Reliable Patch Trackers: Robust Visual Tracking by Exploiting Reliable Patches

Yang Li and Jianke Zhu
College of Computer Science
Zhejiang University, China
{liyang89, jkzhu}@zju.edu.cn

Steven C.H. Hoi
School of Information System
Singapore Management University
chhoi@smu.edu.sg

Abstract

Most modern trackers typically employ a bounding box given in the first frame to track visual objects, where their tracking results are often sensitive to the initialization. In this paper, we propose a new tracking method, Reliable Patch Trackers (RPT), which attempts to identify and exploit the reliable patches that can be tracked effectively through the whole tracking process. Specifically, we present a tracking reliability metric to measure how reliably a patch can be tracked, where a probability model is proposed to estimate the distribution of reliable patches under a sequential Monte Carlo framework. As the reliable patches distributed over the image, we exploit the motion trajectories to distinguish them from the background. Therefore, the visual object can be defined as the clustering of homo-trajectory patches, where a Hough voting-like scheme is employed to estimate the target state. Encouraging experimental results on a large set of sequences showed that the proposed approach is very effective and in comparison to the state-of-the-art trackers. The full source code of our implementation will be publicly available.

1. Introduction

Visual object tracking is a fundamental research topic in computer vision, which enjoys a wide range of applications, including video surveillance, robotics, driverless car, etc. Despite having achieved promising progress over the past decade, it remains very challenging for designing an all-situation-handled tracker that can handle various critical situations, such as illumination changes, geometric deformations, fast motions, partial occlusion, and background clutters, etc.

Structural representation has recently been studied actively in tracking community, which has been shown as an effective approach to enhancing the robustness. Typically, a grid-like structure [41, 17, 12], typically a bounding box, is employed to represent the target object for tracking. Since most of the tracked target is not strictly a rectangular shape, the bounding box representation often inevitably incorporates background information into the model, and thus could degrade the overall performance of the tracker. Therefore, the grid-like structure is not the optimal way to represent real objects that are of non-rectangular shape.

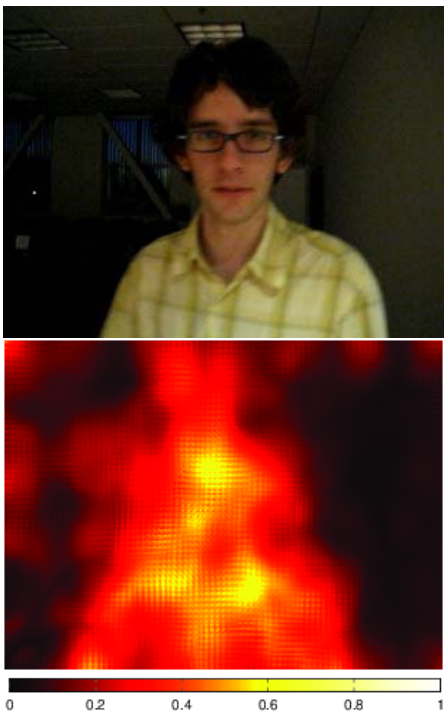


Figure 1. The tracking confidence map of the next frame when the tracker was initialized at different positions with a fixed bounding box. The trackability varies largely in the image.

To account for the object shape in tracking tasks, segmentation-based approaches [22, 27] have been explored to build accurate representations for model-free targets. Although video segmentation [10] has found some promising progress for object tracking, it is often very computationally intensive to generate object proposals and link them across frames. Moreover, it is also quite challenging for the segmentation-based method to deal with cluttered background and occlusions, which often lead to unstable results.

Another family of tracking approaches that are naturally

robust to occlusions and deformations is keypoint-based trackers [32, 4] which represent the visual object by a set of salient points like SIFT or SURF features [24]. However, they may suffer from difficulty in capturing global information of the tracked target by only using the local points, especially for the object with large homogenous region.

To address the above limitations, we propose a novel Reliable Patch Trackers (RPT), which aims to identify and exploit the most reliable patches across video for tracking. Instead of resorting to hand-crafted structures, a trackable confidence function is proposed to compute and select the reliable patches, which is capable of capturing the underlying object geometry, as shown in Figure 1. To locate those patches, both trackable confidence and motion information are incorporated into the particle filter framework, which are further employed to vote for the target location and estimate the object scale. Unlike the conventional particle filter-based approaches [31, 35, 25] that often estimate the target state with affine space, our RPT approach infers the distribution of reliable patches by trackability and motion similarity, which are further employed for tracking through a base tracker. During the tracking process, we resample the patch particles only when it is necessary rather than removing all particles and resample their state space at each frame in traditional methods.

In summary, the main contributions of this paper include: (i) a novel sequential Monte Carlo framework to effectively locate visual objects by directly inferring a set of reliable patches through particle filters; (ii) a reliability confidence function for capturing the underlying object geometry without resorting to a hand-crafted structural representation; (3) a reliable patch tracker with an effective updating scheme that takes advantage of a Hough voting scheme to locate the object and estimate its scale; and (4) encouraging results from extensive experiments on online visual object tracking benchmark by comparing against state-of-the-art trackers.

2. Related Work

Video object tracking has been extensively studied in computer vision over the past decade [38, 39, 16, 7, 23, 43, 42]. Comparing with the trackers using holistic feature representations [31, 35, 15], the structured trackers [34, 17, 40, 26, 30, 27] are generally more promising, particularly in the scenarios of deformations and occlusions.

The first major category of related work is the family of part-based trackers, which often employs a predefined structure. Adam et al. [1] proposed to represent the object by the grid of fragments, where the target position is voted from these fragments. Jia et al. [17] proposed to divide the object into small patches by a regular grid; moreover, l_1 sparsity is adopted as the similarity metric to search the closest candidates in next frame. In [34], a flock of trackers was proposed to track a set of patches with regular grid structure that allows certain drifts. Besides the grid representation, the star model and tree structure have also been studied in [40] with promising results. Kwon and Lee [22] also proposed a star-like appearance model, where a particle filter is employed to find the best state of the tracked target. Moreover, it builds the foreground and background models for segmentation in order to further refine the tracking results. Cai et al. [5] proposed to decompose target into superpixels, and then employ graph-matching to link the neighbor frames. Instead of using the predefined structure, our proposed method attempts to find the underlying structure of the target objects during the tracking process.

Another category is the family of keypoint-based trackers [4, 36], which are widely used in SfM and SLAM [18, 19]. In model-free tracking, Grabner et al. [28] proposed a boosting classifier to match the keypoint between different frames. Godec et al. [27] proposed to train online Hough forests to map image patch onto a probabilistic vote, where back projection along with segmentation is used to estimate the object region. Maresca and Petrosino [26] presented a tracker by Generalized Hough Transform and multiple keypoint detection in a fallback framework. Recently, Nebehay and Pflugfelder [30] attempted to enhance the tracking performance by combining keypoint matching and optical flow, where a consensus method was proposed to detect outliers. Unlike the keypoint-based methods, our approach attempts to find the reliable patches dynamically in the tracking process which potentially could be more robust than the keypoint-based approaches.

Besides, our approach is also related to the methods using multiple trackers [20, 21]. For example, Kwon and Lee [20] introduced a decomposition scheme to incorporate different templates for specific appearance of the target. To make it more robust, the base trackers are sampled from the predefined tracker space into the Markov Chain Monte Carlo (MCMC) framework [21].

Unlike the previous approaches, in this paper, we propose to employ the Correlation Filter-based methods [14, 13, 8, 7] as the base trackers, which take advantages of the convolution theorem to effectively learn the object template model and perform tracking in the Fourier domain under the tracking-by-detection framework.

3. Reliable Patch Trackers

In this section, we present the proposed Reliable Patch Trackers (RPT) for robust visual object tracking. First, we introduce a novel sequential Monte Carlo Framework which takes advantage of reliable patches. Second, we investigate how to compute the likelihood of tracking reliability for a patch using visual information and estimate the patch-on-object likelihood from motion information. Finally, we describe the implementation details of the proposed reliable patch trackers.

3.1. Sequential Monte Carlo Framework

The key idea of the proposed method is to identify and track the reliable patches on visual objects. Since it is hard to find the exact reliable patches, we explore the Sequential Monte Carlo Framework [2] to estimate their probability distribution. In the following, we formally formulate the proposed idea.

In general, an image patch is sampled from a bounding box $\mathbf{x} = [x, y, w, h] \in \mathcal{R}^4$. Given the observations in previous frames $z_{1:t-1} = \{z_1, z_2, ..., z_{t-1}\}$, the probability density function that determines whether patch $\mathbf{x}_t$ in the current frame is reliable can be formulated as:

$$p(\mathbf{x}_t|z_{1:t-1}) = \int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|z_{1:t-1})d\mathbf{x}_{t-1} \quad (1)$$

where $p(\mathbf{x}_{t-1}|z_{1:t-1})$ is the state density function. According to the Bayes rule, it can be recursively calculated as:

$$p(\mathbf{x}_t|z_{1:t}) = \frac{p(z_t|\mathbf{x}_t)p(\mathbf{x}_t|z_{1:t-1})}{p(z_t|z_{1:t-1})} \quad (2)$$

where $p(z_t|\mathbf{x}_t)$ is the observation likelihood, and $p(\mathbf{x}_t|\mathbf{x})$ is the transition density function. Let $\mathcal{N}$ denote Gaussian distribution, $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ is defined as:

$$p(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \mathbf{x}_{t-1}, \Psi(\mathbf{x}_{t-1})). \quad (3)$$

where $\Psi_1(\mathbf{x}) = [\mathbf{0} \quad \mathbf{E}]\mathbf{x}$ is a function for selecting the image coordinates. E represents a 2×2 identity matrix. Note that this assumption will allow the reliable parts to move around the object in order to account for the deformations. This will make the tracker more sensitive to the local structures.

Formally, we define a *reliable patch* that has two properties: (1) being trackable; and (2) sticking on the target object. By assuming these two properties are i.i.d., the observation likelihood $p(z_t|\mathbf{x}_t)$ can be formulated as:

$$p(z_t|\mathbf{x}_t) = p_t(z_t|\mathbf{x}_t)p_o(z_t|\mathbf{x}_t) \quad (4)$$

where $p_t(z_t|\mathbf{x}_t)$ denotes the confidence of a patch to be tracked effectively, and $p_o(z_t|\mathbf{x}_t)$ indicates the likelihood that the patch is on the tracked object.

As the state space for variable $\mathbf{x} \in \mathbb{R}^4$ is too large to be directly inferred, we adopt the particle filter [2] to estimate the posterior $p(\mathbf{x}_t|z_{1:t-1})$. As in the Bootstrap filter [2], the particle weight $w_t^{(i)}$ for the i-th patch can be computed by

$$w_t^{(i)} = w_{t-1}^{(i)} p(z_t|\mathbf{x}_t^{(i)}). \quad (5)$$

In this paper, we represent the visual object by a set of reliable patch particles. For simplicity, a patch particle is defined as $\mathbf{X}_t^{(i)} = \{\mathbf{x}_t^{(i)}, \mathbf{V}_t^{(i)}, y_t^{(i)}\}$, where $\mathbf{V}_t^{(i)}$ denotes the trajectory of patch $\mathbf{x}_t^{(i)}$ within a temporal window, and

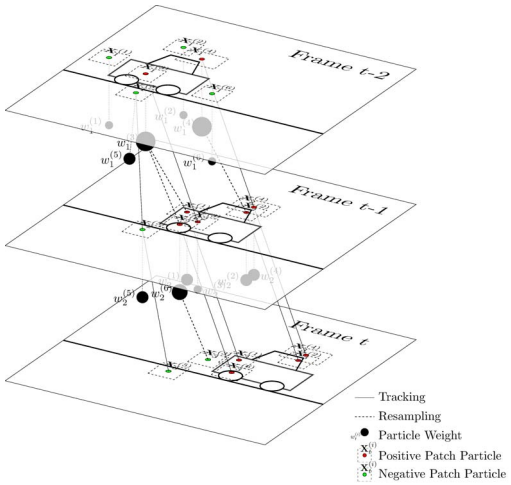


Figure 2. The workflow diagram of the reliable patch particles proceed in three frames. The patch particles estimate the distribution of the reliable patches and track them at the same time.

$y_t^{(i)}$ indicates the label of patch $\mathbf{x}_t^{(i)}$ is positive or negative. Thus, the visual object can be represented as

$$\mathbf{M}_t = \{\mathbf{X}_t^{(1)}, ..., \mathbf{X}_t^{(N)}, \mathbf{x}_t^{target}\}, \quad (6)$$

where $\mathbf{x}_t^{target}$ denotes the final tracked object state. As illustrated in Figure 2, we integrate the process of tracking and estimating reliable patches into a single pipeline. The objective of the proposed tracker is to make $\mathbf{M}_t$ as long as possible while we re-compute the particle weights at each frame. By tracking the reliable patches across frames, the overall performance can be greatly improved.

Remark. Unlike the conventional approaches [31, 35, 25] that estimate the target state with affine space, we directly employ the particle filter to infer whether the reliable patches are on the tracked object. In contrast to traditional particle filters, we do not remove all particles and resample the state space at each frame. Instead, we track the reliable patches through a base tracker. Once the posterior of each reliable patch is computed by the particle weights, it can be further employed to estimate the scale and location of the tracked target through a Hough Voting-like scheme. Therefore, the whole process ensures that we are tracking with the reliable patches related to the visual object, which is thus more robust than the conventional bounding box methods.

3.2. Patch-trackable Observation Likelihood

To estimate how likely a patch can be tracked effectively, we adopt the Peak-to-Sidelobe Ratio (PSR) as a confidence metric, which is widely used in signal processing to measure the signal peak strength in response map. Inspired by [8], we generalize the PSR to the template-based tracker

as a patch trackable confidence function:

$$s(\mathbf{X}) = \frac{\max\left(\mathbf{R}(\mathbf{X})\right) - \mu_\Phi(\mathbf{R}(\mathbf{X}))}{\sigma_\Phi(\mathbf{R}(\mathbf{X}))} \tag{7}$$

where $\mathbf{R}(\mathbf{X})$ is usually a response map. Φ is the sidelobe area around the peak which is 15% of the response map area in this paper. μ_Φ and σ_Φ are the mean value and standard deviation of $\mathbf{R}$ except area Φ, respectively. It can be easily observed that the function $s(\mathbf{X})$ becomes large when the response peak value is strong. Therefore, $s(\mathbf{X})$ can be treated as the confidence for a patch to measure whether it is tracked properly. Since Φ is proportional to the patch size, $s(\mathbf{X})$ can naturally handle scale variations. Figure 1 shows the distribution of the score function in an example image. It can be seen that there are some peak values in the score map which reveals the underlying visual structure of the image. In general, $\mathbf{R}(\mathbf{X})$ can be defined as:

$$\mathbf{R}_{i,j}(\mathbf{X}) \propto \frac{1}{d(\mathbf{T}, f(\mathbf{x} + u(i,j)))} \tag{8}$$

where $d(\mathbf{T}, f(\mathbf{x}))$ denotes the distance between the template $\mathbf{T}$ and the observation. $f(\mathbf{x})$ represents the feature extraction function, and $u(i,j)$ is the deviation of coordinate. Therefore our method is suitable for all template based trackers which have a template $\mathbf{T}$.

As the response value $\mathbf{R}(\mathbf{X})$ is inversely proportional to the distance between the template and sampled patch, the trackable score function $s(\mathbf{X})$ is compatible for most of base trackers, such as Lucas-Kanade method [3], normalized cross correlation and the Correlation Filter-based trackers (CFT) [14, 8, 33]. Due to the high efficiency and impressive performance achieved in the recent competition [29], we choose KCF [14] as our base tracker. Specifically, we directly employ the response map of correlation filters to facilitate the trackable score function. Therefore, trackable observation likelihood can be formulated as follows:

$$p_t(z_t|\mathbf{x}_t) \propto s(\mathbf{X}_t)^\lambda \tag{9}$$

where λ is a coefficient to trade off the contribution of the likelihood. We empirically set $\lambda = 2$ in this paper.

Figure 3 gives an example that plots the selected reliable patches. It can be seen that the proposed method prefers to choose patches around the headlight rather than those around the door as the headlight region has more visual sense compared against the flat door region. Therefore, our method tends to find the underlying structure of the visual target. Similar observations can be found in flock of tracker method [34], which employs a hand-crafted structural representation.

3.3. Patch-on-Object Observation Likelihood

To compute the probability of a patch lying on the tracked object, we exploit the motion information to achieve this goal. Specifically, we track both foreground and background patch particles, and record the relative trajectory for each patch: $\mathbf{V}_t = [\mathbf{v}_{t-k+1}^T, ..., \mathbf{v}_t^T]^T \in \mathcal{R}^{2k}$, where $\mathbf{v}_t = \Psi_2(\mathbf{x}_t - \mathbf{x}_{t-1})$ is the relative movement vector and $\Psi 2 = [\mathbf{E}_{2\times 2}, \mathbf{0}] \in \mathcal{R}^{2\times 4}$ is selective matrix to choose the position vector in the original state.

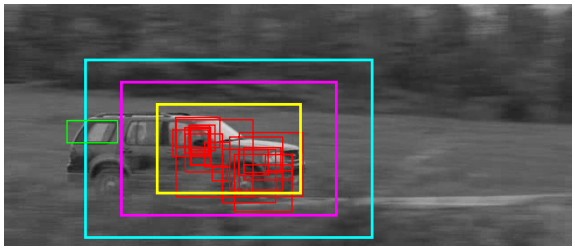


Figure 3. We label all patch particles in magenta box as positive otherwise as negative for motion analysis and delete all particles out of the cyan box.

Since the displacement of patch particles may correspond to different objects, we record k relative movement vectors to make the trajectory information to be more robust. We employ the l_2 norm to measure the distance between trajectories.

Instead of using computationally intensive unsupervised clustering methods such as agglomerative clustering [6] to group the trajectories, we simply divide the image into two regions by a rectangle box centering at the target. Then, we label the patches inside the bounding box as positive and those outside as negative. As shown in Figure 3, the yellow box denotes the bounding box. Therefore, we measure the similarity from a patch to its labelled group by formulating a similarity score function as:

$$l(\mathbf{X}) = y_t \left(\frac{1}{N^-} \sum_{j \in \Omega^-} ||\mathbf{V} - \mathbf{V}^{(j)}||_2 - \frac{1}{N^+} \sum_{i \in \Omega^+} ||\mathbf{V} - \mathbf{V}^{(i)}||_2 \right) \tag{10}$$

where $y_i \in \{+1, -1\}$ is the label to indicate whether $\mathbf{x}_i$ is in the yellow rectangle. Ω_t^+ is a set contains the indexes of the positive patch particles and Ω_t^- for negative ones. N^+ and N^- are size numbers responding to the sets, respectively.

The function $l(\mathbf{X})$ has the high score when the group samplers share the homo-motion and the other group has large motion difference while the negative score for those wrongly labelled samplers. When the motion trajectories between each group have no obvious distance, the function has a value close to zero. Thus, we can softly label each sampler again in the sampler set, and focus the patches' emphasis on the "objects". Thus, we formulate $l(\mathbf{X})$ to compute the probability of being on a visual object as:

$$p_o(z_t|\mathbf{x}_t) \propto \mathrm{e}^{\mu l(\mathbf{X}_t)}. \tag{11}$$

where μ is a coefficient to balance the contribution of object

probability. In this paper, we simply set $\mu = 1$. In case of no obvious motion between foreground and background, p_o is close to 1, which affects the observation model slightly.

Algorithm 1 RPT: the Reliable Patch Tracker algorithm

Require:
The model $\mathbf{M}_{t-1}$ and new arrived image $\mathbf{I}_t$
Ensure:
The updated Model $\mathbf{M}_t$ for tracked target;
The new target state, $\mathbf{x}_t^{target}$;

1: **for** every $\mathbf{X}_{t-1}^{(i)}$ in $\mathbf{M}_{t-1}$ **do**
2: Track $\mathbf{X}_{t-1}^{(i)}$ with the base tracker $\mathcal{T}_{t-1}^{(i)}$ in $\mathbf{I}_t$.
3: Update $\mathbf{x}_t^{(i)}$ and $\mathbf{V}_t^{(i)}$ in $\mathbf{X}_t^{(i)}$.
4: **end for**
5: Calculate the particle weights $\mathbf{W} = [w_t^{(1)}, ..., w_t^{(N)}]$ according to Equation 4.
6: Vote target's rough position $\hat{\mathbf{p}}_t$ according to Equation 13 with patch particle weights $\mathbf{W}$.
7: Resample the patch particles according to ***C1,C2,C3*** with the rough position $\hat{\mathbf{p}}$
8: Get target state $\mathbf{x}_t^{target}$ according to Equation 14.
9: **return** updated $\mathbf{M}_t$ and $\mathbf{x}_t^{target}$;

3.4. The Reliable Patch Tracker Algorithm

Based on the proposed reliable patch particle representation, we can estimate the target state through the statistical method [30, 26]. As in [30], we try to estimate the scale of the tracked object by storing the vector $\mathbf{d}_t^{(i)} = \mathbf{\Psi}_2(\mathbf{x}_t^{target} - \mathbf{x}_t^{(i)}) \in \mathcal{R}^2$ for each patch particle. Then, we calculate the set of each patch changes in scale:

$$D_t = \left\{ \frac{||\mathbf{r}^{i,j}||}{||\mathbf{d}^{i,j}||}, i \neq j \right\} \tag{12}$$

where $\mathbf{r}^{i,j} = \mathbf{\Psi}_2(\mathbf{x}_t^{(i)} - \mathbf{x}_t^{(j)})$ and $\mathbf{d}^{i,j} = \mathbf{d}_t^{(i)} - \mathbf{d}_t^{(j)}$. The scale is estimated by the median of this set $c_t = \text{med}(D_t)$. To make it more robust, a Gaussian filter is employed to smooth the output of target scale.

The object position can be estimated by the Hough-voting scheme [27]. Assuming that reliable patches are structurally consistent with the tracked object, we can treat the normalized particle weight $w_t^{(i)}$ as the confidence measure of tracking results. Thus, we employ all the positive patch particles to vote the center of the target as follows:

$$\mathbf{p}_t^{target} = \sum_{i \in \Omega^+} w_t^{(i)} (\mathbf{\Psi}_2 \mathbf{x}_t^{(i)} + c_t \mathbf{d}_t^{(i)}) \tag{13}$$

and the final state of the tracked object can be estimated by

$$\mathbf{x}_t^{target} = [\mathbf{p}_t^{target}, c_t \Psi(\mathbf{x}_{t-1}^{target})]. \tag{14}$$

As the false positive patches have quite small weights in the large motion gap, this makes the voting results very robust. When the motion change is not obvious, the position of a false positive patch is not far from the true position. As the number of patch particles increase, the estimation of the target state tends to be more accurate.

In contrast to the conventional methods [2] requiring to resample all the particles at each frame, we keep the patch particles from the previous frames and only recompute their weights. Moreover, we resample the particle only when it is necessary. In this paper, we define the following criteria to perform resample:

(C1) Far away from target. As shown in Figure 3, we define two regions with magenta color and cyan color respectively. The particles may become less important when they move too far away from the tracked object. Therefore, we simply remove the patch particles outside the cyan region. Meanwhile, we resample the positive particles outside the magenta region which may probably lead to some drift.

(C2) Imbalance between foreground and background particles. As the computational budget is fixed, we intend to keep the balance between the foreground and background patch particles to maintain the stability. Specifically, we resample the positive particles with low weights when the portion of positive particles is larger than a threshold γ. Similarly, we resample the negative particles when its portion becomes large.

(C3) Low trackable confidence. We resample the patch particles with the low trackable confidence, which usually occurs in the homogeneous region lacking of textures. This could potentially reduce the computational cost while improve the robustness.

During the re-sampling process, the base tracker of the patch particle will be re-initialized from time to time. Therefore, the new appearance will be learnt from the new patch particle. It is also important to note that we update each base tracker individually. From above all, we summarize the whole procedure of our proposed Reliable Patch Tracker (RPT) in Algorithm 1.

4. Experimental Results

In this section, we give details of our experimental implementation and discuss the results of tacking performance evaluation. We examine the effectiveness of the proposed approach on two datasets, including an online object tracking benchmark testbed [37] with 51 sequences and another extra dataset with 10 challenging videos. More experimental results are included in supplemental materials.

4.1. Experimental Setup

We implemented the proposed Reliable Patch Tracker (RPT) in Matlab using a single thread without further op-

timization. All the experiments were conducted on a regular PC with Intel-i7-3770 CPU (3.4 GHz) and 16 GB RAM. For the proposed RPT method, the yellow and cyan regions defined in Section 3.3 are set to 1.5 and 9 times the target bounding box, respectively. Moreover, we record the the trajectories of 5 consecutive frames for the motion analysis. During the tracking, we try to maintain the fixed 4:1 ratio between the number of positive particle patches and negative ones through the resample criteria C2. We use the default setting for the base KCF tracker [14] in RPT except having changed the padding parameter from 1.5 to 3 for better coverage.

To facilitate fair performance evaluations, two typical evaluation criteria are used in our experiments. The first criterion is mean Center Location Error (CEL), which is the pixel distance between the tracked results' center and the ground truth. The other is the Pascal VOC Overlap Ratio (VOR) [9], which is defined as $VOR = \frac{Area(\mathbf{B}_T \cap \mathbf{B}_G)}{Area(\mathbf{B}_T \cup \mathbf{B}_G)}$, where $\mathbf{B}$ denotes a bounding box and G, T indicate the ground truth and the tracking results, respectively.

4.2. Visual Benchmark

Besides 29 trackers in the tracking benchmark dataset [37], we have conducted experiments by comparing with another four recently proposed state-of-the-art trackers including KCF [14], CN [7], STC [39] and TGPR [16] to demonstrate the effectiveness of our proposed RPT approach. Note that we employ the original implementations of these trackers from the authors' websites with the default parameters.

For the visual tracking benchmark [37], the experimental results are illustrated by both precision plot and success plot. The precision plot shows the percentage of successfully tracked frames vs. the CEL in pixels, which ranks the trackers as precision score at 20 pixels. On the other hand, the success plot draws the percentage of successfully tracked frames vs. the VOR threshold, where Area Under the Curve (AUC) is used as metric for ranking. The benchmark covers 51 challenging sequences that is a variety of scenarios used in the previous literature.

To make it clear, we only plot the top 10 ranked trackers and the entire plots are included in supplemental materials. As shown in Figure 5, our proposed method ranks the first and achieves the best performance with a very large margin in all the ranking plots. Specifically, the proposed method achieves 0.576 ranking score in success plot and 0.812 ranking score as illustrated in Figure 5(a). Comparing with the base tracker KCF with 0.514 success ranking score and 0.740 precision ranking score, our method has obtained over 12% and 9.7% improvements, respectively. From Figure 5(b)- 5(h), the performance of different attributed groups indicates that our method is clearly more accurate and robust. Also, our method clearly outperforms those part-based trackers [17] [1] in the benchmark. This demonstrates that the idea of finding reliable patches for tracking is effective and promising in practice.

For better illustration, we further analyze the top 5 performed trackers in the following. As VOR is usually considered to be valid with the score larger than 0.5. Similarly, CEL is treated to be valid with the score smaller than 20. Therefore, we list the total number of sequences that fulfill these criteria in Table 1. It can be seen that the proposed tracker has successfully tracked 38 sequences based on the VOR threshold and passed 41 sequences based the CEL criteria, which account for around 80% of the total number of sequences in the whole benchmark. This demonstrates that tracking with reliable patches enables a base tracker to be capable of handling more challenging situations. Despite having achieved the encouraging performance for the proposed approach, our current implementation runs only around 4 Fps using the non-optimized single-thread MATLAB code. Note that the presented patch particle filter framework can easily extended to a parallel implementation and considerably optimized with more efficient implementation in C/C++. Figure 6 shows the snapshot of the sequences RPT passed with the patch particles visible.

Method	mVOR>0.5	mCEL<20	mFps.
Struck [11]	28	29	10.008
SCM [41]	28	28	0.374
KCF [14]	31	35	339
TGPR [16]	35	36	0.727
RPT	38	41	4.154

Table 1. List the numbers of sequences that the mean VOR is larger than 0.5 or the mean CEL is smaller than 20 for each tracker. The mean FPSs are also presented.

4.3. More Sequences

To further evaluate the performance of our proposed approach, we conduct the experiments with the top five trackers on the additional 10 challenging video sequences from previous studies [41, 17, 29]. Table 4.2 list the mean VOR and the mean CEL of each sequences. Figure 4 illustrates the tracking results of the 5 trackers.

Specifically, in *board*, the proposed tracker is able to track the board in background clutter even when the board turns back. This is mainly due to the motion information in the reliable patch. *panda* is cropped from cartoon video, which contains in plane rotation and a large blank region in background. Since our algorithm is capable of finding the trackable patch, the sequence is tracked perfectly while KCF drifts at the beginning and other trackers fail when the panda cross the tree. As these trackers are top ranked, they all succeed in *stone*, *polarbear* and *surfing*. SCM drifts in *sunshade* as the illumination changed dramatically. As *caviar3* has sever occlusion, all the trackers fails. Our method fails since the negative samples are not on

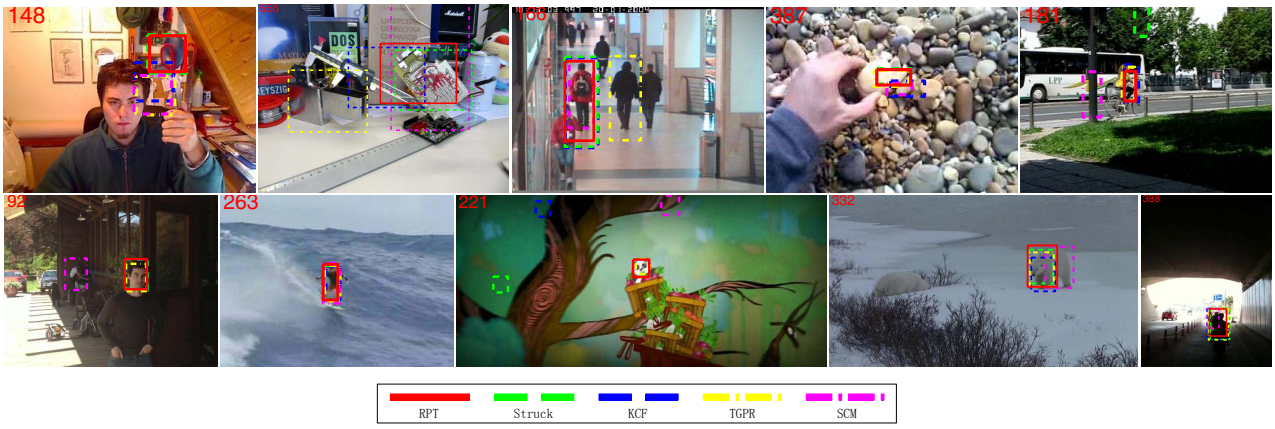
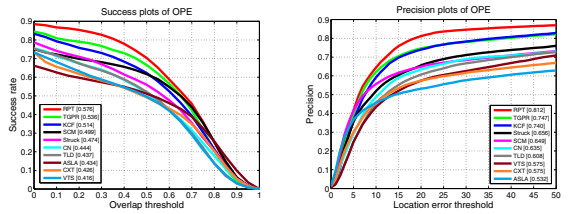
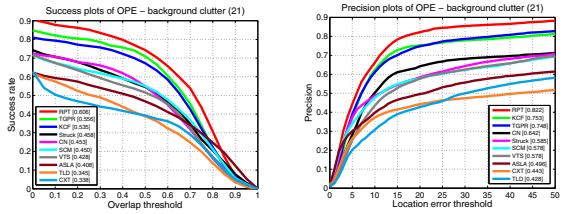


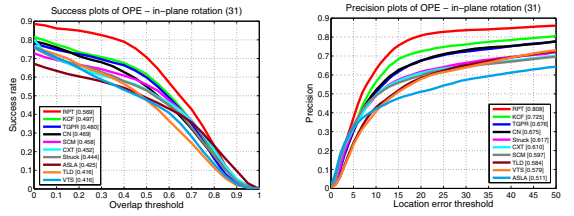
Figure 4. The extensive 10 sequences with top 5 trackers' tracking results



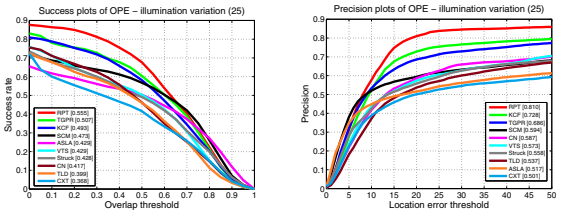
(a) Overall



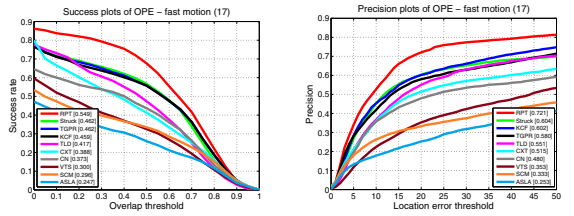
(b) Background Clutter



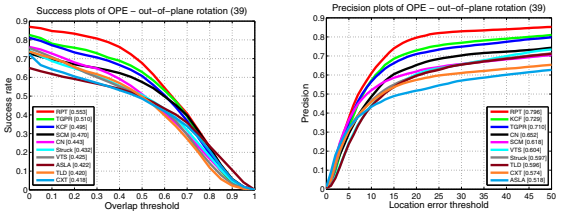
(c) In-plane Rotation



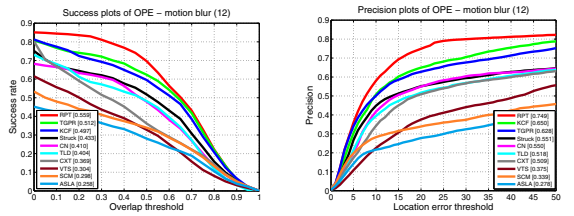
(d) Illumination Changes



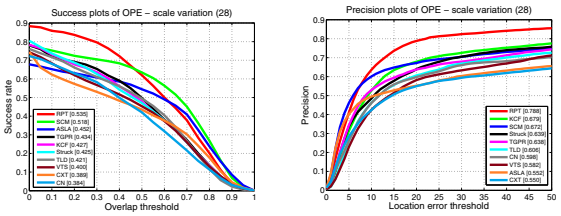
(e) Fast Moving



(f) Out-plane Rotation



(g) Motion Blur



(h) Scale Changes

Figure 5. The success and precision plots of the benchmark.

Figure 6. 20 sequences that the mean CEL of RPT is smaller than 20. The patch particles are also presented with the red boxes and green boxes which represent positive and negative particles respectively. (Better view in color.)

Table 2. The mean VOR and CEL of the 10 sequences for top 5 trackers

	mean VOR					mean CEL				
	RPT	TGPR [16]	KCF [14]	SCM [41]	Struck [11]	RPT	TGPR [16]	KCF [14]	SCM [41]	Struck [11]
board	0.786	0.092	0.803	0.317	0.769	18.7	242.3	17.3	86.3	28.4
caviar3	0.141	0.257	0.136	0.145	0.127	71.3	23.4	69.3	62.5	67.2
panda	0.624	0.677	0.020	0.253	0.358	3.8	2.4	150.1	94.1	90.7
stone	0.526	0.506	0.498	0.596	0.496	2.2	1.6	3.0	3.3	3.1
polarbear	0.718	0.631	0.647	0.715	0.611	9.9	9.5	9.5	9.3	12.2
sunshade	0.723	0.727	0.753	0.409	0.787	5.1	4.8	4.5	45.7	3.7
surfing	0.725	0.862	0.703	0.790	0.870	3.1	1.7	3.3	1.8	1.4
torus	0.803	0.756	0.787	0.472	0.158	3.4	7.1	4.0	30.1	56.0
bicycle	0.463	0.486	0.242	0.432	0.408	5.4	4.8	59.4	55.5	6.9
tunnel	0.512	0.324	0.324	0.618	0.324	4.4	8.6	6.3	7.7	10.5
averg.	0.602	0.532	0.491	0.475	0.491	12.7	30.6	32.7	39.6	28.0

the occluded person at the beginning. Struck and SCM drift in *torus* due to the out of plane rotation while our method can track the target very well. In *bicycle*, the scale of raider changes dramatically. KCF drifts when pass the wire pole. In *tunnel*, all tracker tracks the target successfully while the scale is not accurately estimated.

5. Conclusion

In this paper, we proposed a novel framework of Reliable Patch Trackers (RPT) which attempts to identify and exploit reliable patches for robust visual tracking. To efficiently find the reliable patches, we employed a particle filter-based method with two orthogonal properties, including the trackability and the motion similarity to estimate the distribution of those reliable patches. After finding the reliable patches, we tracked those patches with some effective base tracker and then employed the reliable patch particles to represent the visual target. In addition, an efficient updating scheme was carefully designed to enable the algorithm for running online. By tracking with those more meaningful and reliable patches, the proposed tracker can thus handle more diverse and challenging situations of visual tracking. Finally, we obtained encouraging empirical performance from our extensive experiments by comparing the proposed tracker with several state-of-the-art trackers.

Acknowledgments

The work was supported by National Natural Science Foundation of China under Grants (61103105 and 91120302).

References

[1] A. Adam, E. Rivlin, and I. Shimshoni. Robust fragments-based tracking using the integral histogram. In *CVPR*, 2006.

[2] A.Doucet, N. Freitas, and N.Gordon. *Sequential Monte Carlo Methods in Practice*. Springer-Verlag, 2001.

[3] B.D.Lucas and T.Kanade. An iterative image registration technique with an application to stereo vision. In *Proccedings of Imageing Understanding Workshop*, 1981.

[4] B.Poling, G.Lerman, and A.Szlarm. Better feature tracking though subspace constraints. In *CVPR*, 2014.

[5] Z. Cai, L. Wen, J. Yang, Z. Lei, and S. Li. Structured visual tracking with dynamic graph. In *ACCV*, 2012.

[6] C.D.Manning, P.Raghavan, and H. Schutze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[7] M. Danelljan, F. S. Khan, M. Felsberg, and J. van de Weijer. Adaptive color attributes for real-time visual tracking. In *CVPR*, 2014.

[8] D.S.Bolme, J.R.Beveridge, B.A.Draper, and Y. M. Lui. Visual object tracking using adaptive correlation filters. In *CVPR*, 2010.

[9] M. Everingham, L. Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classese(voc) challenge. *IJCV*, 88(2):303–338, 2010.

[10] M. Grundmann, V. Kwatra, M. Han, and I. Essa. Efficient hierarchical graph based video segmentation. In *CVPR*, 2010.

[11] S. Hare, A. Saffari, and P. H. S. Torr. Struck: Structured output tracking with kernels. In *ICCV*, 2011.

[12] S. He, Q.-X. Yang, R. Lau, J. Wang, and M.-H. Yang. Visual tracking via locality sensitive histograms. In *CVPR*, 2013.

[13] F. Henriques, R. Caseiro, P. Martins, and J. Batista. Exploiting the circulant structure of tracking-by-detection with kernels. In *ECCV*, 2012.

[14] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. High-speed tracking with kernelized correlation filters. *TPAMI*, 2015.

[15] S. Hinterstoisser, V. Lepetit, S. Ilic, P. Fua, and N. Navab. Dominant orientation templates for real-time detection of texture-less objects. In *ICCV*, 11.

[16] J.Gao, H.Ling, W.Hu, and J.Xing. Transfer learning based visual tracking with gaussian process regression. In *ECCV*, 2014.

[17] X. Jia, H. Lu, and M.-H. Yang. Visual tracking via adaptive structural local sparse appearance model. In *CVPR*, pages 1822–1829, Providence, June 2012.

[18] M. J.Milford and Gordon.F.Wyyeth. Seqslam: Visual route-based navigation for sunny summer days and stormy winter nights. In *IEEE International Conference on Robotics and Automation*, 2012.

[19] G. Klein and D. Murray. Vparallel tracking and mapping for small ar workspaces. In *ISMAR*, 2007.

[20] J. Kwon and K. M. Lee. Visual tracking decomposition. In *CVPR*, 2010.

[21] J. Kwon and K. M. Lee. Tracking by sampling trackers. In *ICCV*, 2011.

[22] J. Kwon and K. M. Lee. Highly nonrigid object tracking via patch-based dynamic appearance modeling. *TPAMI*, 35(10):2427–2441, 2013.

[23] Y. Li and J. Zhu. A scale adaptive kernel correlation filter tracker with feature integration. In *ECCV Workshops*, pages 254–265, 2014.

[24] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.

[25] X. Mei and H. Ling. Robust visual tracking using l1 minimization. In *ICCV*, 2009.

[26] M.E.Maresca and A.Petrosino. Matrioska: A multi-level approach to fast tracking by learning. In *ICIAP*, 2013.

[27] M.Godec, P.M.Roth, and H.Bischof. Hough-based traking of non-rigid objects. In *ICCV*, 2011.

[28] M.Grabner, H.Grabner, and H.Bischof. Learning features for tracking. In *CVPR*, 2007.

[29] M.Kristan, R.Pflugfelder, and A. et al. The visual object tracking vot2014 challenge results. In *ECCV Workshops*, 2014.

[30] G. Nebehay and R. Pflugfelder. Consensus-based matching and tracking of keypoints. In *Winter Conference on Applications of Computer Visio*, 2014.

[31] D. Ross, J. Lim, R.-S. Lin, and M.-H. Yang. Incremental learning for robust visual tracking. *IJCV*, 77(1-3):125–141, 2008.

[32] S.Hare, A.Saffari, and P.H.S.Torr. Efficient online structured output learning for keypoing-based object tracking. In *CVPR*, 2012.

[33] J. van de Weijer, C. Schmid, J. J. Verbeek, and D. Larlus. Learning color names for real-world applications. *TIP*, 18(7):15121524, 2009.

[34] T. Vojir and J. Matas. The enhanced flock of trackers. In *Registration and Recognition in Images and Videos*, 2014.

[35] D. Wang, H. Lu, and M.-H. Yang. Least soft-thresold squares tracking. In *CVPR*, Portland, June 2013.

[36] C. Wu, J. Zhu, J. Zhang, C. Chen, and D. Cai. A convolutional treelets binary feature approach to fast keypoint recognition. In *ECCV*, pages 368–382, 2012.

[37] Y. Wu, J. Lim, and M.-H. Yang. Online object tracking: A benchmark. In *CVPR*, 2013.

[38] J. Zhang, S. Ma, and S. Sclaroff. Meem: Robust tracking via multiple experts using entropy minimization. In *ECCV*, 2014.

[39] K. Zhang, L. Zhang, M.-H. Yang, and D. Zhang. Fast tracking via spatio-temporal context learning. In *ECCV*, 2014.

[40] L. Zhang and L. van der Maaten. Preserving structure in model-free tracking. *IEEE Transactions on Pattern Recognition and Machine Intelligence*, 36(4):756–769, 2014.

[41] W. Zhong, H. Lu, and M.-H. Yang. Robust object tracking via sparsity-based collaborative model. In *CVPR*, pages 1838–1845, Providence, June 2012.

[42] J. Zhu, S. C. H. Hoi, and M. R. Lyu. Nonrigid shape recovery by gaussian process regression. In *CVPR*, pages 1319–1326, 2009.

[43] J. Zhu and M. R. Lyu. Progressive finite newton approach to real-time nonrigid surface detection. In *CVPR*, 2007.