Computer Vision and Image Understanding
journal homepage: www.elsevier.com

Occlusion aware particle filter tracker to handle complex and persistent occlusions

Kourosh Meshgi[a,**], Shin-ichi Maeda[a], Shigeyuki Oba[a], Henrik Skibbe[a], Yu-zhe Li[a], Shin Ishii[a]

[a] *Graduate School of Informatics, Kyoto University, Yoshidahonmachi, Sakyo Ward, Kyoto, 606–8501 JAPAN*

ABSTRACT

Although appearance-based trackers have been greatly improved in the last decade, they are still struggling with some challenges that are not fully resolved. Of these challenges, occlusions, which can be long lasting and of wide variety, are often left aside or partly addressed, due to the difficulty in its general treatments. To address this problem, in this study we propose an occlusion aware particle filter framework that employs a probabilistic model with a latent variable representing an occlusion flag. The proposed framework prevents losing the target by prediction of emerging occlusions, updates the target template by shifting relevant information, expands the search area for occluded target, and grants quick recovery of the target after occlusion. Furthermore the algorithm employs multiple features from color and depth domains to achieve robustness against illumination changes and clutter, so that the probabilistic framework accommodates the fusion of those features. Applied to Princeton RGBD Tracking dataset, the performance of our method with different sets of features was compared with those by the state-of-the-art trackers. The results revealed that our method outperformed the existing RGB and RGBD trackers by successfully dealing with different types of occlusions.

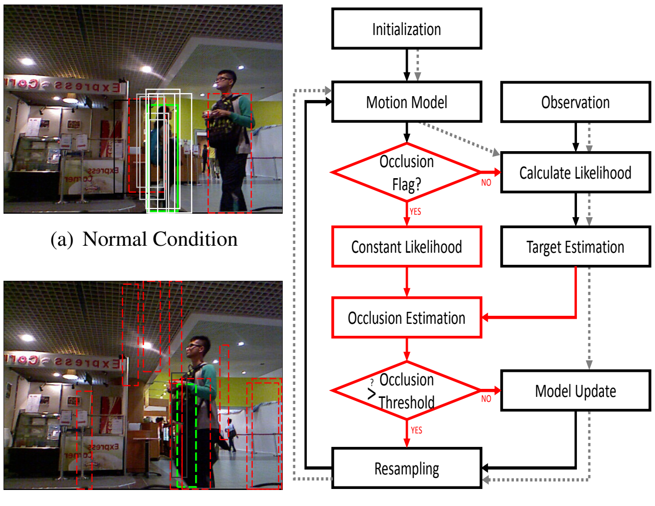
1. Introduction

Object tracking is of an emerging demand in various daily-life applications ranging from human-computer interface, to human behavior analysis, video communication/compression, virtual/augmented reality and surveillance. When applied to video sequences in real-life situations, trackers should cope with numerous difficulties often caused by similarities between objects (same color, etc), illumination changes, motion blur, non-rigid deformations and shape changes, moving camera, and most importantly different occlusions with variety of lengths and extents. Occlusions make a part or the whole target object (the object to be tracked) invisible to the sensor, where the duration of such invisibility is often unknown beforehand. They may be cast upon the target by another moving object, static background objects, or the target object itself (Vezzani et al., 2011). Persistent and complex occlusions are among the most challenging forms of occlusions. The former pertains to the full occlusion which may last for several frames whereas in the later occlusion causes drastic changes in some of the key characteristics of the target object (e.g. appearance, orientation, motion direction, size, and distance from camera).

In this study we propose a tracker which detects emergent occlusions, deal with difficult occlusion scenarios, and quickly perform target recovery after occlusion. This novel method builds upon particle filter trackers (PFTs) and significantly improves its resilience against various kind of occlusions (including persistent and complex ones). The idea behind the success of this tracker is to switch its behavior to a more effective one in the case of occlusion, efficiently and flexibly.

Particle filter, a recursive form of Bayesian filters, has been used for analyzing image series, with a prominent advantage in its applicability to non-linear and non-Gaussian scenarios, which is the case in most real-world videos. The stochastic sampling of posterior possibilities provided particle filter trackers with various extensions. Of these, Condensation PFT (Isard and Blake, 1998) was initially developed to track objects in cluttered environments using edge information, but later extended to use kernels (Nummiaro et al., 2003) and fuse multiple cues (Perez et al., 2004). Regular PFTs such as the ones in Nummiaro et al. (2003) and Brasnett et al. (2007) can handle partial and temporal occlusions, benefiting from many scattered par-

**Corresponding author: Tel.: +81-75-753-4809;
e-mail: meshgi-k@sys.i.kyoto-u.ac.jp (Kourosh Meshgi)
URL: http://ishiilab.jp/member/meshgi-k/oapft.html (Kourosh Meshgi)



(a) Normal Condition

(b) Occlusion Case

(c) OAPFT vs. PFT flowchart

Figure 1. The dynamics of occlusion-aware particle filter (OAPFT) in normal condition (a) and occluded condition (b). Brighter boxes are more similar to the target and red boxes are marked as occluded. Green box is the estimated target location. The comparative flowchart of PFT and the proposed method is illustrated in panel (c).

ticles, but their accuracy degrades especially when there is a change in target's trajectory during occlusions. However, the tracker is unable to capture target deformations and illumination changes when the target template is fixed throughout the tracking. Hence, like many other trackers, PFT should update its template with new observation (Perez et al., 2004). Nevertheless for longer occlusions, occluded target cannot be recovered since the model is corrupted by non-target observations –occluder or background– during occlusion, a problem called "model drift" (Pan and Hu, 2007). This argument also stands for dominant or persistent partial occlusions. Additionally, complex occlusions impair many trackers due to drastic template changes during occlusion. To attenuate their effect, researchers try to combine several features (Mihaylova et al., 2007) each of which is invariant to different variations (e.g. scale-, rotation-, or illumination-invariant). Moreover, in case of complex occlusion, drastic trajectory change of the target (e.g., bouncing) may render specialized motion models useless, such that the target is easily lost due to the improper search region, i.e. away from the target.

Despite the crucial need for occlusion prediction/detection to tailor occlusion-proof trackers, explicit occlusion indicators are rarely found in the literature, mostly task specific. A classic example is the ratio of the observable foreground points to model points (Zhao and Nevatia, 2004). Other methods rely on their appearance model to handle partial and temporal full occlusions (Wu and Nevatia, 2006) or switch the tracking algorithm when the main tracker fails to continue tracking (Thome and Miguet, 2005). Utilizing depth information of RGBD observation, yet (Song and Xiao, 2013) introduced another occlusion indicator for RGBD trackers.

Not being able to use explicit occlusion indicators, several researchers tried to expand the abilities of the PFT to handle wider range of occlusions. Focusing on Condensation PFT revealed that it has inherent drawback called the outlier problem, i.e., a large difference between prior and posterior distributions causes a crude approximation of the posterior distribution. Occlusion, clutter, moving distractors, and insufficient approximation of target dynamics can cause such a problem. It can be concluded that occlusion troubles this tracker in two ways: (i) outliers shifts the approximated posterior away from the real posterior, or (ii) particles are allocated to sample around outliers. To alleviate these problems several other versions of particle filters (e.g. Auxiliary PF by Pitt and Shephard (1999)) are proposed which tweak the trackers in different scenarios. Additionally hybrid and switching particle filters emerged to compensate the shortcomings of a tracker with the merits of another. However, balancing the trade-off between the different trackers heavily depends on the task in hand and scenario conditions, especially occlusion type. As mentioned earlier, switching between PFT and other kinds of trackers (e.g. Mean-shift in (Thome and Miguet, 2005)) is the subject of some researches, yet the inconsistency between different aspects of the trackers impedes successful switching and hence avoids occlusion handling in many occasions. In Duan et al. (2009), the motion model of PFT switches to random walk in the event of occlusion to facilitate target recovery. Once the total likelihood of the particles falls below a certain threshold, the system reports an occlusion case. Such measurement is not uniquely caused by occlusion (e.g. trajectory changes have same effect). In another study (Bando et al., 2006), a shared pool of particles are sampled by two different versions of particle filter, one good at handling occlusions and the other powerful in accurate localization by pre-determined linear switching functions. This over-simplified approach suffers from parameter sensitivity and is unable to handle various occlusions which requires high flexibility.

In summary, the major issues that will hinder the tracking of all template-matching trackers (e.g., PFTs) during occlusions are *(i)* lack of occlusion detection module, *(ii)* model drift, *(iii)* local optima of the feature space (Brasnett et al., 2007), *(iv)* the outlier problem, *(v)* noise and other defects in feature calculation, and *(vi)* misguided/uninformed search for occluded target.

To address these issues, we propose a binary flag to be attached to each particle, which expresses the state of the tracker's belief regarding the particle occlusion. Based on this "occlusion flag", the particle goes through the feature-based template matching process (no-occlusion case) or branches to an occlusion case in which all of the occluded particles are treated uniformly. The latter case guarantees the quick expansion of search area. The state transition model devised for the occlusion case allows quick recovery of reappeared target and robust prediction of emergent occlusions. Moreover, the occlusion detection stops the model update to prevent the model from being corrupted by irrelevant data which in tern leads to resolution of the model drift problem. Predicting occlusions and preemptive suspension of the model update are plausible solutions to handle the model drift problem. Such occlusion-awareness not only resolves this issue, but also accelerates target recovery if handled effectively. Additionally, the framework accommodates arbitrary number of features to be fused. Such feature fusion enables the tracker to localize the target by using the available information effectively. The fusion procedure also renders the

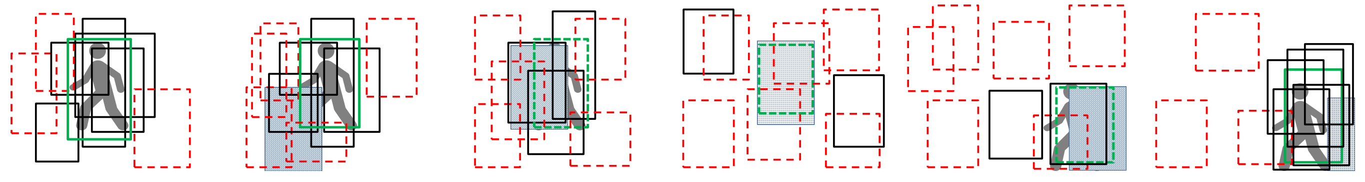


Figure 2. Phases of the occlusion recovery, Black: non-occluded particles, Red: occluded particles, Green: estimated target, Dashed: occlusion flag set

feature-space easier to approximate the target and hence improves the mapping between observation and target estimation. Besides it brings robustness against feature noise and failures (such as illumination change for color feature). Using depth channel in addition to color channels foster handling appearance changes, camera movements and spatial disambiguation of similar objects.

Following this introduction, the proposed method is elaborated in section 2. Then this algorithm is compared with the original PFT and a couple of state-of-the-art RGB and RGBD trackers. The manuscript is then summed up with discussion and future works.

2. Proposed Method

This study proposes a tracker which exposes the particle filter to probabilistic treatment of occlusions. As a result this tracker handles persistent and complex occlusions and enjoys quick occlusion recovery, while benefiting from fusion of various features collected from color and depth channels. This section present an overview of the particle filter tracker, followed by the basic idea underlying our proposed method.

2.1. Overview of Particle Filter Tracker

Visual tracking attempts to localize the designated target(s) given sequential observations. Denoting target state at a discretized time t as X_t and corresponding observation as Y_t, the goal of tracking is defined as obtaining the posterior of the target state X_t given all the previous observations, $p(X_t|Y_1, Y_2, \ldots, Y_t)$ while the initial distribution $p(X_1)$ is assumed to be known. A first-order Markov assumption allowed the stochastic process of the target state and observation to be represented by two state space equations, the transition model (or motion model in the tracking realm) and the observation model:

$$X_{t+1} = \kappa(X_t) + \omega_{t+1}\ ,\ Y_t = \psi(X_t) + \upsilon_t. \quad (1)$$

Here, ω_t and υ_t are i.i.d. system and observation noises respectively, both from known probability distributions. Besides, $\kappa(.)$ and $\psi(.)$ are known functions. In many real-world scenarios, the former two are non-Gaussian and the latter two are non-linear (Nummiaro et al., 2003).

The particle filter is a technique to allow a set of particles to represent the posterior and hence the expectation based on the posterior to yield the approximation of the target, $\hat{X}_t \equiv \mathbb{E}[X_t|Y_1, Y_2, \ldots, Y_t]$. In visual tracking domain the information I_t is acquired from sensors, X_t is the target representation (bounding boxes, skeletons, etc.), and the observation Y_t is the area of the input information defined by target representation X_t, i.e., $Y_t = \psi(I_t; X_t)$. The tracking task is then reduced to a template matching problem (Perez et al., 2004), where the target distribution (i.e. corresponding observation) should match the template closely. The template is initialized by first target observation and evolves by time. The template matching is done in the feature space to make use of the advantages of well-established features, such as color, texture, edges, gradients, and 3D shapes. To estimate the target state (e.g., location) as the posterior distribution given the previous observations, PFT uses a number of (N) particles. These particles are initialized on the initial target state X_1, and then are moved according to the motion model, $X_t = \kappa(X_{t-1})\ ,\ t > 1$. Next, the likelihood is assigned to each particle to show how likely the particle represents the target state, which is measured by the similarity between the particle and the current template θ_t. The target state is then estimated as $\hat{X}_t$ by a weighted sum of all particles. In the next step, the tracker updates the template with the correspondent observation of the estimated state $\hat{Y}_t = \psi(I_t, \hat{X}_t)$ (for details, see eq (9)). Finally, all particles are going through a resampling phase in which the same number of particles are newly drawn from the previous set to be proportional to the current posterior distribution. This step reproduces the best set of particles which closely represent the current posterior and prepares the tracker for the subsequent observations.

The intuition behind tracking with PFTs is to use several candidates for target, all sampled around the expected target location considering its motion pattern. These candidates resemble the target to a certain degree. This similarity serves as the weight in the linear combination of all particles voting for the new location of the target. The dashed gray arrows in Figure 1(c) depict these steps. The additional steps of the proposed method are colored as red in the same flowchart, with the solid arrows indicating the control flow of the algorithm.

2.2. Proposed Model

To handle persistent and complex occlusions, the algorithm should predict/detect emergent occlusions to secure template from updating with irrelevant data. During the occlusion, the particles should not be disturbed by non-target parts of the scene. Further, the search area should be expanded gradually to capture the target, in the case that its course and/or velocity change during occlusion. After the occlusion, however, the tracker should converge to the target and keep on tracking. To realize these goals, in this proposed method a binary occlusion flag is introduced to the state representation of every particle in the PFT.

These flags partition the whole particle population into two fraction: the ones which are considered as occluded and the ones which are not. In the beginning of the algorithm a few of the particles are stochastically marked as occluded (Figure

```
input  : RGBD sequence $I_{t=1,\dots,T}$, Target box $B_1$
output: Target bounding box $\hat{B}_t$ and occlusion state $\hat{Z}_t$
1  Initialize target template [eq(8)];
2  Create N particles ($B_{1,j} \leftarrow B_{init}, Z_{1,j} \leftarrow 0$);
3  for $t \leftarrow 2$ to $T$ do
       // loop over frames
4      for $j \leftarrow 1$ to $N$ do
           // loop over particles
5          if $Z_{j,t} == 1$ then
               // particle occluded
6              Apply random walk motion model [eq(5)];
7              Calculate likelihood [eq(3)];
8          else // particle not occluded
9              Apply motion model to the particle [eq(5)];
10             Extract features from the particle, calculate its
               distance from target template, and calculate
               likelihood [eq(4)] ;
11         end
12     end
13     Normalize particles' likelihoods [section (2.4)];
14     Approximate new target state $\hat{B}_t$ from non occluded
       particles [eq(6)];
15     Calculate occlusion state $\hat{Z}_t$ from all particles [eq(7)];
16     if $\hat{Z}_t == 0$ then
           // target is voted as non occluded
17         Update template $\theta_{t+1}$ using estimated $\hat{B}_t$ [eq(9)];
18     end
19     Resample particles based on their probabilities;
20 end
```

Algorithm 1: Occlusion aware particle filter tracker. This algorithm assumes no prior knowledge about the target, or its motion model. The motion model in line 9 is arbitrary (e.g. second degree motion model).

2(a)), and start to scatter away from the target. If they stumble upon an occluder, they remain occluded while other particles join them in the occluded fraction (Figure 2(c)). If many of the particles are marked as occluded, the target state is perceived as an occlusion state and the model update stops (Figure 2(d)). During the occlusion, the occluded particles moving in a random walk pattern, scatter away from the last known position of the target and search a wider area by time (Figure 2(e)). If a strong resemblance to target is found, that particle is replicated multiple times in the population, and in few frames (up to 3 frames in our experiments) the target is recovered from occlusion state (Figure 2(f)) and the tracking continues. Formally speaking, the observation is augmented with a binary latent variable to enable the sampling method to arbitrary select between two motion models and likelihood calculation.

In no-occlusion situations, the algorithm works similar to the original PFT, with a few particles marked as occluded, illustrated by red in Figure 1(a). In occlusion cases, however, the occluded particles overcount the non-occluded ones, imposing the occlusion flag of the estimated target to be set and preventing the model update. The particles start to scatter quickly as shown in Figure 1(b) which illustrates the particles on 3 frames after the start of the full occlusion. The whole algorithm scheme is displayed in Figure 2 and Algorithm 1. [1]

2.3. Formalism

In our framework, a single particle $X_{j,t}$ ($j = 1, \dots, N$) at time t is represented by bounding box $B_{j,t}$ paired with the binary occlusion flag $Z_{j,t}$, i.e., $X_{j,t} \equiv \{B_{j,t}, Z_{j,t}\}$. The box is characterized by its center coordinates, width and height, $B_{j,t} \equiv \{x_{j,t}, y_{j,t}, w_{j,t}, h_{j,t}\}$. The particle index j is omitted for readability hereafter. An RGBD observation with sensors such as Microsoft Kinect is composed of two channels, the color image $I_{t,\text{rgb}}$ (which is the projection of 3D shape on the image plane) and the depth map $I_{t,\text{d}}$ (which only gives the depth measurement of the points in the line-of-sight). Together, these channels shape the observation $I_t \equiv \{I_{t,\text{rgb}}, I_{t,\text{d}}\}$ which is an incomplete representation of 3D space containing only the visible points. Having defined Y_t as a patch of I_t embodied in the bounding box B_t, the observation itself has color and depth components, $Y_t \equiv \{Y_{t,\text{rgb}}, Y_{t,\text{d}}\}$ (Figure 3(a)).

The observation model is divided into an occlusion case $p\,(Y_t|B_t, Z_t = 1, \theta_t)$ and a no-occlusion case $p\,(Y_t|B_t, Z_t = 0, \theta_t)$. These two cases are treated differently: while the occlusion case allows the tracker to take exploratory behaviors during occlusion, the no-occlusion case promotes a feature-based template matching process between particle B_t and template θ_t.

$$\begin{aligned} p(I_t|X_t, \theta_t) \propto\ & (1 - Z_t) p\,(Y_t|B_t, Z_t = 0, \theta_t) \\ & + Z_t p\,(Y_t|B_t, Z_t = 1, \theta_t) \end{aligned} \tag{2}$$

The likelihood in the occlusion case follows a uniform distribution while the no-occlusion case is derived from fusing different features. All the particles marked as occluded would be treated similarly since none of them has any valid information about the target.

$$p\,(Y_t|B_t, Z_t = 1, \theta_t) \propto 1 \tag{3}$$

The second term in eq (2) enables the particle to follow the target by evaluating multiple features in the no-occlusion case. Assuming the independence between M features, the likelihood is given by

$$p\,(Y_t|B_t, Z_t = 0, \theta_t) \propto \prod_{i=1}^{M} \exp\left(-\frac{D_i\,(f_i(Y_t), \theta_{i,t})}{\sigma_i}\right), \tag{4}$$

where $D_i(f_i(Y_t), \theta_{i,t})$ measures the dissimilarity (distance) between the i-th feature f_i extracted from the observation patch Y_t in the bounding box and the respective section of the template feature vector, $\theta_{i,t}$. Parameter σ_i is a weighting factor for the i-th feature; although its value was determined heuristically, its setting was not very important due to the normalization process explained below. Moreover, M is the number of features in the feature set $\mathbb{F} = \{f_1, \dots, f_M\}$ (Table 1 enumerates the features used in our implementation).

[1]A preliminary non-peer-reviewed version of this algorithm was presented in Meshgi et al. (2014).

Assuming an independent temporal smoothness on the bounding box B_t and the occlusion flag Z_t, the motion model is given by

$$p(X_{t+1}|X_t) = p(B_{t+1}, Z_{t+1}|B_t, Z_t) = p(B_{t+1}|B_t)p(Z_{t+1}|Z_t). \quad (5)$$

The motion model for the non-occluded particles is arbitrary (e.g. second order) while for occluded particles the motion model of the bounding box, $p(B_{t+1}|B_t)$, is a zero-mean Gaussian with a diagonal covariance matrix. The variance of the model is given by σ_B and its value is consistent over the positions and scales of the bounding box. The transition model of occlusion, $p(Z_{t+1}|Z_t)$, is a 2×2 probabilistic matrix and was determined by cross-validation over different videos in the dataset.

Based on equations (2) and (5), the bounding box B_t and the occlusion flag Z_t are estimated according to the usual algorithm of the particle filter. This process yields a set of particles which approximates the posterior distribution. As for the estimates of the bounding box B_t and the occlusion flag Z_t, we take expectation with respect to the posterior distribution $p(B_t, Z_t|I_1, \cdots, I_t)$.

$$\begin{aligned} &\mathbb{E}\left[B_t | I_1, \ldots, I_t, Z_t = 0\right] \\ &\approx \sum_{j \in \mathcal{J}_t} B_{j,t} \frac{p(I_t|B_t = B_{j,t}, Z_{j,t} = 0, \theta_t)}{\sum_{j' \in \mathcal{J}'_t} p(I_t|X_{j',t}, \theta_t)} = \hat{B}_t \end{aligned} \quad (6)$$

$$\begin{aligned} &u(\mathbb{E}\left[Z_t|I_1, \ldots, I_t\right] - \delta_{occ}) \\ &\approx u\left(\sum_{j=1}^{N} Z_{j,t} \frac{p(I_t|B_t = B_{j,t}, Z_{j,t}, \theta_t)}{\sum_{j'=1}^{N} p(I_t|B_t = B_{j',t}, Z_t = Z_{j',t}, \theta_t)} - \delta_{occ}\right) = \hat{Z}_t \end{aligned} \quad (7)$$

Here $X_{j,t} = \{B_{j,t}, Z_{j,t}\}$ is a sample from $p(X_t|I_1, \ldots, I_{t-1}, \theta_t)$. Equation (7) is given by a weighted voting of the particles compared against the occlusion threshold δ_{occ}. This is done by letting $\mathcal{J}_t^{nocc}$ be the set of non occluded particles while step function $u(x)$ is set to one if x is positive and zero otherwise.

As for the target model, or the template, θ_t, the initial template is given by detecting features from the target bounding box $X_1 = \{B_{init}, Z_1 = 0\}$ using a detector (e.g. face detector) or user input (B_{init}).

$$\theta_1 = \{\theta_{1,i}\} = \{f_i(Y_1)\} = \{f_i(\psi(B_{init}))\}\ ,\ i = 1, \ldots, M \quad (8)$$

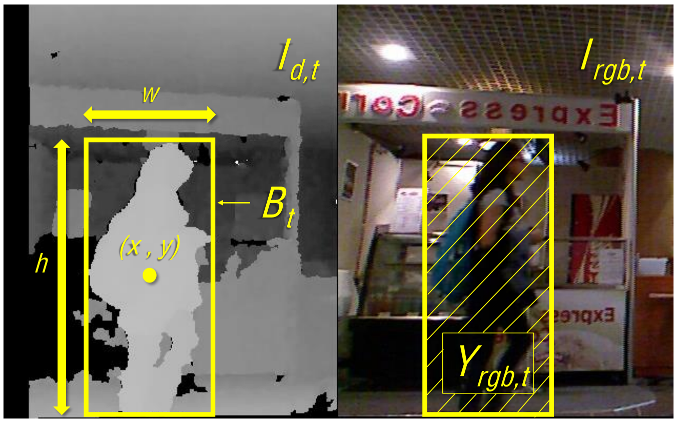
For $t > 1$, the template is updated individually for each feature i after the resampling by a leaky memory scheme. Unless an occlusion state is detected:

$$\theta_{i,t+1} = \begin{cases} \theta_{i,t} & , \hat{Z}_t = 1 \\ \lambda_i f_i(\hat{Y}_t) + (1 - \lambda_i)\,\theta_{i,t} & , \hat{Z}_t = 0 \end{cases} \quad (9)$$

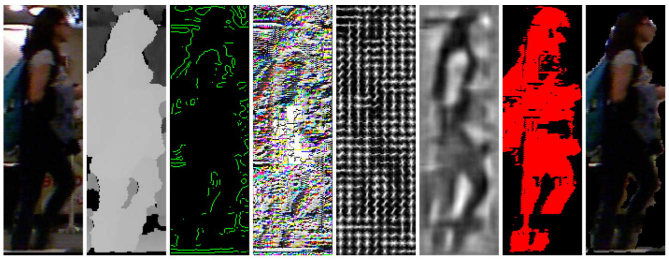
where λ_i is a forgetting factor. In the above equation, $\hat{Y}_t$ is the image patch embodied by the estimated bounding box $\hat{B}_t$, i.e., $\hat{Y}_t = \psi(I_t; \hat{B}_t)$.

2.4. Features

In this study, we used several features, listed in Table 1, to boost the performance and improve the robustness and resilience of the tracker. The corresponding dissimilarity functions were chosen referring to the previous studies, and their parameters were set as recommended in the original papers, otherwise listed in Table 1.



(a) Sensory Information I_t has color and depth components, so does the observation Y_t which is the area of the input images embodied by B_t.



(b) Template image observed from different channels and transformed to various feature spaces; from left to right: (1) RGB color, (2) Depth (brighter=closer to camera), (3) Edges (described by LoG), (4) Texture (described by LBP), (5) HOG domain, (6) HOG image reconstruction (as machine can see it described by iHOG (Vondrick et al., 2013)), (7) Foreground mask, and (8) Foreground mask (filtering out the pixels out of the range of $\pm 20\textit{units}$ from the median depth values in bounding box).

Figure 3. Observation and Feature Space

In addition to the features established in the literature, we proposed a "2D Projection Confidence" feature to enhance the scale adaptation and localization of the target. In a typical bounding box, some parts contain many foreground pixels and others contain many background pixels; such non-uniformity may produce non-uniform information distribution of the foreground object over the bounding box (Nummiaro et al., 2003). Such non-uniformity can be useful, and one simple idea is to decompose the bounding box into a regular $k \times k$ grid, so that each grid cell is represented by the ratio of foreground pixels to all pixels. However, this simple feature template is not robust against object deformation and rotation. In this study, the feature template was represented by a set of Beta distributions, one for each grid cell, each of which regards the ratio as a random variable within the range of $[0, 1]$. To tune the parameters of the Beta distribution in each cell c, α_c and β_c, we used a number of bounding boxes taken from the cross-validation video sequences, in which the target was in different poses, orientations, and shapes. More concretely, from the first image in each video sequence, target objects were extracted using the temporal median technique introduced by Lo and Velastin (2001). Then each bounding box containing a target was decomposed to $k \times k$ grid cells, and for each grid cell the ratio of the number of foreground pixels to the number of all pixels was calculated. By

Table 1. Letter code of features and employed dissimilarity measures

Code	*Feature Name*	*Dissimilarity Function*	*Parameters*
B	Projection Confidence [eq (10)]	L1	3×3, Tuned on validation set
C	Hist. of Colors (Comaniciu et al., 2003)	KL Divergence	Adaptive Binning 40
D	Hist. of Depth	Chi-Sqaure	256 bins
E	Template of Edges (Huttenlocher et al., 1993)	Hausdorff	–
G	Hist. of Oriented Gradients (Comaniciu et al., 2003)	L2	36 bins, Signed Orientation
S	3D Shape Parameters (Johnson, 1997)	L1	Grid Size 10 pixels
T	Hist. of Texture (Ojala et al., 2002)	Bhattacharyya	64 bins for each color channel

performing this process over different video sequences, the parameters of the set of cell-wise Beta distributions $Beta(x; \alpha_c, \beta_c)$ were estimated. While tracking, the score of each particle grid was calculated as:

$$f_{proj} = \{Beta(r_c; \alpha_c, \beta_c)\}\ ,\ c = \{(1,1), (1,2), \cdots, (k,k)\}. \quad (10)$$

A vectorized version of the above matrix then serves as the "2D Projection Confidence" feature. Intuitively, this feature detects the rough sketch of the target's shape in the bounding box. The coarse nature of gridding provides robustness against deformations or articulations. Thus this feature encourages better scale selection for the bounding box. The feature obtained from a 3×3 grid is adopted for our implementation, and compared against the template using $L1$-norm.

To each feature, an independent regularized linear normalization is applied such that the histogram of the likelihood term (the exponential term in equation (4)) over all the particles is mapped to the $[\epsilon, 1]$ range $(0 < \epsilon < 1)$. After this normalization, the aforementioned likelihood was re-calculated. This normalization was effective for achieving robustness of the tracker. If small number of particles have large distance from the target in terms of a specific feature, they should be eliminated. In contrary, if all the particles are distant from the template, that feature is better to be ignored. This case may happen due to feature failure, caused by for instance a sudden illumination change. Furthermore if the feature likelihood of a certain particle is miscalculated (e.g., due to observation noise), that feature should not disable the particle so that there is a chance that other features mitigate this failure. The regularization term was thus introduced to ensure that the particles have similar values in case of feature failures. On the other hand, the upper bound of the feature likelihood guaranteed that the tracker would not be stuck to local optima in the existence of features with high similarity.

2.5. Parameters

The proposed algorithm has several sets of parameters. Feature-related parameters (σ_i and λ_i) control the feature fusion and template update. The dynamics of the particle filter are governed by noise variance of the motion model, number of particles and transition matrix. Besides, the ability of the algorithm to detect emergent occlusions and to safely recover from that depends on occlusion threshold and occlusion-case likelihood. The latter parameter is a constant value selected for equation (3), that is $p(Y_t|B_t, Z_t = 1, \theta_t) = L_{occ}$.

Due to normalization of the likelihood, the parameter σ_i was not very sensitive and hence set at an appropriate constant. Occlusion threshold (δ_{occ}), occlusion-case likelihood (L_{occ}), occlusion transition matrix ($p(Z_{t+1}|Z_t)$) and model forgetting factors(λ_i) were determined by cross-validation. The number of particles (N) and noise of the random walk process (σ_B) were determined heuristically (random walk noise was set at three times of the largest movement of the target since the start of tracking).

3. Evaluations

3.1. Experiment

The proposed occlusion aware particle filter tracker (OAPFT) was evaluated in terms of its ability to handle occlusions, its accuracy, and its flexibility, in comparison to an ordinary PFT and state-of-the-art RGB and RGBD trackers. In the experiment, we also explored the optimal set of features for the tracker. The videos used in the experiment, the output of all the trackers on each video, and respective detailed evaluation plots are available in the supplementary material.

Princeton Tracking database (Song and Xiao, 2013) contains 100 manually annotated RGBD video sequences. These videos were captured by Microsoft Kinect which include various occlusion scenarios with different duration. From this database, we took five different videos, `new_ex_occ4`, `face_occ_5`, `bear_front`, `child_no1` and `zcup_move_1`, with total number of frames of 1202, which involve complex, persistent, full, partial, and self- occlusions, respectively. Using these five videos, we expect to evaluate the trackers' performance from different aspects: features, occlusion conditions, and motion patterns.

To explore the optimal set of features, the proposed OAPFT was examined with different feature sets; in the following, we use the letter codes (Table 1) for the employed features. For instance, tracker *CDE* used three features: histogram of colors (C), histogram of depth (D), and template of edges (E).

The model update scheme utilized in our algorithm is motivated by Nummiaro et al. (2003). In the mentioned study, the authors presented an adaptive particle filter with a model update scheme based on the feature of color histogram of pixels. In Table 2, this algorithm is denoted by *ACPF* (Adaptive Color-based Particle Filter).

As a representative of state-of-the-art RGBD trackers, we chose a tracker presented by Song and Xiao (2013) which performs a learning-based tracking accompanied by an occlusion detector. This algorithm updates the template of the target appearance regularly during the tracking, by using an SVM trained based on several features such as color histogram, depth histogram, histogram of oriented gradients over color and depth images, and 3D shape parameters. Once the tracker detects an occlusion, it stops updating its template. This SVM-tracker is denoted by *OI+SVM* hereafter.

Additionally, we compared our results with the results of *STRUCK* (Hare et al., 2011), one of the most successful appearance-based trackers in handling occlusions (Wu et al., 2013). This algorithm employed kernelized structured output SVM which learns to predict the target location in a tracking-by-detection framework. This algorithm utilizes structured output SVM to directly estimate the object transformation between frames. The authors applied a budgeting mechanism on SVM to meet the real-time processing requirement of online trackers. This tracker is capable of using different feature kernels such as Haar-like features, intensity histogram, and raw features. In our experiment, we used the same kernels and parameters listed in Table 2 in the original paper (Hare et al., 2011).

The performance of each tracking algorithm was evaluated in terms of several criteria. The overall performance was measured using tracking success degree S_t, which is given by the ratio of the intersection between the estimated target area and the real target area to the union of these two areas, if not occluded. More concretely, the tracking success degree is given by (Song and Xiao, 2013)

$$S_t = \begin{cases} \frac{|\hat{B}_t \cap B_t^*|}{|\hat{B}_t \cup B_t^*|} & , \hat{Z}_t = Z_t^* = not\ occluded \\ 1 & , \hat{Z}_t = Z_t^* = occluded \\ -1 & , otherwise \end{cases} \qquad (11)$$

where $|.|$ denotes the number of pixels in the defined region, and $\hat{B}_t$ and B_t^* denote the estimated and real target, respectively. The occlusion states of the estimated and real targets are denoted by $\hat{Z}_{t+1}$ and Z_{t+1}^* respectively. To measure the tracker's overall performance on all video frames, we counted the number of frames in which the success degree was larger than a given threshold t_o. The area under the curve (AUC) of the resulting plot (success frames vs. t_o) is reported in Table 2.

To provide better insight into the algorithm's outcomes, the errors of central point location (CPE) and scale (SAE) were defined as the L2-norm difference of center position (x, y) and scale (w, h) between the estimated bounding box and the true bounding box, respectively. Their average values for all video frames are presented in Table 2.

Along with these measures, the reliability of the tracker, i.e., its ability to deal with occlusions, is measured. In a false tracking case, the tracker did not recognize that the target was in fact occluded; FT denotes the rate of such bounding boxes to the whole set of bounding boxes. In a target miss case, the target was visible but the tracker failed to track it, assuming the target was still in the occlusion state; MI denotes the rate of such cases. Furthermore, a mismatch case was detected when the estimated bounding box had no overlap with any of the true bounding box; MT denotes the ratio. Finally the execution time of the algorithms was measured in frames per second (FPS) and presented for comparison. All the experiments were conducted on a 3.50 GHz 64-bit Pentium IV computer where the proposed algorithm (*OAPFT*) and *ACPF* were implemented with Matlab, *STRUCK* was implemented with C++ and *OI+SVM* was implemented with Matlab/C.

Table 2. Performance evaluation of algorithms. OAPFT (proposed) with different features on the upper panel, and OI+SVM (Song and Xiao, 2013), STRUCK (Hare et al., 2011), and ACPF (Nummiaro et al., 2003) on the lower panel. Each metric, averaged for all videos, is presented. Bold font shows the best value for each metric.

Tracker	cc[#]	AUC	CPE	SAE	MI	FT	MT	FPS
C	🟥	53.99	34.44	14.04	**0.0**	0.8	11.4	8.2
D	🟩	61.02	31.53	17.57	**0.0**	1.6	20.6	**14.4**
E	🟦	21.09	90.95	23.63	12.6	**0.0**	81.6	8.2
S	🟦	26.41	124.07	21.05	3.4	0.6	121.	21.2
CD	🟪	68.52	16.26	15.83	**0.0**	0.8	2.4	8.3
CE	🟧	37.81	56.66	20.37	12.0	0.6	39.4	5.6
CG	⬜	54.76	34.61	12.72	**0.0**	1.6	26.4	6.1
CDE	🟪	58.20	24.63	17.93	2.8	0.8	**0.0**	5.7
CGS	🟫	37.69	47.56	16.71	**0.0**	83.6	32.2	1.1
CGT	⬛	55.14	28.06	13.66	**0.0**	1.6	3.8	3.8
CDET	🟨	63.48	19.49	14.81	**0.0**	1.4	**0.0**	3.7
CDGT	🟪	74.14	12.30	11.81	**0.0**	1.2	**0.0**	3.8
CDEST	🟦	54.94	34.20	16.25	2.8	0.8	29.6	1.0
CDEGST	🟩	72.94	12.03	11.25	**0.0**	1.6	**0.0**	0.9
BCDEGST	🟧	**76.50**	**9.59**	**7.32**	**0.0**	2.4	**0.0**	0.9
OI+SVM	🟪	69.15	9.68	12.04	0.4	20.0	0.8	0.4
STRUCK	🟪	46.67	68.74	26.61	12.6	**0.0**	64.4	13.4
ACPF	🟦	27.55	90.38	35.27	12.6	**0.0**	31.0	1.4

[#] *cc – Color Code for the Tracker*

3.2. Results

Figure 4 illustrates the performance of the trackers for the first video sequence. The vertical dashed lines in the lower panels show the start and end of full occlusion. We can see that our OAPFT (here *BCDEGST*) successfully detected an emergent occlusion a few frames prior to the full occlusion and again retrieved the target soon after it re-appeared. And in most cases, our OAPFT was successful in keeping the template during the occlusion, and smooth tracking after the occlusion. When losing the target, on the other hand, the algorithm cautiously prevented false tracking by maintaining an occlusion state and preserving the last known template. As demonstrated in Table 2, our OAPFT actually showed fairly low false tracking (FT) rates.

Figure 5 clearly reflects that that an appropriate combination of features was crucial for successful tracking. A good example is the video illustrated in Figure 4, where color, edges and shape features and their pure combination were not adequate, and misled the tracker to frequently track the background or other similar objects. When the features covered different aspects of the object and teamed up with each other, the performance of the tracker was reasonably enhanced, see e.g., *CGS*. There were some cases that one of the features contributed substantially in tracking, but when combined with other features, its accuracy was further improved (e.g., depth feature here). Table 2 compares all the algorithms and provides more insight into the optimal feature set in terms of the AUC and CPE values.

In the event of a partial occlusion followed by a full occlusion, the size of the corresponding bounding box would increase, because the tracker attempts to keep the boxes which

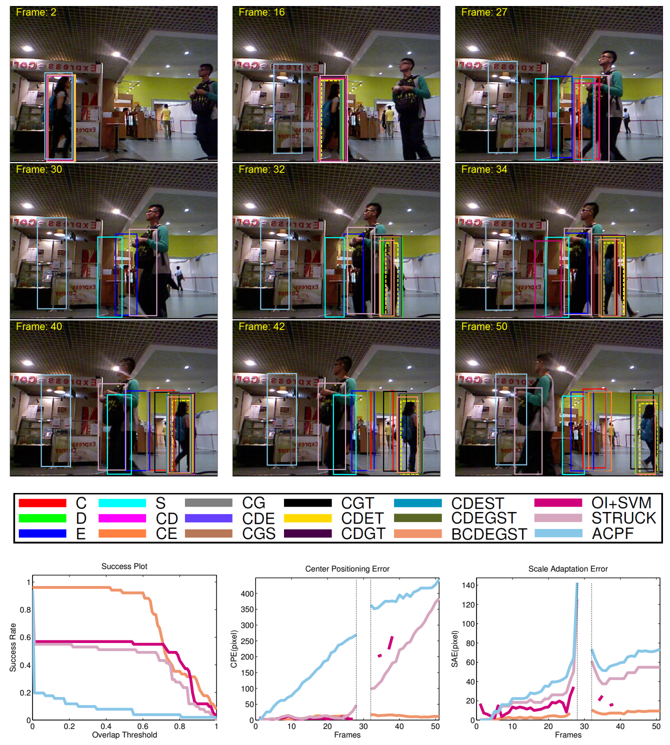


Figure 4. Tracking by various trackers (upper panel). The ground truth is marked with yellow dashed line. This figure shows that ACPF tracker was trapped in the background clutter and could not follow the target. Also OI+SVM lost the target then it tried to recover the target around frame 38 (see lower panel) but failed eventually. In the performance plots (lower panel), the proposed OAPFT (*BCDEGST*) shows the best tracking performance. Refer to supplementary material for further detailed analysis of all video sequences.

widely cover the visible parts of the target(s) and larger boxes often have higher probability of having those included. After recovering from the occlusion, the size of the bounding box quickly reduced to suit the target (Figure 4).

The fastest processing time was achieved by the depth tracker (*D*), while by addition of features such as 3D shape or HOG, more time was needed. It should be noted however that the current implementation used Matlab and the speed could be boosted using GPU programming or low-level languages such as C++ to meet the real-time requirements.

In summary, Table 2 and the plots in Figure 4 show that the advantages of the proposed OAPFT involve quick recovery from occlusion, better robustness against tracker's failure (since the irrelevant particles are marked as occluded stochastically), and tighter grip on the target. The reasults indicate that the proposed OAPFT with the full set of features (*BCDEGST*) outperformed the existing trackers, *OI+SVM*, *STRUCK*, and *ACPF*. Furthermore, good choice of features, including a newly developed 2D projection confidence feature improved the tracking performance even in variety of occlusion conditions.

4. Conclusion

In this study, we presented a novel modification of particle filter trackers with a particular interest in handling persistent and complex occlusions. Thanks to the newly developed 2D projection confidence feature, this tracker further exhibits high adaptability to changes in object scale and trajectory. Each particle in the proposed framework was evaluated in terms of multiple features, and then compared to the target template to measure the similarity, which conveys the probability of the target. The data revealed that our set of seven features outperforms other any subset of this feature set, and selecting effective features requires keen attention to the aspects of the video sequence other features are monitoring. Interestingly, it was observed that adding more features does not necessarily improve tracking accuracy. The most important novelty of the proposed method comprises to handle persistent and complex occlusions explicitly using an occlusion flag attached to each particle and its probabilistic treatment. The flag signals if the bounding box is occluded, then triggers the stochastic mechanism that expands the search area and stops the template update.

This framework accommodates arbitrary number of features from different channels. As shown by the results, the algorithm was capable of more accurate tracking compared to each feature in isolation or a subset of them. The performance of the tracker could be further boosted using better features such as trained convolutional neural networks which is ground breaking in many computer vision domains (Razavian et al., 2014) or using adaptive weights for each feature channel (Dou and Li, 2014). The most significant merits of the framework are preventing model drift, quick recovery from occlusions, and facilitating the fusion of features. The next step toward having a further robust and accurate tracker would be to incorporate the confidence of each data channel into the tracking, and to update the model adaptively by each observation.

Acknowledgments

We are grateful to Maryam Sadat Mirzaei, the reviewers, and the editor for their careful and close reading of the manuscript.

References

Bando, T., Shibata, T., Doya, K., Ishii, S., 2006. Switching particle filters for efficient visual tracking. RAS .

Brasnett, P., Mihaylova, L., Bull, D., Canagarajah, N., 2007. Sequential monte carlo tracking by fusing multiple cues in video sequences. Image and Vision Computing 25, 1217–1227.

Comaniciu, D., Ramesh, V., Meer, P., 2003. Kernel-based object tracking. PAMI, IEEE Trans. 25, 564–577.

Dou, J.f., Li, J.x., 2014. Robust visual tracking base on adaptively multi-feature fusion and particle filter. Optik-International Journal for Light and Electron Optics 125, 1680–1686.

Duan, Z., Cai, Z., Yu, J., 2009. Occlusion detection and recovery in video object tracking based on adaptive particle filters, in: Control and Decision Conference, 2009, IEEE. pp. 466–469.

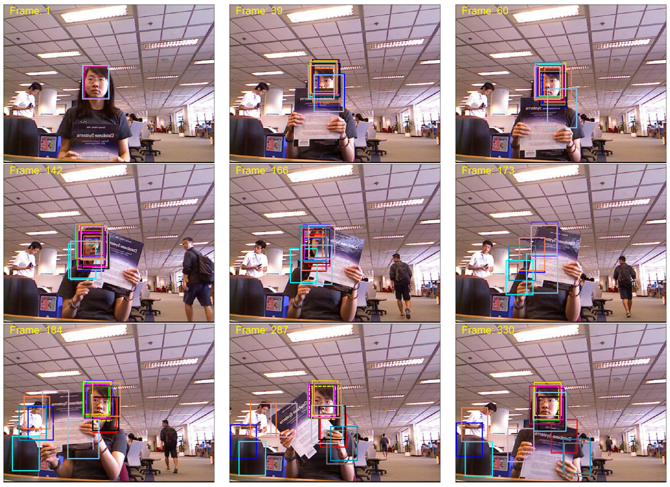
Hare, S., Saffari, A., Torr, P.H., 2011. Struck: Structured output tracking with kernels, in: ICCV, 2011 IEEE Intl. Conf., IEEE. pp. 263–270.

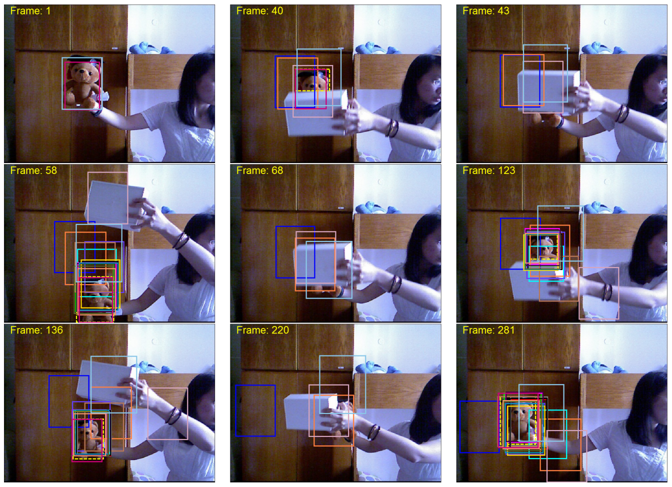
Huttenlocher, D.P., Klanderman, G.A., Rucklidge, W.J., 1993. Comparing images using the hausdorff distance. PAMI, IEEE Trans. 15, 850–863.

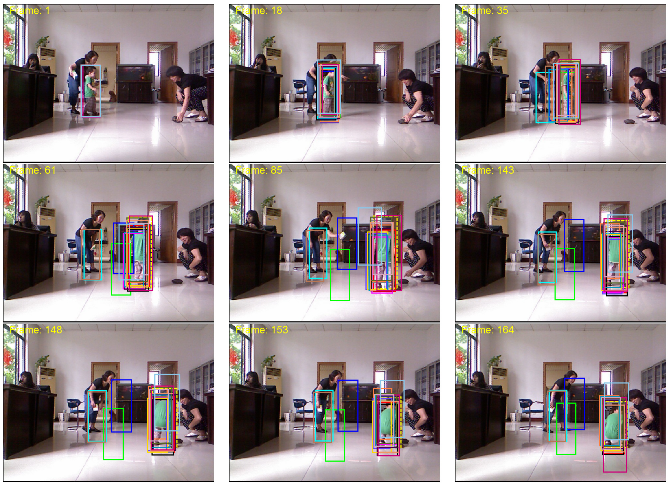
Isard, M., Blake, A., 1998. Condensation—conditional density propagation for visual tracking. IJCV 29, 5–28.

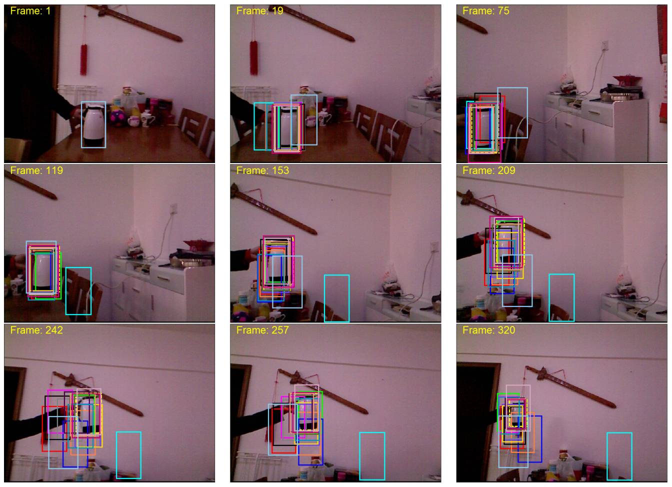
Johnson, A.E., 1997. Spin-images: a representation for 3-D surface matching. Ph.D. thesis. Citeseer.

Lo, B., Velastin, S., 2001. Automatic congestion detection system for underground platforms, in: Intelligent Multimedia, Video and Speech Processing, 2001. Intl. Symp., IEEE. pp. 158–161.


(a) (C)olor, (D)epth


(b) (C)olor, (D)epth


(c) (C)olor, (G)radients


(d) (D)epth, (G)radients, (T)exture

🟥	C	🟦	S	⬜	CG	⬛	CGT	🟦	CDEST	🟪	OI+SVM
🟩	D	🟪	CD	🟦	CDE	🟨	CDET	🟩	CDEGST	⬜	STRUCK
🟦	E	🟧	CE	🟫	CGS	🟪	CDGT	🟧	BCDEGST	🟦	ACPF

Figure 5. A visual review of the trackers performance, along with their most effective features to track the target. For more information refer to supplementary material and project webpage.

Meshgi, K., Maeda, S.i., Oba, S., Ishii, S., 2014. Fusion of multiple cues from color and depth domains using occlusion aware bayesian tracker. Technical Report of Neuro Computing 113, 127–132.

Mihaylova, L., Brasnett, P., Canagarajah, N., Bull, D., 2007. Object tracking by particle filtering techniques in video sequences. Advances and Challenges in Multisensor Data and Information , 260–268.

Nummiaro, K., Koller-Meier, E., Van Gool, L., 2003. An adaptive color-based particle filter. J. Image and Vision Computing 21, 99–110.

Ojala, T., Pietikainen, M., Maenpaa, T., 2002. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. PAMI, IEEE Trans. 24, 971–987.

Pan, J., Hu, B., 2007. Robust object tracking against template drift, in: ICIP 2007. IEEE Intl. Conf., IEEE. pp. III–353.

Perez, P., Vermaak, J., Blake, A., 2004. Data fusion for visual tracking with particles. Proceedings of the IEEE 92, 495–513.

Pitt, M.K., Shephard, N., 1999. Filtering via simulation: Auxiliary particle filters. Journal of the American statistical association 94, 590–599.

Razavian, A.S., Azizpour, H., Sullivan, J., Carlsson, S., 2014. Cnn features off-the-shelf: an astounding baseline for recognition. arXiv preprint arXiv:1403.6382 .

Song, S., Xiao, J., 2013. Tracking revisited using rgbd camera: Unified benchmark and baselines, in: ICCV 2013. IEEE Intl Conf., IEEE.

Thome, N., Miguet, S., 2005. A robust appearance model for tracking human motions, in: Advanced Video and Signal Based Surveillance, 2005. AVSS 2005. Conf., IEEE. pp. 528–533.

Vezzani, R., Grana, C., Cucchiara, R., 2011. Probabilistic people tracking with appearance models and occlusion classification: The ad-hoc system. Pattern Recognition Letters 32, 867–877.

Vondrick, C., Khosla, A., Malisiewicz, T., Torralba, A., 2013. HOGgles: Visualizing Object Detection Features. ICCV 2013, Intl. Conf. .

Wu, B., Nevatia, R., 2006. Tracking of multiple, partially occluded humans based on static body part detection, in: CVPR 2006, IEEE. pp. 951–958.

Wu, Y., Lim, J., Yang, M.H., 2013. Online object tracking: A benchmark, in: CVPR, 2013 IEEE Conf., IEEE. pp. 2411–2418.

Zhao, T., Nevatia, R., 2004. Tracking multiple humans in complex situations. PAMI, IEEE Trans. 26, 1208–1221.