Fragments Based Tracking with Adaptive Cue Integration

Erkut Erdem[a,1,*], Séverine Dubuisson[c], Isabelle Bloch[b]

[a] *Department of Computer Engineering, Hacettepe University, Ankara, Turkey*
[b] *Institut TELECOM, Télécom ParisTech, CNRS LTCI, Paris, France*
[c] *Laboratoire d'Informatique de Paris 6, UPMC, Paris, France*

Abstract

In this paper, we address the issue of part-based tracking by proposing a new fragments-based tracker. The proposed tracker enhances the recently suggested *FragTrack* algorithm to employ an adaptive cue integration scheme. This is done by embedding the original tracker into a particle filter framework, associating a reliability value to each fragment that describes a different part of the target object and dynamically adjusting these reliabilities at each frame with respect to the current context. Particularly, the vote of each fragment contributes to the joint tracking result according to its reliability, and this allows us to achieve a better accuracy in handling partial occlusions and pose changes while preserving and even improving the efficiency of the original tracker. In order to demonstrate the performance and the effectiveness of the proposed algorithm we present qualitative and quantitative results on

*Corresponding author.
Email addresses: erkut@cs.hacettepe.edu.tr (Erkut Erdem), severine.dubuisson@lip6.fr (Séverine Dubuisson), isabelle.bloch@telecom-paristech.fr (Isabelle Bloch)

[1] This study was completed when the author was with the Institut TELECOM, Télécom ParisTech, CNRS LTCI, Paris, France.

a number of challenging video sequences.

Keywords: video sequences, object tracking, particle filter, fragment-based trackers, adaptive cue integration

---

1. Introduction

Tracking an object in an accurate way is essential for applications like activity analysis, man-machine interaction and visual surveillance. However, for many real-world problems, the ambiguities inherent to the visual data and the tracking process make it difficult to develop accurate, robust and efficient trackers. During the tracking process, usually, the target object becomes occluded by the other objects in the scene, its pose or appearance undergoes some changes, or the lighting conditions vary.

A common solution to these issues is to use complementary observations/cues from different sources. Within such a strategy, each cue provides a likelihood or a matching score for the possible positions of the object, and the tracker determines the final output by the product of individual likelihoods or the summation of the matching scores. This highly improves the tracking performance. In the literature, such tracking frameworks are named *multi-cue trackers*.

Multi-cue trackers require to use visual cues that are orthogonal to each other as much as possible so that if one cue fails, the other cue or cues can compensate its deficiency. There are mainly two ways to obtain such orthogonal cues [1]. One possibility is to consider visual cues that express different features of the target object. An alternative solution is to consider a single visual feature and to use it to describe different sections of the

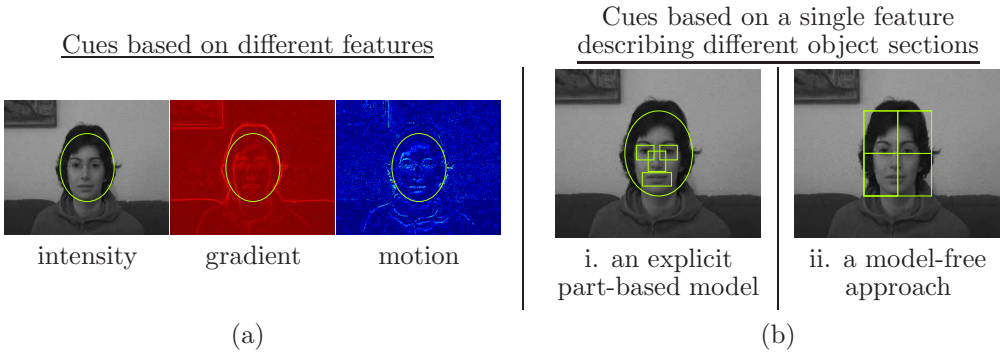

Figure 1: Multi-cue trackers employ complementary visual cues, which provide rich information about the target object, to improve the tracking performance. These visual cues are based on either (a) different features (e.g., intensity, gradient, motion) or (b) a single feature (e.g., only intensity), which is used to describe different object sections via i. a part-based model or ii. a model-free approach.

target object within a single framework. We provide a sample illustration of approaches to multi-cue tracking in Figure 1.

The trackers in the first line of works determine the target position at each frame by combining cues that are based on several different visual features [2, 3, 4] (Figure 1(a)). For example, in one of early studies [2], Birchfield suggested to track heads using intensity gradients and color histograms together. The critical point with this approach is that during tracking the cues do not always provide reliable information about the target object. Giving equal importance to all the features in determining the combined result generally leads to false outcomes. Triesch and von der Malsburg [5] proposed a dynamic framework called the *Democratic Integration* to adaptively integrate different cues, addressing this problem. In their framework, each cue has an adaptive reliability value associated with it, and each cue contributes to the joint result according to its reliability. A number of studies, e.g. [1, 6, 7],

follow such a strategy by performing an adaptive integration of cues to boost the accuracy of the tracking process.

Considering a single feature and using it to describe different sections of the target object via an explicit part-based model is another strategy (Figure 1(b) i.). For instance, in human trackers a human can often be subdivided into parts corresponding to head, limbs and torso [1, 8]. Splitting an object into parts introduces a kind of supplementary shape information regarding the target object by providing the relative spatial arrangements of different object sections. This offers an important advantage over the classical region-based trackers where the content of the region of interest is modeled with a single histogram with the loss of spatial information. This kind of part-based trackers mostly aims at tracking articulated/non-rigid objects (e.g., [8, 9]), and generally requires the model of the target object to be known or given a priori.

Although one can easily devise such models for domain-specific trackers (e.g. for faces or humans) and thus make use of strong prior information, it is not convenient to use this approach for generic object tracking. The recently proposed robust fragments-based tracker called *FragTrack* [10] mainly tackled this issue. In that study, the target object is again represented with multiple image fragments or patches each describing a different object section, but these patches are extracted arbitrarily without considering any reference object model (Figure 1(b) ii.). This makes the suggested part-based tracking algorithm applicable to any object without changing the representation. Particularly, at each frame of the sequence, different object sections vote on possible target locations and scales, and the tracker combines these

Table 1: Multi-cue Tracking Algorithms

Tracking Algorithm	Feature(s) used for tracking	Model-free approach	Adaptive integration of cues	Part-based target model	Online learning of appearance
Birchfield [2]	color, edge	no	no	no	no
Wu and Huang [3]	color, shape	no	no	no	no
Pérez, Vermaak and Blake [4]	color, {sound or motion}	no	no	no	no
Triesch and von der Malsburg [5]	color, motion, shape, contrast	no	yes	no	no
Maggio, Smeraldi and Cavallaro [6]	color, orientation	no	yes	no	no
Brasnet et al. [7]	color, texture, shape	no	yes	no	no
Nickel and Stiefelhagen [1]	color, motion, detector, stereo	no	yes	yes	no
Nejhum, Ho and Yang [8]	intensity	no	yes	yes	no
Grabner, Grabner and Bischof [11]	Haar-like features	yes	n/a	no	yes
Grabner, Leistner and Bischof [12]	Haar-like features	yes	n/a	no	yes
Woodley, Stenger and Cipolla [13]	Subspace model	yes	n/a	no	yes
Babenko, Yang and Belongie [14]	Haar-like features	yes	n/a	no	yes
Adam, Rivlin and Shimshoni [10]	intensity	yes	no	no	no
Ours	intensity	**yes**	**yes**	no	no

votes using a robust statistics scheme to obtain an outcome by performing an exhaustive search on this combined vote map.

Another line of works that also tackles model-free tracking includes boosting-based tracking frameworks [11, 12, 13, 14], and it has drawn quite a bit of attention lately. Like in the *FragTrack*, in these works the tracking is cast as a sequential detection problem but the detection is performed via on-line learning of an object-specific classifier using boosting. The common approach is to represent the target object by a set of Haar-like features and to define the strong classifier which serves as the object detector as an adaptive combination of several weak classifiers that are selected from the most discriminative features in the current set. The aforementioned multi-cue tracking strategies are summarized in Table 1.

In this paper, we also address the issue of part-based object tracking. Our

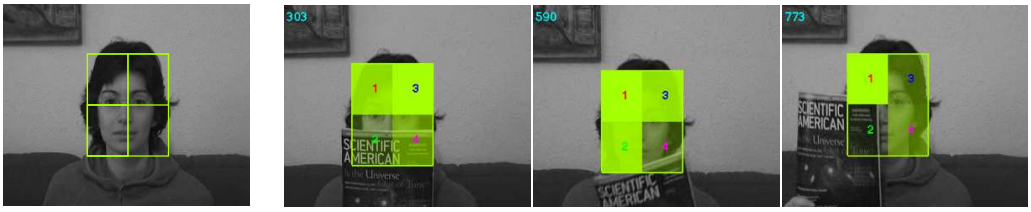

Figure 2: The proposed multi-cue tracker combines the arbitrary-fragments based object representation [10] with the concept of adaptive multi-cue integration [5]. A reliability value is defined for each fragment, and it is dynamically adjusted at each frame with respect to the current context. Each fragment contributes to the joint tracking result according to its reliability, and the ones having low values have little effect on the outcome. For the sample track illustrated in this figure, the reliabilities are shown in green tones with the higher values shown in high intensity. As can be seen, when a fragment becomes occluded during tracking, the dynamic framework decreases its reliability automatically, and by this way, tracking is affected to a minimal degree.

proposed model-free tracker combines the *FragTrack*'s arbitrary-fragments based object representation [10] and the concept of adaptive multi-cue integration [5] (Figure 2). Our main contributions can be summarized as follows:

- Our algorithm employs *an adaptive cue integration scheme* [5]. A reliability value is associated to each fragment that describes a different part of the target object and it is dynamically adjusted at each frame with respect to the current context. The vote of each fragment contributes to the joint tracking result according to its reliability, and the ones having low values have little effect on the outcome. This allows us to achieve a better tracking accuracy in handling partial occlusions and pose changes. Note that *FragTrack* does not assign reliabilities to fragments and does not adaptively integrate the corresponding cues.

- Our adaptive cue integration scheme allows us to make inferences about the current status of the target object as well, through the fragments and the reliabilities associated with them. The dynamic reliability maps present the most informative fragments at each frame according to the current context, and for example might provide simultaneous information on the visible and occluded object sections.

- Tracking is realized by means of a particle filter-based framework [15, 16, 17], which enables the adaptive cue integration scheme to be derived, and which additionally provides an increased computational efficiency as exhaustive search on a combined vote map is no more required. It also allows us to easily include the scale information into the object state and gives a more natural way to estimate it.

- We present experimental results on challenging tracking sequences, which show that the adaptive formulation proposed in this paper results in a more accurate, efficient and robust tracking than the *FragTrack* and it competes and even outperforms the recently proposed boosting-based trackers [11, 12, 14] that use online appearance learning mechanisms.

The remainder of the paper is organized as follows: We begin with a brief summary of the *FragTrack* algorithm in Section 2. It is followed, in Section 3, by the description of the particle filter. In Section 4, we introduce our fragments-based tracking algorithm with an adaptive integration scheme, which is the main contribution of this paper. Following this, in Section 5, we present our experimental results and discuss different aspects of the proposed framework.

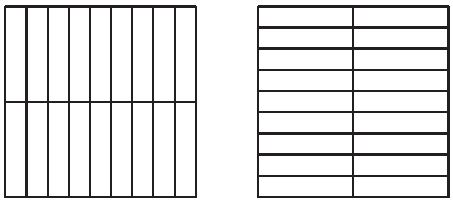

Figure 3: The patches utilized to represent the target object in the *FragTrack* algorithm [10].

2. *FragTrack* - Robust Fragments-Based Tracking [10]

In the *FragTrack* algorithm, tracking is treated as a sequential detection process. The target object is described by a template patch T, and in each image frame I, the detection is carried out by matching the template T to the image I. The output is a rectangular region enclosing the target object. As we discussed in the introduction, the novelty of the *FragTrack* comes from the arbitrary image patches and the corresponding robust estimation scheme for cue integration used in the template matching process.

Specifically, the tracking process is performed as follows: The template patch T is subdivided into multiple image patches $\{P_T\}$ with each of them describing a different section of the target object. The important point is that these multiple patches are chosen arbitrarily and are not based on any predetermined object model. The authors suggested to use the patch layouts presented in Figure 3 (overall 36 patches; 18 vertical, 18 horizontal).

For a hypothesized object position (x, y) in the current frame I, the patch $P_T = (dx, dy, w, h)$ defines a corresponding rectangular image patch $P_{I;(x,y)}$ with its center at $(x + dx, y + dy)$ and having a width w and a height h. Then, the dissimilarity between the patch P_T and the related image

patch $P_{I;(x,y)}$ is used to measure the degree of the compliance of the hypothesized object location (x, y) over the current frame, which is denoted by $V_{P_T}(x, y) = d(P_{I;(x,y)}, P_T)$. The *FragTrack* algorithm uses the intensity histograms populated from these patches and a metric d for comparing them to obtain this dissimilarity or voting score.

The sequential detection is carried out by first extracting vote maps for every patch P_T. That is, each object section described by the patch P_T votes on the possible target locations in the current frame. Due to computational efficiency, these vote maps are computed only for the pixels which are in a neighborhood of the previous estimate of the target position (the search neighborhood is defined with a fixed radius of 7 pixels from the previous target position). The real-time tracking capability is originated from the *Integral Histogram* data structure [18], which is used for rapidly determining the histograms required to extract these vote maps.

Once the vote maps are obtained, in the next step, the individual votes of the template patches are combined to obtain the joint tracking result. For that, the authors adopted the following robust estimation scheme for cue integration: For a hypothesized object position (x, y), they simply sort the dissimilarity scores $\{V_P(x, y)\}$ and choose the Q'th smallest one as the joint result $C(x, y)$.

The final tracking result is determined by considering all the joint dissimilarity scores $\{C(x, y)\}$ over the search window and looking for the image location with the minimal joint score. The parameter Q is related to the degree of potential occlusions that is expected to be encountered in the tracking sequence. It determines the maximal number of template patches that could

always be relied on for the measurements they provide. In [10], Q is taken as 25% of the total number of patches, i.e. *it is presumed that at least a quarter of the patches will be visible throughout the tracking sequence*. The authors also extended the proposed tracking algorithm so that a hypothesis about the scale of the target object is included into the computations. This is carried out by additionally enlarging and shrinking the template and accordingly the corresponding template patches by 10%, and selecting the location and scale with the lowest joint score.

It can be argued that the *FragTrack* algorithm uses a competitive approach in the integration step since the joint result is based on the dissimilarity value coming from a single template patch. Although this single dissimilarity value is determined by considering the observations coming from all the fragments, it does not reflect the dissimilarity values coming from the other fragments. Multiple image patches compete with each other to describe the target object and in the end only one of them wins this competition, although the winner does not always provide is not the one having the smallest score. The critical point here is that the winner patch might not always provide sufficiently reliable measurements. For instance, assume that only the fragments whose dissimilarity scores are in the first 20% quantile of all the scores provide reliable information regarding the target object. Then, even if the tracker has accurate observations coming from these fragments, it gives an inaccurate result since the result is determined by the fragment whose estimated dissimilarity score falls into the 25% quantile. In our work, as an alternative to this strategy, we propose to use an adaptive collaborative integration scheme in which the vote of each image patch directly contributes

to the joint voting result according to its reliability, and which will allow us to achieve a better tracking accuracy.

3. Particle Filter Framework

We realize tracking by means of a particle filter [15, 16, 17], and propose a fragments-based method within this framework. For the sake of completeness, we now recall in this section the main lines of particle filtering.

The idea behind the particle filter is to approximate the posterior $p(\mathbf{s}_k \mid \mathbf{z}_{1:k})$ by a weighted set of particles $\{\mathbf{s}_k^{(i)}, \pi_k^{(i)}\}_{i=1}^N$ as:

$$p(\mathbf{s}_k \mid \mathbf{z}_{1:k}) \approx \sum_{i=1}^{N} \pi_k^{(i)} \delta_{\mathbf{s}_k^{(i)}}(\mathbf{s}_k). \tag{1}$$

In Equation (1), $\mathbf{z}_{1:k}$ denotes the set of measurements up to and including the time step k, $\delta_{\mathbf{s}}$ is the Dirac delta mass centered on $\mathbf{s}$, and each particle represents a possible state $\mathbf{s}_k$ and a weight $\pi_k \in [0, 1]$ associated with it. The weights describe the confidence measures for the corresponding states.

Considering this approximation scheme, the recursive estimation process is performed as follows: First, a prediction phase generates new particles from the old particle set $\{\mathbf{s}_{k-1}^{(i)}, \pi_{k-1}^{(i)}\}_{i=1}^N$ by sampling them from a known proposal function, $\mathbf{s}_k^{(i)} \sim q(\mathbf{s}_k \mid \mathbf{s}_{0:k-1}^{(i)}, \mathbf{z}_{1:k})$. The simplest choice for the proposal function is $q(\cdot) = p(\mathbf{s}_k \mid \mathbf{s}_{k-1})$, which means to use the state evolution model itself for sampling. This approach is known as the CONDENSATION algorithm [16]. The second step of the recursive estimation includes a measurement phase which adjusts the weights of the new particles according to the new observations $\mathbf{z}_k$. This is simply performed by using the formula:

$$\pi_k^{(i)} \propto \pi_{k-1}^{(i)} \frac{p(\mathbf{z}_k \mid \mathbf{s}_k^{(i)}) p(\mathbf{s}_k^{(i)} \mid \mathbf{s}_{k-1}^{(i)})}{q(\mathbf{s}_k \mid \mathbf{s}_{0:k-1}^{(i)}, \mathbf{z}_{1:k})} \tag{2}$$

with $\sum_{i=1}^{N} \pi_k^{(i)} = 1$. One can additionally consider a resampling phase that removes the particles with low weights and accumulates the particles with high weights. Resampling is generally employed to avoid the degeneracy of the particles [15]. Finally, the decision about the tracking process for the current time step k can be obtained from the particle set by estimating the weighted average of the hypothesized states:

$$\hat{\mathbf{s}}_k = \sum_{i=1}^{N} \pi_k^{(i)} \mathbf{s}_k^{(i)}. \tag{3}$$

4. Proposed Tracking Algorithm

We propose a novel approach for tracking by integrating the multiple-fragments based object representation [10] with a particle-filter based adaptive multi-cue integration scheme. We describe the target object by a template patch denoted by T, and carry out tracking by searching for the image region in each frame I of the tracking sequence with appearance characteristics similar to the template T. This estimation process is done by splitting the target object into multiple arbitrary patches with each of them describing a different part of the target and accordingly providing the multi-cue information.

In our work, an object is parameterized by a state vector $\mathbf{s} = (x, y, s_x, s_y)$. Consequently, each particle describes an image region whose center is at (x, y), and which has a width and height of the size of the template T respectively scaled up with the scale factors s_x and s_y. Including the scale information into the state model makes the state dynamics handle the changes in the target's scale. Each particle represents a hypothesis that has additional scale information. The computational cost is not affected much since the

Input: The particle set at time step $k-1$, $\{\mathbf{s}_{k-1}^{(i)}, \pi_{k-1}^{(i)}\}_{i=1}^{N}$

1. **prediction**: generate the new particles by propagating the old ones (Equation 4), $\mathbf{s}_k^{(i)} \sim p(\mathbf{s}_k \mid \mathbf{s}_{k-1})$
2. **measurement**: evaluate the multiple cues and adjust the weights of the new particles (Equation 6), $\pi_k^{(i)} \propto p(\mathbf{z}_k \mid \mathbf{s}_k^{(i)})$
3. **resampling**: simulate $a_i \sim \{\pi_k^{(n)}\}_{n=1}^{N}$, and replace $\{\mathbf{s}_k^{(i)}, \pi_k^{(i)}\} \leftarrow \{\mathbf{s}_k^{(a_i)}, \frac{1}{N}\}$
4. **update**: adjust the cue reliabilities (Equation 7) and adapt the cue models (Equation 10)

Output: The particle set at time step k, $\{\mathbf{s}_k^{(i)}, \pi_k^{(i)}\}_{i=1}^{N}$

Figure 4: Outline of the Proposed Tracking Algorithm

computational complexity of the algorithm is mainly related to the number of particles used during tracking which provide the tested hypotheses.

We summarize the basic outline of our tracking algorithm in Figure 4. As it illustrates, we nearly follow the classic flow of a particle filter-based framework. The proposed tracker consists of *prediction*, *measurement*, *resampling* phases with an additional *update* step. Among these phases, the *measurement* and the *update* steps are the most important ones since we carry out the integration of the multiple cues and the consequent adaptation of the cue models in these steps.

4.1. Prediction Step

For predicting the new locations of the particles, we follow the CONDENSATION algorithm [16], and use the state evolution model itself for sampling. The state dynamics is modeled using the following formula:

$$p(\mathbf{s}_k \mid \mathbf{s}_{k-1}) \sim \mathcal{N}(\mathbf{s}_{k-1}, \Lambda) \tag{4}$$

with $\mathcal{N}(\mathbf{s}_{k-1}, \Lambda)$ denoting a Gaussian distribution with mean $\mathbf{s}_{k-1}$ and covariance matrix $\Lambda = diag(\sigma_x^2, \sigma_y^2, \sigma_{s_x}^2, \sigma_{s_y}^2)$. This model assumes mutually independent Gaussian random walks for each component in propagating the old particle set to obtain the new particle set.

4.2. Measurement Step

Once the new particles are generated by the prediction step, they provide the hypothesized object locations and scales for the current frame I. In the measurement step, the likelihoods of these hypotheses are determined by matching them to the template T. One of the critical components of our tracker is this matching process. For that, we split the template patch T into multiple smaller arbitrary image patches or fragments denoted by $\{P_T\}$ using the patch layouts given in Figure 3. Simply, a template patch $P_T = (dx, dy, w, h)$ consists of a displacement vector (dx, dy) from the template center, a width w and a height h. For a sample particle $\mathbf{s}^{(i)} = (x^i, y^i, s_x^i, s_y^i)$, it defines a corresponding image patch $P_{I;(x^i,y^i)}$ whose center is at $(x^i + s_x^i dx, y^i + s_y^i dy)$, and which has a width $s_x^i w$ and a height $s_y^i h$.

We represent each of these patches with an intensity histogram, and as in [10], use the Integral Histogram data structure [18] to estimate them in a fast way. Matching a template patch P_T and the corresponding image patch

$P_{I;(x^i,y^i)}$ is then carried out by estimating the similarity between them by comparing their histograms. For that purpose, we use the Bhattacharyya distance [19], which is defined as:

$$D(\mathbf{h}_1, \mathbf{h}_2) = \left(1 - \sum_{i=1}^{B} \sqrt{h_{i,1} h_{i,2}}\right)^{1/2}, \tag{5}$$

with B denoting the number of bins, and $h_{i,1}$ representing the i^{th} bin of histogram 1.

The similarity between P_T and $P_{I;(x^i,y^i)}$ provides us a likelihood value for the sample particle $\mathbf{s}^{(i)}$. The key point here is that we have a number of similarity scores. Each template patch P_T gives us some kind of confidence value regarding the target object's location. Thus, we need a mechanism to combine these similarities to reach a final joint likelihood. How to integrate them is important and has a direct effect on the tracking outcome. For example, during tracking some parts of the target object may not be visible due to external factors such as occlusions or changes in lighting conditions, which makes the similarity scores estimated from these parts unreliable.

Recall that the *FragTrack* algorithm performs the similar cue integration by using a robust estimation scheme. The integration is carried out based on an assumption on the maximal number of template patches that could always be relied on for the measurements they provide (Section 2), which can be seen as a drawback of the method. In our work, we do not make any assumption regarding the target object and the tracking sequence, and propose to use an adaptive, more collaborative multi-cue integration scheme in which each image patch contributes to the joint likelihood according to its

reliability as:

$$p(\mathbf{z} \mid \mathbf{s}) = \exp\left(-\frac{\sum\limits_{P_T \in \mathcal{P}} r_{P_T} d\left(P_{I;(x,y)}, P_T\right)^2}{2\sigma^2}\right) \tag{6}$$

with $\mathbf{s} = (x, y, s_x, s_y)$ denoting a possible state, σ being a scalar, $\mathcal{P} = \{P_T\}$ denoting the set of all template patches, $r_{P_T} \in [0, 1]$ indicating the reliability of the template patch P_T, $\sum_{P_T \in \mathcal{P}} r_{P_T} = 1$, and $d\left(P_{I;(x,y)}, P_T\right)$ being the Bhattacharyya distance between the histograms of $P_{I;(x,y)}$ and P_T, respectively.

Given a state hypothesis (i.e. a particle) $\mathbf{s_k^{(i)}}$ at time step k, the joint likelihood value in Equation 6 provides the probability of the observed outcome $\mathbf{z_k}$. This likelihood value is then used to define the new confidence measure or the new weight for the corresponding state as $\pi_k^{(i)} \propto p(\mathbf{z}_k \mid \mathbf{s}_k^{(i)})$. In Equation 6, the fragments having low reliability values have little contribution on the value of the joint likelihood. As a consequence, the framework assigns more accurate confidence values for the particles. Since the final tracking result is estimated by the weighted average of the hypothesized states (Equation 3), we would then expect that the fragments with low reliabilities are less taken into account in this step as well, which increases the robustness of the tracking system.

4.3. Update Step

The adaptive nature of our formulation stems from the fact that the reliabilities of the template patches denoted by $\{r_{P_T}\}$ are dynamically updated at each frame of the tracking sequence depending on the current visual context. This update process is performed based on the *Democratic Integration*

approach [5]. The goal of this adaption process is to suppress the fragments that are not in agreement with the joint result, and at the same time to give the fragments that are in line with the joint result a higher influence in the future. This is carried out by using the following dynamic update equation:

$$r_{P_T}^k = (1-\tau) r_{P_T}^{k-1} + \tau \frac{q_{P_T}}{\sum\limits_{P_T \in \mathcal{P}} q_{P_T}} \tag{7}$$

with q_{P_T} denoting a quality measure which quantifies the degree of agreement between the joint result and the result the cue (in our case the fragment P_T) individually suggests, and τ representing a parameter which controls the speed of the adaptation.

The democratic integration of cues and tracking with particle filters have been addressed first by Spengler and Schiele in [20]. In [20], the cue integration is, however, carried out by holding the cue weights or reliabilities constant throughout the tracking process. These weights are specified prior to tracking and are not allowed to change. In our work, we go one step further by including adaptive characteristics, which are key features of the democratic integration method.

In the context of particle filter framework, Shen, van den Hengel and Dick [21] proposed a novel quality measure for an individual cue, which can be estimated from the current particle set. For a cue, its quality is estimated by means of a sigmoid function of the Euclidean distance between the result suggested by that cue alone and the tracking result that is currently agreed upon by all the cues. As pointed out by Nickel and Stiefelhagen [1], this quality definition has some drawbacks. For example, if the target becomes idle, the resampling step of the particle filter framework could make

the particles spread around the true result. A cue providing unreliable observations during this time assigns uniform likelihoods to all the particles, and consequently the result obtained by that cue alone becomes consistent with the joint tracking result. This makes the quality of that cue be interpreted as high even with the cue is in fact not successful in locating the object of interest.

In our adaptive integration scheme, we make use of a quality measure in the line with the one proposed by Nickel and Stiefelhagen [1]. This quality measure depends on the proposition that a reliable cue (in our case a template fragment) should not only offer a result that is close to the joint hypothesis $\hat{\mathbf{s}}$ agreed upon all the cues, but also give a high likelihood value at the location provided by $\hat{\mathbf{s}}$.

To measure how well the outcome suggested by the hypothesis $\mathbf{s} = (x, y, s_x, s_y)$ is according to the fragment P_T, we define the following likelihood model:

$$p_{P_T}(\mathbf{z} \mid \mathbf{s}) = \exp\left(-\frac{d\left(P_{I;(x,y)}, P_T\right)^2}{2\sigma^2}\right), \tag{8}$$

where $d\left(P_{I;(x,y)}, P_T\right)$ denotes the Bhattacharyya distance between P_T and the corresponding image patch $P_{I;(x,y)}$. This model gives us a likelihood value that is estimated considering only the part of the target object pointed out by the template patch P_T. One can also use it to assign an additional weight to each particle $\mathbf{s}^{(i)}$, $\pi_{P_T}^{(i)} \propto p_{P_T}(\mathbf{z} \mid \mathbf{s}^{(i)})$ for $P_T \in \mathcal{P}$, providing a confidence value for the particle with respect to the individual template patch P_T.

The quality measure for a template patch P_T is then given by:

$$q_{P_T} = \frac{\sum\limits_{i=1...n} \frac{1}{n} \left|\mathbf{s}^{(i)} - \hat{\mathbf{s}}\right|^{\lambda}}{\sum\limits_{i=1...n} \pi_{P_T}^{(i)} \left|\mathbf{s}^{(i)} - \hat{\mathbf{s}}\right|^{\lambda}} p_{P_T}(\mathbf{z}|\hat{\mathbf{s}}) \tag{9}$$

with the exponent λ adjusting the distinctiveness of the quality measures. The lower the value of λ, the more similar the patches' qualities. The term $p_{P_T}(\mathbf{z} \mid \hat{\mathbf{s}})$ within the formula makes the estimated quality value depend on the success of the estimated joint result on describing the target by considering only the part pointed out by the template patch P_T. That is, a fragment has a high quality value only if that fragment suggests a high likelihood value for the estimated state. Additionally, the term $\sum\limits_{i=1...n} \pi_{P_T}^{(i)} \left|\mathbf{s}^{(i)} - \hat{\mathbf{s}}\right|$ in the denominator quantifies how well the particle set forms a group around the the joint result. The term $\sum\limits_{i=1...n} \frac{1}{n} \left|\mathbf{s}^{(i)} - \hat{\mathbf{s}}\right|$ which considers uniform weights is for making the quality value independent of the actual position of the particles.

The update step of our formulation also involves the adaptation of the cue models. Instead of fixing the template T, and accordingly the set of template fragments $\{P_T\}$ extracted from T at the first frame, and then using them during the whole sequence without any update, we follow an additional re-calibration strategy. We adjust all of the template patch models considering the current joint result by using the following dynamic adaptation equation defined for a single patch:

$$P_T^k = (1 - \tau_c) P_T^{k-1} + \tau_c \hat{P}_T \tag{10}$$

where $\hat{P}_T$ is the template patch extracted from the image region the joint hypothesis $\hat{\mathbf{s}}$ provides, and τ_c denotes a time constant. We set the parameter

Table 2: Parameters for the tracking experiments

# of particles	$N = 500$
# of bins	$B = 16$
standard deviation of likelihoods	$\sigma = 0.1$
exponent for cue reliability	$\lambda = 4$
time constant for cue reliabilities	$\tau = 0.1$
time constant for cue adaptations	$\tau_c = 0.01$
standard deviations of dynamics	$\sigma_x = \sigma_y = 5$
	$\sigma_{s_x} = \sigma_{s_y} = 2.5 \times 10^{-3}$

τ_c to a small value (as compared to that of τ) since we do not want the appearance model of the target object to be changed so quickly.

4.4. Implementation Details

We have implemented the proposed algorithm in MATLAB on a MacBook with a 2.2 GHz Intel Core2 Duo processor. In all the experiments, we used the fixed set of parameters given in Table 2 unless stated otherwise. Our implementation additionally includes some MEX C++ subroutines, which helps us to achieve a real-time tracking performance. For a video sequence containing 320×240 image frames, our tracker runs at approximately 10 frames per second.

5. Experiments

In this section, we present our experimental results. We perform three groups of experiments to illustrate the performance and the effectiveness of

the proposed multi-fragments based tracking algorithm. In the first set of experiments, we demonstrate the basic features of the proposed tracker on illustrative tracking sequences. The second set of experiments is about the qualitative and the quantitative analysis of the proposed method in terms of tracking accuracy. Following that, in the final group of experiments, we compare the running times of the *FragTrack* and the proposed algorithm. The videos showing the results of these experiments are provided as supplementary material. In these experiments, the proposed tracker and the other trackers in comparison are initialized by manually marking the image region surrounding the target object in the first frame of the sequence under consideration. At each frame, we use the weighted average of the hypothesized states (Equation 3) to represent the tracking result of our algorithm.

5.1. Illustrative results

In this section, we focus on two key points: to show how the fragments are working as complementary cues and to show how the adaptive multi-cue integration works. For that, we begin with a video sequence of tracking a woman's face. The main difficulty with this sequence is that the face is often occluded by a magazine throughout the sequence. To illustrate the key characteristics of the proposed tracker in a more clear way, in our first experiment, we divide the template patch into 4 fragments and run the tracker accordingly. In Figure 5, we present a few frames from the sequence together with a plot of reliability values of each fragment along the sequence. For the sample frames, the reliabilities are shown in green tones where the higher values are shown in high intensity.

As can be seen, the reliability of a fragment decreases when it is hidden.

For instance, fragments #2 and #4 correspond to an occluded part of the face around frame 300, and lower values of reliability are obtained for these fragments. On the contrary, fragments #1 and #3 correspond to visible parts of the face at that time, and their reliabilities are then much higher. This is because the used dynamics makes these adjustments in the fragments' reliabilities considering the changes in the current visual context. One should keep in mind that the current reliability value of a fragment depends not only on the current observation but also on the observation history of that fragment. For example, around frame 773, fragments #3 and #4 have reliabilities smaller than that of fragment #1 even if they are all visible. The reason of this is that fragments #3 and #4 were previously considered unreliable seeing them occluded in some of the preceding frames. When they become visible again, the dynamics of the tracker increases their reliabilities (see the related plot of reliabilities in Figure 5). Around frame 800, all the fragments are visible and all have nearly equal reliabilities.

Next, we employ the layout introduced in Figure 3 to represent the target object, and run the tracker accordingly. Here, the template match is subdivided into 36 fragments (18 vertical and 18 horizontal), which provides a more detailed description of the object. We present sample frames in Figure 6 where the superimposed reliability maps are computed by adding the individual reliabilities of the vertical and the horizontal fragments. In all our tests, we observed that very low reliability values always correspond to occluded areas. Moderately low values can also have other causes (strong change in appearance, lack of information, etc.), and it is also an interesting feature of our approach that in such cases the corresponding fragments have

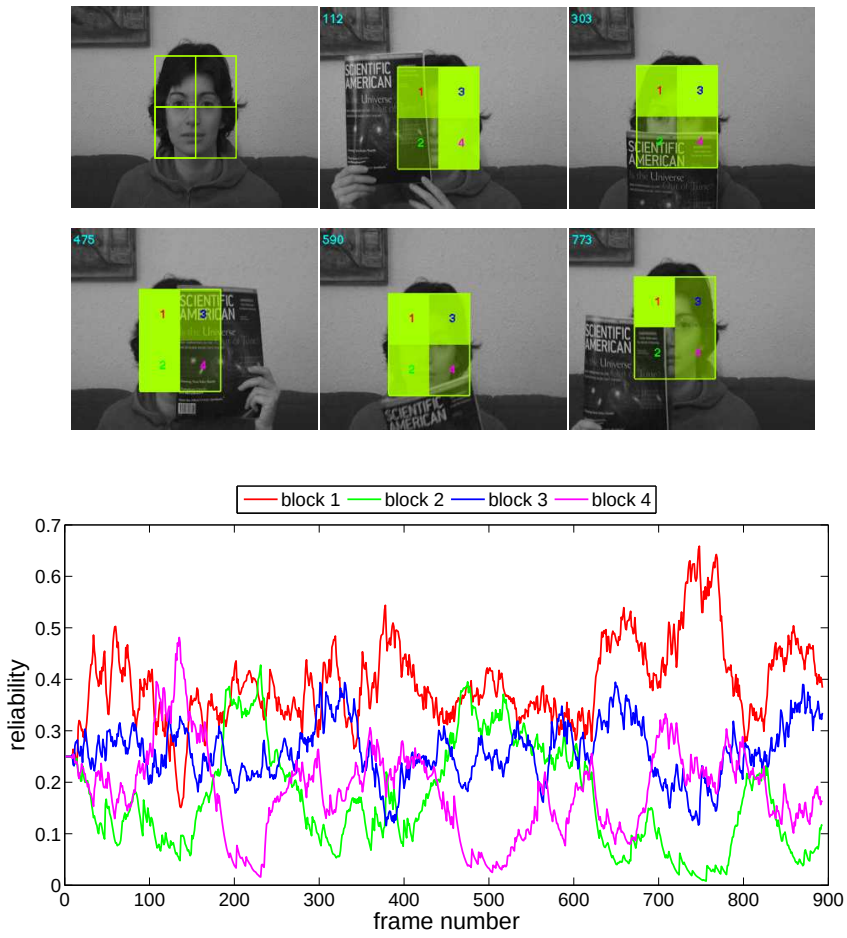

Figure 5: Sample frames with the superimposed reliabilities (using 4 fragments) and the plot of reliability values of each fragment along the sequence. Observe that the reliabilities of the visible fragments are generally higher than those of the occluded fragments.

a moderate influence on the tracking. It is also important to note that the *FragTrack* lacks this feature, and that it does not permit similar kind of inferences to be drawn from the tracking results. Among the boosting-based tracking frameworks mentioned in the introduction, only the one proposed by Woodley, Stenger and Cipolla [13] has the ability to detect occluded parts of

Figure 6: Sample frames with the superimposed reliability maps (using 36 fragments; 18 vertical and 18 horizontal). Our method copes with the partial occlusions, simultaneously providing information regarding which sections of the target object are occluded or not important for tracking.

the tracked object, but it achieves this by additionally learning a generative object model from a set of relatively large number of sample frames.

As a third experiment, we use a video sequence showing a walking woman. As illustrated in Figure 7, the woman goes behind cars, becomes occluded, and her pose undergoes some changes along the sequence. Here, we compare two contrasting strategies: setting the reliabilities of the fragments to a fixed value and adapting the reliabilities of the fragments depending on the context. For both cases, we used $N = 2000$ particles and a variable scale model. For the tracker that employs fixed reliabilities, all the fragments have the same influence, such lowering the role of the really relevant ones. For example, at some points, only a small part of the body remains visible, corresponding to only few patches in the fragment decomposition. As a result, the tracker does not accurately track the target object. On the other hand, the

proposed tracker never loses the track of the target by adaptively modifying the fragment reliabilities and consequently by adapting itself to the changes in the scene. It is interesting to note that the tracking is successful even only few fragments have a good reliability. For example, when the bottom part of the woman is occluded by the car, strong reliabilities are located on the relevant visible part of her body, in particular in her shirt, which guarantees that they have the main influence during the tracking. This experiment also illustrates that using adaptive reliabilities allows the tracker to focus on key parts of the target object. That is, a low reliability value does not always mean that the corresponding part is occluded. It actually means that the corresponding part has not a major role in tracking the object of interest.

The contribution of adaptive multi-cue integration to tracking is additionally evaluated by using the ground truth data of the sequence. The error plots are reported in Figure 8. The error is small and stable when reliabilities are adapted from the context. On the opposite, when reliabilities are fixed and equal for all the blocks, the tracking accuracy drastically decreases. For this sequence, we get an average center location error of 7.02 pixels with adaptive reliabilities, and of 98.54 pixels with fixed reliabilities.

5.2. Qualitative and quantitative analysis

We evaluate our tracking algorithm on the video sequences which have been used in a recent tracking paper [14], and for which the ground truth information is available for every 5 frames. These video sequences exhibit a wide variety of challenges including changes in the pose, scale and orientation of the target object, varying illumination conditions, and partial occlusions. We compare our results to those of the *FragTrack* [10] and

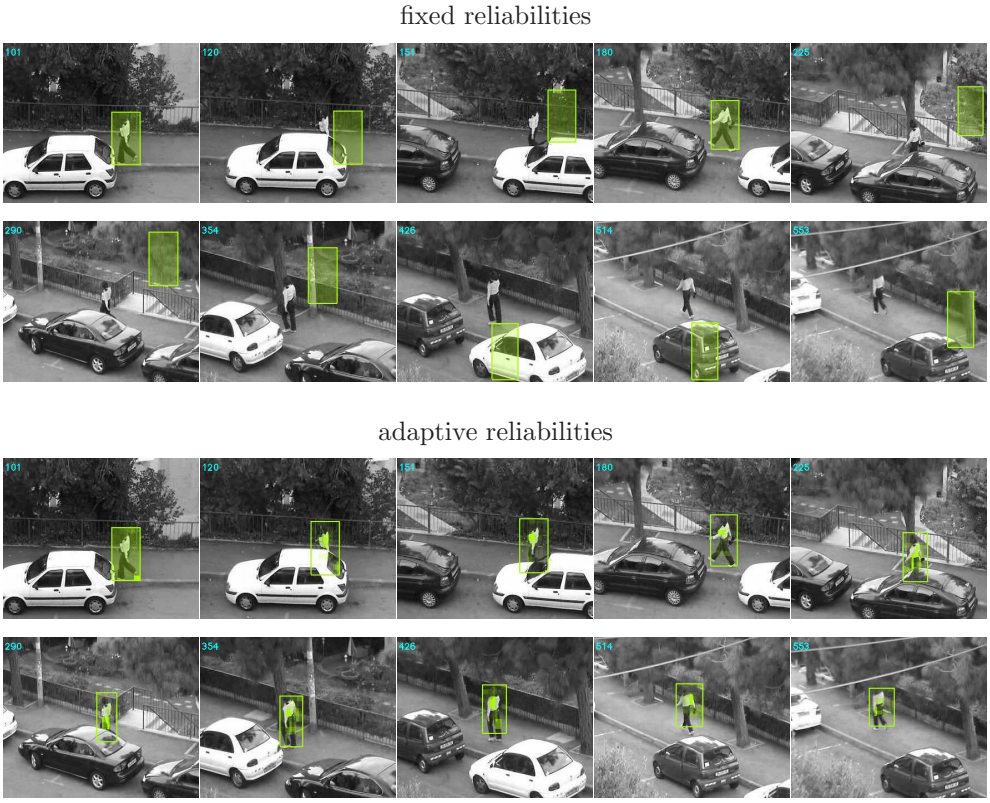

Figure 7: Sample tracking results with the superimposed reliability maps. First two lines, reliabilities are fixed; last two lines, reliabilities are adapted depending on the context. From the given results, it can be easily seen that adaptive multi-cue integration boosts the accuracy of the tracking process.

the boosting-based tracking frameworks[2] which respectively employ *Online-AdaBoost (OAB)* [11], *SemiBoost* [12] and *Multiple Instance Learning (MIL-Track)* [14] algorithms. For the *FragTrack* algorithm, we fixed the search radius to 7 pixels from the previous position of the target object, the number

[2]We used the results presented in [14] and the data provided by the authors at `http://vision.ucsd.edu/~bbabenko/project_miltrack.shtml`.

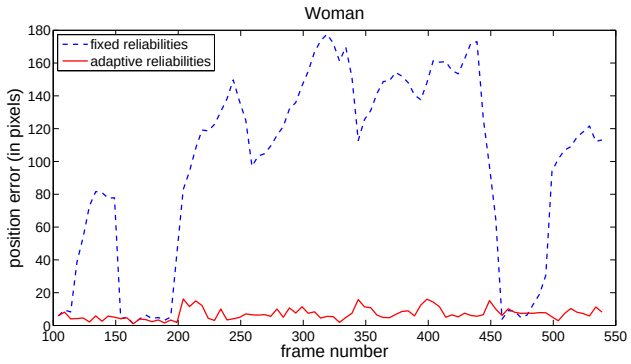

Figure 8: Error plots for the walking woman sequence: in blue, all the blocks have a fixed reliability value, and in red, the reliabilities are adapted during the sequence, depending on the context.

of bins of the histograms to 16, and used 25% of the number of fragments for the value of Q in the robust integration scheme as suggested by the authors. We tested our method considering both a variable scale model and a fixed single scale model (with $\sigma_{s_x} = \sigma_{s_y} = 0$) for the target object. Note that the tested *FragTrack* and the learning-based trackers offer single scale solutions. As pointed out in [14], the Haar-like features used in the learning-based trackers, however, provide an object representation that is fairly insensitive to scale changes. If one wants to explicitly incorporate multi-scale information into these trackers, the only way is to enlarge and shrink the template box by a pre-determined fixed scale, e.g. by 10%, when locating the target object in a new frame, and accordingly to run several trackers at once with increased computational load.

For quantitative analysis, we ran our tracker 5 times and took the average for each video sequence since our formulation involves some randomness. Table 3 summarizes these results. Figures 9-11 show the initial templates

Table 3: Center location errors (in pixels) w.r.t. manually marked ground truth (the best and the second best performances are indicated in red and blue, respectively)

Video Sequence	OAB1	OAB5	SemiBoost	MILTrack	FragTrack	Our Method (single scale)	Our Method (variable scale)
David Indoor	49	72	59	**23**	69	46	**15**
Sylvester	25	79	22	**11**	20	**14**	**11**
Occluded Face	44	105	41	27	**5**	**4**	**5**
Occluded Face 2	21	93	43	20	16	**13**	**14**
Girl	48	68	52	32	24	**14**	**16**
Tiger 1	35	58	46	**15**	39	**33**	34
Tiger 2	34	**33**	53	**17**	40	42	36
Coke Can	**25**	57	85	**21**	65	48	64

and some sample tracking results for five of the sequences. We also provide the error plots for each sequence in Figure 12. For the sake of clarity, in these figures, we only present the outcomes of our method and the *FragTrack* along with those of the *MILTrack* since the *MILTrack* produced the best results among the other boosting-based tracking frameworks. Apart from these sequences, we also provide our tracking outcomes for three other sequences `Dudek`[3], `Dog`[4] and `Girl with Many Eyes` of which sample snapshots are given in Figures 13 and 14. These sequences, however, have not been evaluated against any ground truth. We only present qualitative comparison against the *FragTrack* algorithm.

It can be seen from these results that the proposed algorithm outperforms the *FragTrack* algorithm in terms of tracking accuracy. The reason for this mainly stems from our adaptive cue integration scheme. It removes

[3]The sequence is available at `http://www.cs.toronto.edu/vis/projects/dudekfaceSequence.html`.

[4]The sequence is available at `http://www.cs.toronto.edu/~dross/ivt`.

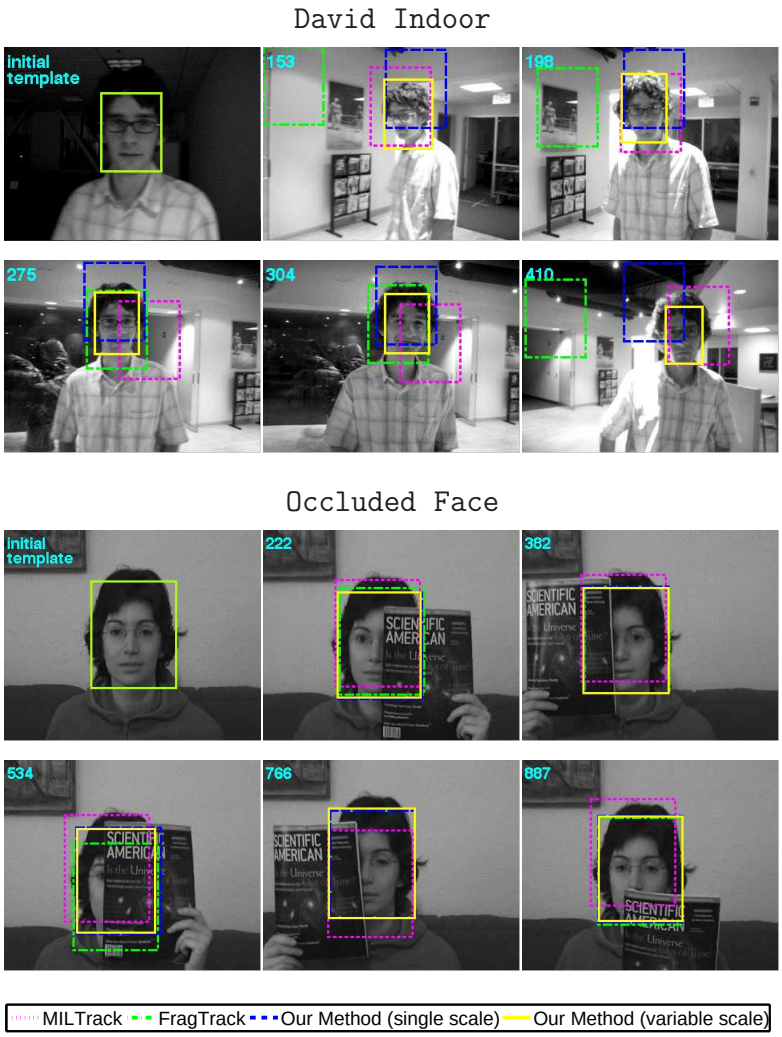

Figure 9: In the `David Indoor` sequence, our method with the variable scale model provides the best performance. It deals with varying illumination conditions, and pose and scale changes. In the `Occluded Face` sequence, our method and the *FragTrack* algorithm perform well and give nearly similar results. However, between the frames 515 and 580 where the occlusion is more severe than the others, our method performs better than the *FragTrack* (see the related error plot in Figure 12).

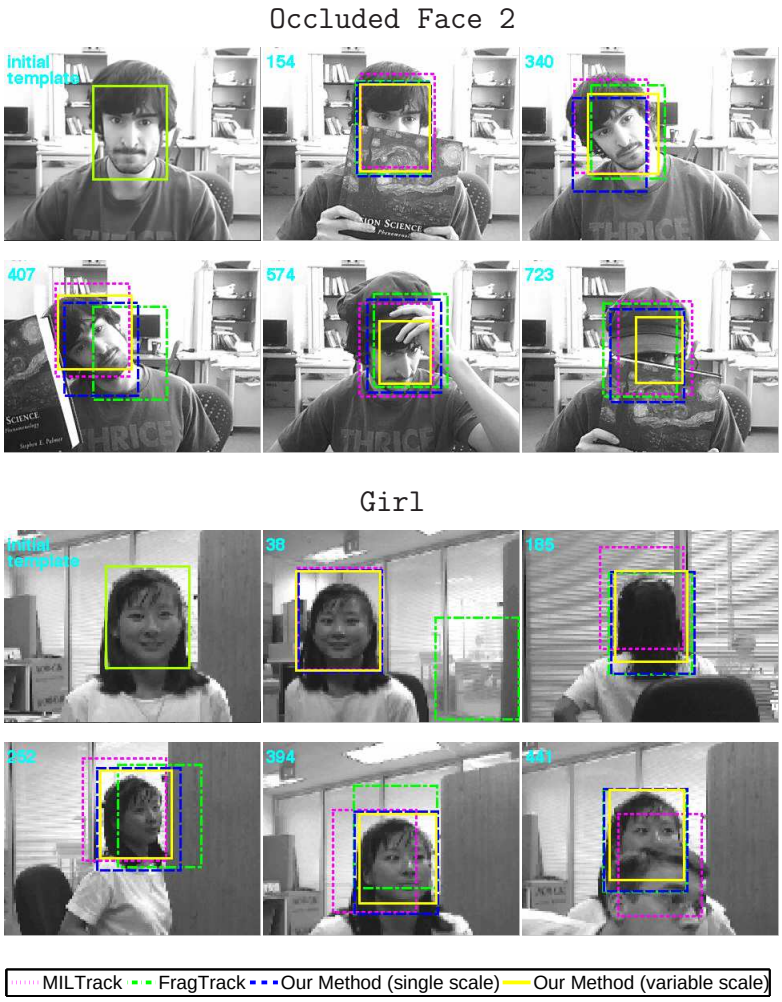

Figure 10: The `Occluded Face 2` and the `Girl` sequences show the target objects in occlusion and in different poses. For these sequences, our method generally gives better results than the others. In the `Girl` sequence, our method with the variable scale model handles the changes in the target's scale as well.

the assumption that the *FragTrack* makes on the degree of potential occlusions, and replaces its competitive approach to cue integration with a more cooperative strategy. At each frame, the target is determined by using all

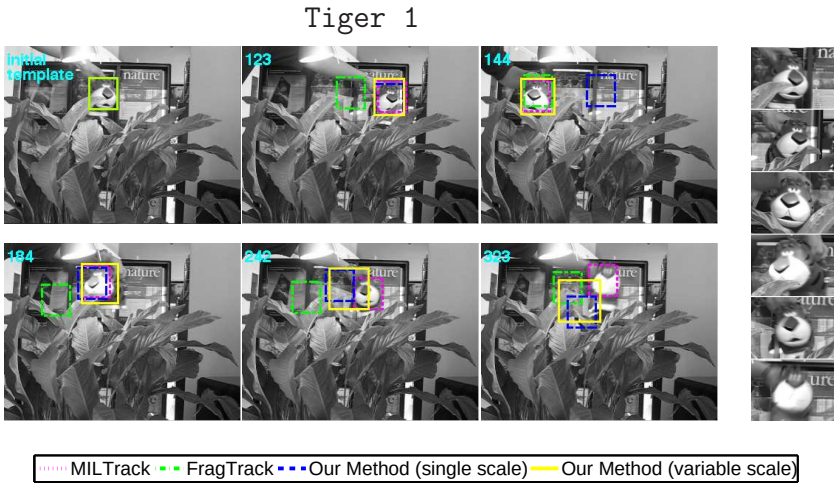

Figure 11: In the `Tiger 1` sequence, the target object moves very fast and its appearance changes quickly. To highlight these cases, we additionally give enlarged target regions in the right-hand side of the sample tracking results. For this sequence, the *MILTrack* performs the best since it updates the appearance model at each frame using an online learning mechanism. Our proposed tracker sometimes loses the track of the target object.

the template patches, but additionally considering their reliabilities, rather than by just using a single patch which itself may provide poor or inaccurate measurements. Moreover, our variable scale model copes well with the scale changes in the target object during the tracking process. However, sometimes with a variable scale model, there may be a tendency for the target to shrink under partial occlusions (see the `Occluded Face 2` seq. in Figure 10). This tendency is a result of the partial vs. full explanation dilemma [10] when choosing the scale, and is an important issue both for the *FragTrack* and our method.

Boosting-based trackers are proved to be very robust against severe appearance changes since they learn a new object model at each frame by using

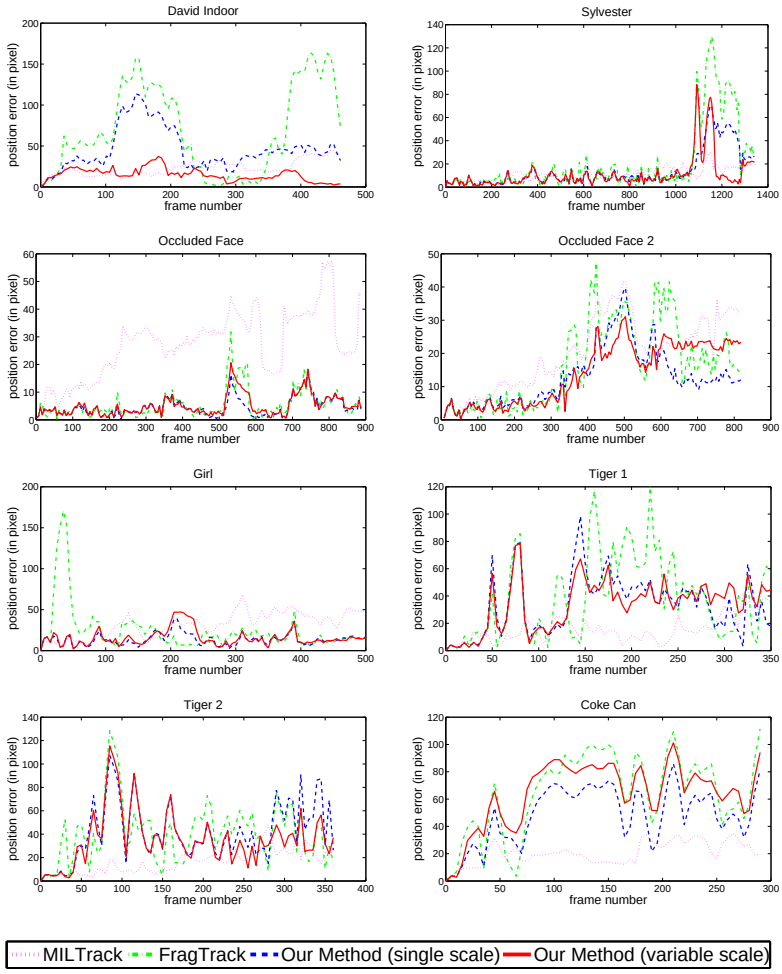

Figure 12: Error plots for the video sequences used in the quantitative analysis.

online appearance learning mechanisms. However, they yield, in many cases, poor tracking results when the target object becomes heavily occluded by the other objects in the scene, and this is clearly noticeable in our experiments. Our method and partly the *FragTrack* algorithm cope with these kinds of occlusions better than most of the boosted trackers. As a result, for most of the video sequences, our algorithm provides the best results. How-

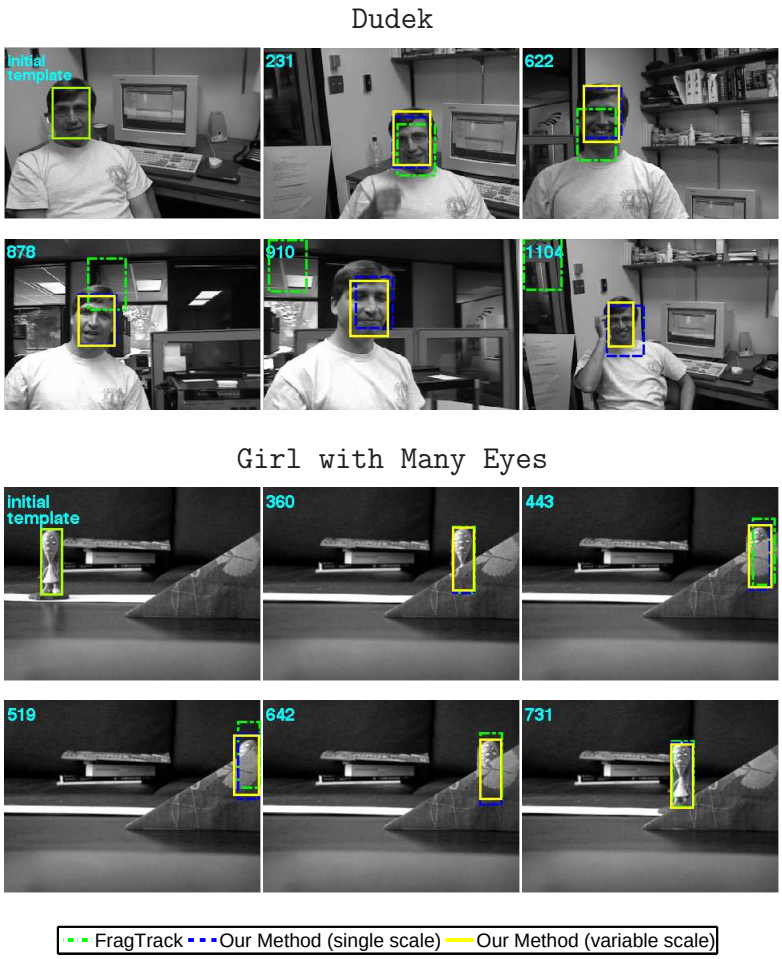

Figure 13: In the `Dudek` sequence, the target person undergoes significant pose, expression and appearance changes. Our method successfully deals with these changes during tracking. The *FragTrack* algorithm, on the other hand, gives poor results. The `Girl with Many Eyes` sequence shows the target object in severe occlusion. From the given results, we see that the proposed algorithm outperforms the *FragTrack* algorithm. Accurate tracking of the target under a high degree of potential occlusion is achieved as a result of our adaptive formulation for the multi-cue integration.

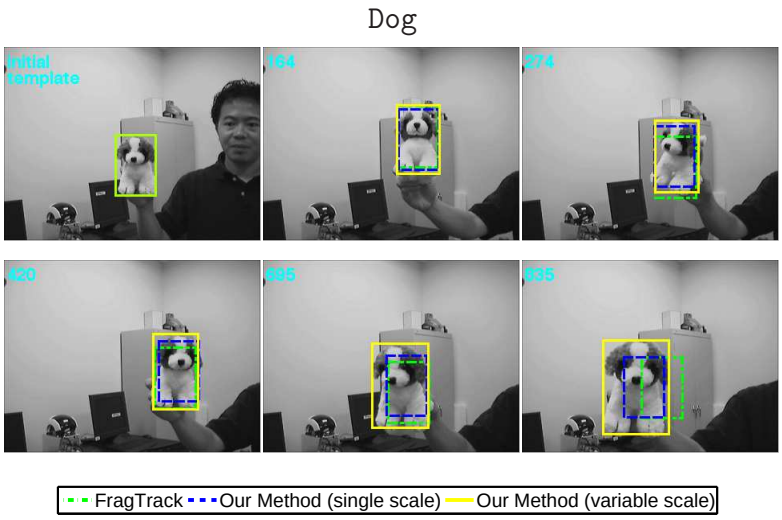

Figure 14: In the `Dog` sequence, the tracked object experiences large scale variation and pose changes. Thus, the approaches that consider fixed single scale models, including ours, yield inaccurate outcomes. However, our approach with a variable scale model is able to keep track of the target well.

ever, the *MILTrack* and the *OAB* tracker give better results than ours for the video sequences `Tiger 1`, `Tiger 2`, `Coke Can`. In these sequences, the appearances of the target objects vary too much due to either fast motion activity or changes in the lightning conditions or both (e.g., see the sample close up shots of the target object given in Figure 11). Consequently, our (fragments-based) object model which depends on a single view does not suffice to produce meaningful results and our tracker sometimes loses the track of the target. For these video sequences and the other sequences with similar characteristics, we believe that the tracking can be improved by using multiple views of the target acquired prior to the tracking [22]. The advantage of our formulation is its adaptive nature which lets us easily combine different

target views, but generally with a loss of computational efficiency. It would be interesting to focus on developing more efficient solutions to this problem in future work.

5.3. Runtime analysis

Finally, in our last experiment, we investigate the runtime of the *FragTrack* algorithm with respect to the search radius r, and of the proposed method with respect to the standard deviations σ_x and σ_y. These parameters are critical since they respectively determine the hypothesis space of the methods, i.e. the region in the current frame where the target object is searched for. In Table 4, we summarize the runtime performances of both methods for different values of r, and σ_x and σ_y for the `Cartoon` sequence[5] where the target object moves very rapidly. We also provide sample tracking results in Figure 15 (for the sake of clarity we only show the results obtained by using the smallest and the largest values). From these results, we see that for small values of r, and σ_x and σ_y, both the *FragTrack* and our tracker could not cope with the rapid movement of the target and lose its track. Increasing their values improves the tracking accuracy individually, however, for the *FragTrack*, this means a loss in the runtime performance as increasing r increases the number of pixels examined during tracking. Increasing σ_x and σ_y, on the other hand, does not change the runtime performances since the runtime complexity of our method is not related to σ_x and σ_y, but to the number of particles N.

[5]The sequence is from the authors of [8], and is available at `http://www.cise.ufl.edu/~smshahed/tracking.htm`

Table 4: Approximate runtimes (in frames per second) for the `Cartoon` sequence.

Tracking Method	**Runtime (fps)**
FragTrack ($r = 7$)	7
FragTrack ($r = 21$)	4
FragTrack ($r = 35$)	2
Our Method ($\sigma_x = \sigma_y = 5$)	10
Our Method ($\sigma_x = \sigma_y = 20$)	10
Our Method ($\sigma_x = \sigma_y = 35$)	10

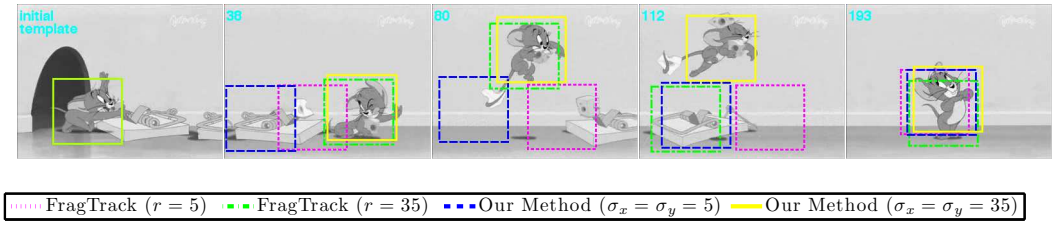

Figure 15: Sample frames with the tracking results superimposed. The trackers lose the track of the target object when small values are used for r, and σ_x and σ_y. When their values are increased, the trackers are able to follow the target and cope better with the jumps between the frames. However, for the *FragTrack* algorithm increasing the value of r decreases the runtime performance (see Table 4).

6. Summary and Discussion

In this paper, we have presented a new approach for model-free tracking. It combines the arbitrary-fragments based object representation [10] and the concept of adaptive multi-cue integration [5]. Our approach associates the image fragments describing different parts of the target object with some reliability values, which are dynamically adjusted during tracking. In this way, the vote of each fragment contributes to the joint tracking result according to its reliability, allowing to achieve the needed precise cue integration. We have

demonstrated the potential and the effectiveness of the proposed approach on various challenging video sequences with different tracking scenarios. As our experimental validation reveals, the proposed approach works generally better than the *FragTrack* tracker and the other tested learning-based trackers for the sequences with significant amount of occlusions. When there is smooth changes either in the target's appearance or its movements, i.e. when they do not change very abruptly, our method can adapt itself accordingly by changing the fragments' reliabilities and their internal parameters, yielding more accurate and more robust outcomes.

Another important feature of our adaptive tracker algorithm is the ability to estimate dynamic reliability maps. Through the fragments and the reliabilities associated with them, at each frame, we can form informative maps for the object of interest by adding the individual reliabilities of the vertical and the horizontal fragments used in our object representation. With an example, we present that the tracking results superimposed with the corresponding reliability maps provide simultaneous information on the distinctive and occluded object sections (see Figure 6).

An important issue for model-free tracking is the so-called *drifting* problem. During tracking, a model-free tracker should be able to adapt itself to the changes in the target's appearance while preventing drifting to the background or focusing on the occluding elements in the scene. Although we do not directly address this problem in our work, we obtained fairly good results as compared to works that specifically learn object-specific detectors via object/background classifiers [11, 12, 13, 14]. As a future work, it would be interesting to address this problem within our framework by considering

the fragment reliabilities in updating the appearance model of the target object or by combining different views of the target object during tracking. Similarly, we think that one can use these dynamic reliabilities to find a way to guide the scale of the target and to resolve the *partial vs. full explanation* dilemma that we mentioned in our experimental validation section.

[1] K. Nickel, R. Stiefelhagen, Dynamic integration of generalized cues for person tracking, in: Proc. Eur. Conf. Computer Vision (ECCV), 2008, pp. 514–526.

[2] S. Birchfield, Elliptical head tracking using intensity gradients and color histograms, in: Proc. Conf. Computer Vision and Pattern Recognition (CVPR), 1998, pp. 232–237.

[3] Y. Wu, T. S. Huang, A co-inference approach to robust visual tracking, in: Proc. Int. Conf. Computer Vision (ICCV), 2001, pp. 26–33.

[4] P. Pérez, J. Vermaak, A. Blake, Data fusion for visual tracking with particles, Proceedings of the IEEE 92 (3) (2004) 495–513.

[5] J. Triesch, C. von der Malsburg, Democratic integration: Self-organized integration of adaptive cues, Neural Computation 13 (9) (2001) 2049–2074.

[6] E. Maggio, F. Smeraldi, A. Cavallaro, Combining colour and orientation for adaptive particle filter-based tracking, in: Proc. British Machine Vision Conf. (BMVC), 2005, pp. 659–668.

[7] P. Brasnett, L. Mihaylova, D. Bull, N. Canagarajah, Sequential Monte

Carlo tracking by fusing multiple cues in video sequences, Image Vision Comput. 25 (8) (2007) 1217–1227.

[8] S. M. S. Nejhum, J. Ho, M.-H. Yang, Visual tracking with histograms and articulating blocks, in: Proc. Conf. Computer Vision and Pattern Recognition (CVPR), 2008, pp. 1–8.

[9] P. Chockalingam, N. Pradeep, S. Birchfield, Adaptive fragments-based tracking of non-rigid objects using level sets, in: Proc. Int. Conf. Computer Vision (ICCV), 2009, pp. 1530–1537.

[10] A. Adam, E. Rivlin, I. Shimshoni, Robust fragments-based tracking using the integral histogram, in: Proc. Conf. Computer Vision and Pattern Recognition (CVPR), 2006, pp. 798–805.

[11] H. Grabner, M. Grabner, H. Bischof, Real-time tracking via on-line boosting., in: Proc. British Machine Vision Conf. (BMVC), 2006, pp. 47–56.

[12] H. Grabner, C. Leistner, H. Bischof., Semi-supervised on-line boosting for robust tracking., in: Proc. Eur. Conf. Computer Vision (ECCV), 2008, pp. 234–247.

[13] T. Woodley, B. Stenger, R. Cipolla, Tracking using online feature selection and a local generative model, in: Proc. British Machine Vision Conf. (BMVC), 2007.

[14] B. Babenko, M.-H. Yang, S. Belongie, Visual tracking with online multiple instance learning, in: Proc. Conf. Computer Vision and Pattern Recognition (CVPR), 2009, pp. 983–990.

[15] N. Gordon, D. Salmond, A. Smith, Novel approach to nonlinear/non-Gaussian Bayesian state estimation, IEE Proc. F 140 (2) (1993) 107–113.

[16] M. Isard, A. Blake, CONDENSATION-conditional density propagation for visual tracking, Int. J. Comput. Vis. 29 (1) (1998) 5–28.

[17] A. Doucet, A. M. Johansen, A tutorial on particle filtering and smoothing: Fifteen years later, Tech. rep., Department of Statistics, University of British Columbia (2008).

[18] F. Porikli, Integral histogram: A fast way to extract histograms in Cartesian spaces, in: Proc. Conf. Computer Vision and Pattern Recognition (CVPR), Vol. 1, 2005, pp. 829–836.

[19] A. Bhattacharyya, On a measure of divergence between two statistical populations defined by their probability distributions, Bull. Calcutta Math. Soc. 35 (1943) 99–109.

[20] M. Spengler, B. Schiele, Towards robust multi-cue integration for visual tracking, Mach. Vision Appl. 14 (1) (2003) 50–58.

[21] C. Shen, A. van den Hengel, A. Dick, Probabilistic multiple cue integration for particle filter based tracking, in: Proc. Int. Conf. Dig. Image Comp. Tech. Appl. (DICTA), 2003, pp. 399–408.

[22] I. Leichter, M. Lindenbaum, E. Rivlin, Mean shift tracking with multiple reference color histograms, Comput. Vis. Image Understand. 114 (3) (2010) 400–408.