 3d human pose using shape contexts matching. In: ACCV. (2007) 419–429
24. Prakash, C., Paluri, B., Pradeep, S., Shah, H.: Fragments based parametric tracking. In: ACCV. (2007) 522–531
25. Werlberger, M., Trobin, W., Pock, T., Wedel, A., Cremers, D., Bischof, H.: Anisotropic huber-l1 optical flow. In: BMVC. (2009)
26. Breiman, L.: Random forests. Mach. Learning **45** (2001) 5–32
27. Boiman, O., Shechtman, E., Irani, M.: In defense of nearest-neighbor based image classification. In: IEEE CVPR. (2008)
28. Shechtman, E., Irani, M.: Matching local self-similarities across images and videos. In: IEEE CVPR. (2007)
29. Vedaldi, A., Fulkerson, B.: VLFeat: An open and portable library of computer vision algorithms. http://www.vlfeat.org/ (2008)
30. Birchfield, S.: Elliptical head tracking using intensity gradients and color histograms. In: IEEE CVPR. (1998) 232–237
31. Ross, D., Lim, J., Lin, R., Yang, M.: Incremental learning for robust visual tracking. IJCV **77** (2008) 125–141

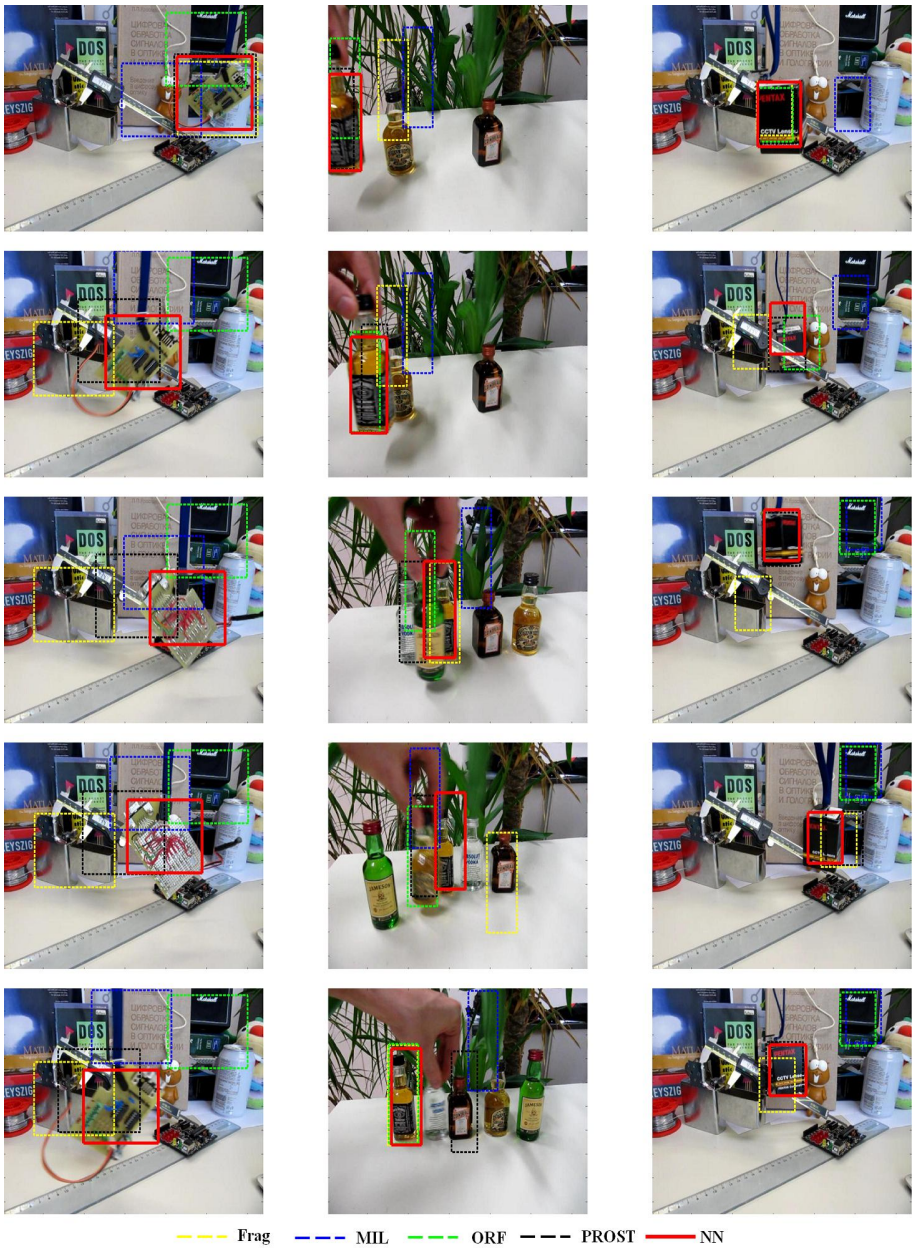

Fig. 6. Each column (left: board; middle:liquor; right:box) shows the performance of NN, MIL, Frag, PROST and ORF on selected frames. NN typically outperforms all other methods in cases with significant occlusion, scale and appearance change.