Tracking Using Multilevel Quantizations

Zhibin Hong[1], Chaohui Wang[2], Xue Mei[3],
Danil Prokhorov[3], and Dacheng Tao[1]

[1] Centre for Quantum Computation and Intelligent Systems,
Faculty of Engineering and Information Technology,
University of Technology, Sydney, NSW, Australia
[2] Max Planck Institute for Intelligent Systems, Tübingen, Germany
[3] Toyota Research Institute, North America, Ann Arbor, MI, USA

Abstract. Most object tracking methods only exploit a single quantization of an image space: pixels, superpixels, or bounding boxes, each of which has advantages and disadvantages. It is highly unlikely that a common optimal quantization level, suitable for tracking all objects in all environments, exists. We therefore propose a hierarchical appearance representation model for tracking, based on a graphical model that exploits shared information across multiple quantization levels. The tracker aims to find the most possible position of the target by jointly classifying the pixels and superpixels and obtaining the best configuration across all levels. The motion of the bounding box is taken into consideration, while Online Random Forests are used to provide pixel- and superpixel-level quantizations and progressively updated on-the-fly. By appropriately considering the multilevel quantizations, our tracker exhibits not only excellent performance in non-rigid object deformation handling, but also its robustness to occlusions. A quantitative evaluation is conducted on two benchmark datasets: a non-rigid object tracking dataset (11 sequences) and the CVPR2013 tracking benchmark (50 sequences). Experimental results show that our tracker overcomes various tracking challenges and is superior to a number of other popular tracking methods.

Keywords: Tracking, Multilevel Quantizations, Online Random Forests, Non-rigid Object Tracking, Conditional Random Fields.

1 Introduction

Online object tracking is a classic topic in computer vision and is used in many practical applications, such as video surveillance and autonomous driving. Given the position of the target in one frame, a tracker should be able to track the target in subsequent frames and be able to overcome various challenges, such as appearance variations, occlusions and illumination changes. Building an effective tracker is therefore extremely difficult, especially without prior knowledge of the appearance of the object to be tracked. However, a number of trackers have been proposed and show promising results [39,54,55].

D. Fleet et al. (Eds.): ECCV 2014, Part VI, LNCS 8694, pp. 155–171, 2014.

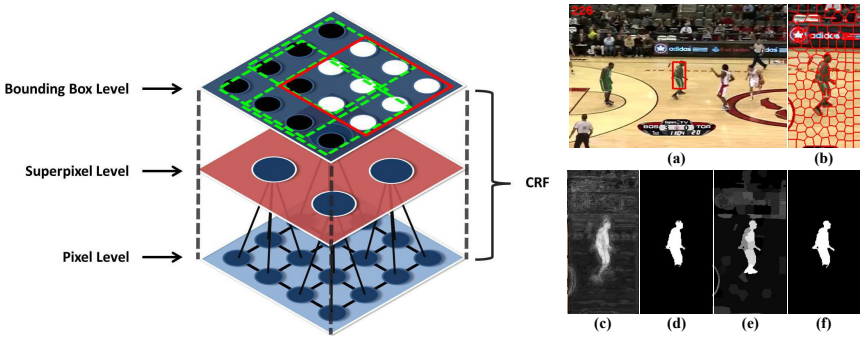

Fig. 1. Illustration of the structure of the proposed hierarchical appearance representation model (left) and a practical example (right). In the proposed framework, a node in the Conditional Random Field (CRF) models each pixel, superpixel, and bounding box. At the pixel level, each pixel receives a measurement from a Random Forest and connects to the corresponding superpixel at the middle level. At the superpixel level, each superpixel also obtains a probability output by another Random Forest and suggests the pixels within the same superpixel to share the same label. At the bounding box level, different candidate bounding boxes (green) are considered, and the best position (red) with the best configuration is found. (a) shows the tracking result (in red bounding box) at Frame #226 in the *Basketball* sequence. (b) displays the superpixelization of the image. (c) and (d) are the output of the pixel-level RF and final labeling result, respectively, while (e) and (f) are the output of the superpixel-level RF and final labeling result.

Many tracking methods operate by making a single quantization choice in an image space, i.e., using pixels [14], superpixels [52], or bounding boxes [46], each of which has its pros and cons. For example, a tracker built on pixel-level quantization may be able to capture and thus better handle non-rigid deformation, but performs relatively poorly in scenarios where there is excessive background clutter due to the lack of holistic appearance of the target. In contrast, trackers that utilize higher-level quantization, such as bounding box-based matching, are robust to occlusions but tend to fail when the target undergoes non-rigid deformation. Therefore, a single optimal quantization level suitable for all objects in all environments is unlikely to exist.

Motivated by the above observation, in this paper, we propose a novel hierarchical appearance representation formulation of object tracking based on Conditional Random Fields (CRFs), which unifies multiple disparate quantizations of the image space. Based on the information derived from different quantization levels (pixel, superpixel, bounding box), we integrate them into a principled framework to optimize the decision-making. At the lowest level, an Online Random Forest (ORF) [48] equipped with color-texture features is employed to provide a soft label of each pixel, which indicates the probability that the pixel belongs to the target. At the middle level, superpixels are generated by considering various cues such as the spatial relationship and feature similarity

between pixels, which suggests a consistent pixel labeling within each superpixel. Besides, another ORF based on normalized histogram features of superpixels is also trained on the mid-level quantization. At the highest level, a bounding box-level regularization term is introduced, which enables to flexibly incorporate other information of a given bounding box, such as shape and motion, or even the measurement given by other trackers. The model bridges the hierarchical appearance representations by fusing multilevel quantization information and efficiently solves the optimization with the use of dynamic graph cuts [27]. However, the contribution of this paper is not limited to the application of the novel hierarchical appearance representation framework to object tracking. We also address appearance variations by exploiting color-texture features and powerful, yet efficient, ORFs. These ORFs are strategically updated in the framework to capture appearance changes due to deformation or illumination over time. The proposed method is illustrated in Fig. 1.

2 Related Work

There has been recent progress in visual object tracking research, with several ideas that focus on various challenges being proposed; these are extensively discussed in elsewhere [39,55]. In addition, standard benchmark datasets and quantitative evaluation metrics have been developed [15,54] to facilitate research in this area.

Some existing approaches use the pixel level of the image space to explore low-level cues for tracking. For instance, Avidan [4] proposed an ensemble tracker to track the object based on the result of the pixel-level classification. Although the discriminative setting enables the tracker to distinguish foreground from background, the pixel-based representation still limits robustness to a cluttered background and heavy occlusion [52]. More recently, Duffner and Garcia proposed the PixelTrack [14], which also addresses tracking at the pixel-level. The tracker works by combining a Hough voting-based detector with a soft segmentation approach similar to [3]. Although an efficient implementation is achieved, the tracker appears to be sensitive to the initialization and is prone to fail in grayscale sequences due to the dependence on the segmentation performance and the lack of global information.

Compared to pixel-level representations, mid-level visual cues provide more information about the local structure of images, while retaining the flexibility to model non-rigid deformation [2,11,28,52]. In particular, Adam *et al.* [2] employed an appearance model and used local patches to handle partial occlusion. Superpixel tracking [52] aims to explore the mid-level cues and use the superpixel as the object representation. The normalized color histogram of each superpixel is extracted, and a confidence map is obtained by the superpixel-level, rather than pixel-level, classification [4]. In [11], the target is represented by a set of different regions. The regions are modeled by a Gaussian mixture model in a joint feature-spatial space, and the motion of the target is modeled by the level set evolution.

Many trackers are built to exploit high-level visual information using holistic appearance models [5,19,41,46]. In [46], Ross *et al.* presented a tracking method that incrementally learns a low-dimensional subspace representation of the target. The L1 tracker, proposed by Mei *et al.* [41], and its variations [21,38,42,56] appear to be robust to the illumination changes and occlusions but sensitive to non-rigid deformation. Babenko *et al.* [5] employ Multiple Instance Learning (MIL) to overcome the label ambiguity problem, in which the training samples are collected as bag of image patches. Random Forests (RFs) [9] have become increasingly popular in computer vision due to their attractive properties, and have been used in tracking [25,48,49]. Specifically, Saffari *et. al* [48] proposed an online version of RFs, which grows extremely randomized trees online, rather than offline. The Online RFs (ORFs) are then adopted for tracking by utilizing the features captured at the bounding box level, and this method has demonstrated better performance over the online boosting [16].

Multilevel data fusing has been exploited for image segmentation and labeling using Conditional Random Fields (CRFs) or Markov Random Fields (MRFs) [18,22,31,50,53]. In [31], a single optimization framework is presented in which a hierarchical random field model allows integration of features computed at different levels of the quantization hierarchy. In [18], labeling information from local image statistics, regional label features, and global label features are combined in order to label images with a predefined set of class labels. In [22], a hierarchical two-stage CRF model is used to combine parametric and nonparametric image labeling methods. In [32,53], integration of object detection and pixel-wise scene labeling boosts the performance of both tasks, since they are mutually beneficial. Also, previous works have addressed tracking problems by combining multilevel information [26,51,57]. For example, in [57], a collaborative model is proposed to combine a Sparsity-based Discriminative Classifier (SDC) with Sparsity based Generative Model (SGM), which collaboratively considers holistic object templates and local image patches for target representations. Motivated by these advances, here we propose the Multilevel Quantization Tracker (MQT), which explores the quantization hierarchy from coarse to fine and unifies the information derived from multiple quantization levels in a coherent CRF framework. In this way, each quantization level benefits from other levels and, as a consequence, the overall performance of each individual level is enhanced.

3 Tracking with Multilevel Quantizations

The proposed tracker combines multilevel quantizations as a single graphical model to produce an efficient and robust solution to online object tracking. We first introduce the general multilevel quantization model and then describe other important components of the tracker, including feature extraction, online color-texture forests, ORF training, and occlusion handling.

3.1 Multilevel Quantizations Model

The whole model is built on multiple quantizations from three hierarchical appearance representation levels, namely pixels, superpixels and bounding boxes. We first extract information at each level before fusing them using a graphical model so as to perform inference.

Pixel is the finest quantization level in an image, and is the most obvious choice for quantization. Let each pixel $i \in \mathcal{P}$ ($\mathcal{P}$ denotes the set of pixels) be represented as a d-dimensional feature vector $\mathbf{f}_i \in \mathbb{R}^d$ that consists of some local information, and associated with a unique binary label $x_i \in \{0$ (background), 1 (foreground/object)$\}$. The pixel-level unary energy function is defined as:

$$\phi_i^{\mathrm{p}}(x_i) = -\log p(x_i; \mathbf{H}^{\mathrm{p}}), \tag{1}$$

where $p(x_i; \mathbf{H}^{\mathrm{p}})$ denotes the probability that pixel i is labeled as class x_i, output by an ORF with parameters $\mathbf{H}^{\mathrm{p}}$, which are updated online (Section 3.3). An example of $p(x_i; \mathbf{H}^{\mathrm{p}})$ output by an ORF is shown in Fig. 1(c).

Superpixels provide very useful mid-level support for image understanding tasks (e.g., [1,37]). In order to exploit mid-level information, we employ the SLIC (Simple Linear Iterative Clustering) algorithm [1] to cluster the pixels and generate superpixels as shown in Fig. 1(b). For each superpixel $k \in \mathcal{S}$ ($\mathcal{S}$ denotes the set of superpixels), we also assign a binary label $y_k \in \{0, 1\}$, which is similar to x_i at pixel level. Again, an ORF is trained to output the probability that the superpixel belongs to the foreground or background, using the features extracted from each superpixel (Fig. 1(e)). Similarly, superpixel-level energy function is defined as:

$$\phi_k^{\mathrm{s}}(y_k) = -\log p(y_k; \mathbf{H}^{\mathrm{s}}), \tag{2}$$

where the symbols are analogous to those in (1).

At the highest level, like many existing online trackers [46,17], we use a bounding box to delimit the object of interest. Let $\mathbf{B}(z)$ denote the bounding box with pose parameters z and energy function $\varphi(\mathbf{B}(z))$ encode the occurrence likelihood of the target in bounding box $\mathbf{B}(z)$. In contrast to other online trackers (e.g., [19,41,46]) which optimize merely $\varphi(\mathbf{B}(z))$ to get the tracking solution, we unify $\varphi(\mathbf{B}(z))$ with information from the other quantization levels, as explained above. The choice of $\varphi(\mathbf{B}(z))$ is modular and it can vary from simple matching techniques [10] to sophisticated classification models [17].

In our experiments, we use the Median Flow Tracker (MFT) [24] for the bounding box level quantization. MFT uses the feature matching to estimate the motion of target. Moreover, it measures the discrepancies of the forward and backward tracking in consecutive frames and reports failure when the target is lost [25]. We assign 0 to the tracking result z^M if failure is detected. The bounding box energy function $\varphi(\mathbf{B}(z))$ is defined as:

$$\varphi(\mathbf{B}(z)) = \begin{cases} 0 & , z^M = 0 \\ D^2(\mathbf{B}(z), \mathbf{B}(z^M)), & \text{otherwise} \end{cases} \tag{3}$$

where $D(\mathbf{B}(z), \mathbf{B}(z^M))$ is the distance between the centers of two bounding boxes $\mathbf{B}(z)$ and $\mathbf{B}(z^M)$ (the results of MFT) in the image.

Given the above three levels, we adopt a Conditional Random Field (CRF) model to fuse the information from different levels. Each unit (pixel, superpixel, bounding box) at different levels is represented by a node in the graph, and use corresponding unary potential functions to encode those terms in (1), (2), and (3). Then we capture the interactions between these nodes via connecting them using CRF's edges with appropriate potential functions.

Firstly, we associate an edge between a pair of neighboring pixel nodes (4-neighborhood system is considered in the experiment) and the following potential function to encode the interaction between the labeling of the pixels:

$$\psi_{i,j}(x_i, x_j) = \begin{cases} \exp(-\frac{\|\mathbf{f}_i - \mathbf{f}_j\|^2}{\sigma^2}), \text{ if } \ x_i \neq x_j \\ 0 \qquad\qquad\qquad\quad , \text{otherwise} \end{cases} \tag{4}$$

where $\|\mathbf{f}_i - \mathbf{f}_j\|$ is the distance between x_i and x_j in the feature space, and σ is a parameter controlling the shape of the monotonically decreasing function, which is similar to [7]. We use $\mathcal{E}^{\text{pp}}$ to denote all such edges between neighboring pixels.

One important fact regarding the pixel- and superpixel-level quantizations is that the pixels in the same superpixel tend to share the same superpixel label. Hence, for each pixel i in superpixel k, we associate an edge using the Potts model as its potential function:

$$\xi_{i,k}(x_i, y_k) = \begin{cases} 1, \text{ if } \ x_i \neq y_k \\ 0, \text{ otherwise} \end{cases} \tag{5}$$

which penalizes the inconsistency in labeling between superpixels and pixels. We use $\mathcal{E}^{\text{sp}}$ to denote all such edges.

We also connect all pixel nodes with the bounding box node. The pairwise potential function $w_i(z, x_i)$ is used to encourage consistency between pixel labeling and the pose of the bounding box:

$$w_i(z, x_i) = \begin{cases} d(z,i), \text{ if } (x_i = 1,\, i \in \mathcal{P}^{\text{Out}}_{\mathbf{B}(z)}) \text{ or } (x_i = 0,\, i \in \mathcal{P}^{\text{In}}_{\mathbf{B}(z)}) \\ 0 \qquad\quad , \text{otherwise} \end{cases} \tag{6}$$

where $d(z, i)$ represents the minimum normalized distance (which considers the size of bounding box and is detailed in Section 4.1) between the pixel i to the boundary of the bounding box $\mathbf{B}(z)$; $\mathcal{P}^{\text{in}}_{\mathbf{B}(z)}$ and $\mathcal{P}^{\text{Out}}_{\mathbf{B}(z)}$ denote the set of pixels inside and outside the bounding boxes, respectively. The choice of function is based on the observation that the pixels inside the bounding box tend to belong to the object, while the pixels outside the bounding box tend to belong to the background. Moreover, the closer the pixel is to the boundary, the more ambiguous the pixel label is. The pixel is penalized for having different label from what is expected, using a cost that is proportional to the distance between the pixel and the boundary of the bounding box, which is similar to the idea in [51].

Finally, given an image I, the joint probability of the realization $(z, \mathbf{x}, \mathbf{y}) = (z, \mathbf{x} = (x_i)_{i\in\mathcal{P}}, \mathbf{y} = (y_k)_{k\in\mathcal{S}})$ of all random variables in the CRF model is formulated as a Gibbs distribution $P(z, \mathbf{x}, \mathbf{y}|I) = e^{-E(z,\mathbf{x},\mathbf{y})}$. The corresponding

Gibbs energy $E(z, \mathbf{x}, \mathbf{y})$ is defined as the sum of the above unary potentials and pairwise potentials:

$$\begin{aligned} E(z, \mathbf{x}, \mathbf{y}) =& \mu\varphi(\mathbf{B}(z)) + \sum_{i \in \mathcal{P}} \phi_i^{\mathrm{p}}(x_i) + \alpha \sum_{k \in \mathcal{S}} \phi_k^{\mathrm{s}}(y_k) + \lambda \sum_{i \in \mathcal{P}} \omega_i(x_i, z) \\ &+ \beta \sum_{\{i,k\} \in \mathcal{E}^{\mathrm{sp}}} \xi_{i,k}(x_i, y_k) + \gamma \sum_{\{i,j\} \in \mathcal{E}^{\mathrm{pp}}} \psi_{i,j}(x_i, x_j), \end{aligned} \tag{7}$$

where μ, α, λ, β, γ are the weight coefficients which balance the importance of each potential term.

In the tracking problem, we aim to determine the optimal pose parameters z for the bounding box. Since the minimization of $E(z, \mathbf{x}, \mathbf{y})$ with respect to $\mathbf{x}$ and $\mathbf{y}$ can be efficiently solved using the well-known graph cuts [8] for each possible z, we define an auxiliary function $\hat{E}(z)$ and search for the optimal z^* for $\hat{E}(z)$ using any off-the-shelf optimization algorithm:

$$z^* = \underset{z}{\operatorname{argmin}} \{\hat{E}(z) = \min_{\mathbf{x} \in \{0,1\}^{|\mathcal{P}|}, \mathbf{y} \in \{0,1\}^{|\mathcal{S}|}} E(z, \mathbf{x}, \mathbf{y})\}. \tag{8}$$

For example, one can use the local dense sampling search as done in [5,17]. In this paper, we simply adopt the Nelder-Mead Simplex Method [33] to directly search for the solution. Note that during the search of z in the problem (8), the update of z only causes small change[1] in ω_i, which motivates the use of dynamic MRF algorithms [26] (e.g., dynamic graph cuts [27]) to efficiently obtain the value of $\hat{E}(z)$ and significantly accelerate the optimization.

3.2 Online Color-Texture Forests

The selection of features and an appropriate online learning process has been shown to be very important for tracker performance [4,20,46,48]. In this section, we elaborate online color-texture forests, which are used to obtain pixel- and superpixel-level potentials in (1) and (2).

The color feature is one of the most widely used visual features in tracking. The most important advantages of color feature are power of representing visual content of images, simplicity in extracting color information of images and high efficiency, independent of image size and orientation. However, only using color feature is difficult to tackle many real-world tracking scenarios, such as distractive background clutter and drastic illumination change. We combine texture with color as a complementary feature for tracking to better represent object appearance. For each pixel, we extract RGB (3-dim), CIELAB (3-dim) and texture features (48-dim) as the pixel-level representation with 54 dimensions. The texture feature is generated by the Leung-Malik (LM) Filter Bank [36], which consists of the first and second derivatives of Gaussians at 6 orientations and 3 scales, 8 Laplacian of Gaussian (LOG) filters, and 4 Gaussian filters. With respect to the superpixel level, we utilize normalized histogram-based features to

[1] $\mu\varphi(\mathbf{B}(z))$ would change but would not affect the optimum of $E(z, \mathbf{x}, \mathbf{y})$ with respect to $\mathbf{x}$ and $\mathbf{y}$.

capture the photometric properties of each superpixel, similar to [52].We extract a 64-bin normalized histogram in the HSV color space and a 10-bin normalized histogram based on uniform rotation-invariant local binary patterns (LBPs) [43], to form a 74 dimensional color-texture feature for each superpixel.

Random forests consist of a set of randomized decision trees. In each decision tree, an internal node corresponds to a random test on an input feature, which determines to which child node the feature should go. Therefore, a feature vector is presented to the root of a tree and it follows a specific path to a leaf node, which stores a histogram (occurrence frequency of each class) obtained during the training phase. Given a test sample $\mathbf{f}$, the probability is estimated by averaging the probabilities of all the trees:

$$p(\text{class} = c|\mathbf{f}) = \frac{1}{N}\sum_{n=1}^{N} p_n(\text{class} = c|\mathbf{f}),$$

where N denotes the number of the trees, and $p_n(\text{class} = c|\mathbf{f})$ is the probability that the feature belongs to class c output by the tree n.

RFs have demonstrated great promise in various computer vision tasks including object recognition [35] and image classification [6]. We adopt the Online Random Forests [48] to incorporate the high-dimensional color-texture feature for our online tracking. The resulting online color-texture forest turns out to provide very discriminative classification results for our potential functions.

3.3 ORF Training and Occlusion Handling

To train the two RFs for pixels and superpixels, a key issue is how to get positive and negative samples for training. In the first frame, given the target bounding box, Grabcut [47] is adopted to automatically determine the pixels corresponding to the object which are then used as positive examples for training the RF for pixels. This generally improves the accuracy compared to treating all pixel inside the bounding box as foreground, since the object may not occupy the whole bounding box due to its shape. To deal with cases that object is not well segmented by Grabcut, we check the percentage of pixels with foreground labels in the bounding box. If it is greater than 70%, the result of Grabcut is accepted, otherwise it is rejected and all the pixels inside the bounding box are used as the positive samples. On the other hand, all the pixels outside the bounding box are used as negative samples. For superpixels, they are labeled using a voting scheme, i.e., the label of the superpixel is decided by the majority of the pixels inside it.

During the tracking, the ORFs are progressively updated to handle the appearance changes. Since pixels and superpixels are labeled in the whole formulation by jointly exploiting the information from multiple levels during the tracking, we only treat the pixels and superpixels as candidate positive samples if they are inside the target bounding box $\mathbf{B}(z^*)$ and labeled as positive by our tracker using (8). The pixels and superpixels outside the bounding box are treated as candidate negative samples. Moreover, only the candidate samples

Algorithm 1. Tracking with Multilevel Quantizations

Input: The target bounding box $\mathbf{B}(z_1^*)$ in the first frame; T frames to track.
Output: Estimated target position $\mathbf{B}(z_t^*)$, where $t = 2, 3..., T$ is the frame index.

```
/*Initialization*/
Apply Grabcut[47] to find the positive and negative samples.
Train pixel- and superpixel-level RFs using the collected samples.
/*Start to track*/
for t = 2 to T do
    /*Pixel level*/
    Extract features for each pixel i and obtain the pixel-level measurement p(x_i; H^p).
    /*Superpixel level*/
    Apply SLIC [1] to generate superpixels.
    Extract features for each superpixel k and obtain the superpixel-level measurement p(y_k; H^s).
    /*Bounding box level and combine multilevel quantizations*/
    Estimate the motion of target using MFT [24] and obtain B(z_t^M).
    Find the target B(z_t^*) by solving (8) using [33] with dynamic graph cuts [27].
    if not occluded then
        Update H^p of the pixel-level RF using X_p^+, X_p^-.
        Update H^s of the superpixel-level RF using X_sp^+, X_sp^-.
    end if
end for
```

that are not classified with a high confidence or incorrectly classified by their respective RFs are assigned to RFs for updates. A similar strategy is employed in [25]. More specifically, the final positive sample set $\mathcal{X}_p^+$ and negative sample set $\mathcal{X}_p^-$ for the pixel-level RF update are respectively determined as:

$$\mathcal{X}_p^+ = \{i | x_i = 1, p(x_i = 1; \mathbf{H}^{\mathrm{p}}) < \varepsilon_p^+, i \in \mathcal{P}^{\mathrm{In}}_{\mathbf{B}(z^*)}\}, \tag{9}$$

$$\mathcal{X}_p^- = \{i | p(x_i = 1; \mathbf{H}^{\mathrm{p}}) > \varepsilon_p^-, i \in \mathcal{P}^{\mathrm{Out}}_{\mathbf{B}(z^*)}\}, \tag{10}$$

where ε_p^+, ε_p^- (and ε_{sp}^+, ε_{sp}^- below) are the predefined thresholds. For the superpixel-level RF, the positive sample set $\mathcal{X}_{sp}^+$ and negative sample set $\mathcal{X}_{sp}^-$ are similarly determined as

$$\mathcal{X}_{sp}^+ = \{k | y_k = 1, p(y_k = 1; \mathbf{H}^{\mathrm{s}}) < \varepsilon_{sp}^+, k \in \mathcal{S}^{\mathrm{In}}_{\mathbf{B}(z^*)}\} \ , \tag{11}$$

$$\mathcal{X}_{sp}^- = \{k | p(y_k = 1; \mathbf{H}^{\mathrm{s}}) > \varepsilon_{sp}^-, k \in \mathcal{S}^{\mathrm{Out}}_{\mathbf{B}(z^*)}\} \ , \tag{12}$$

where $\mathcal{S}^{\mathrm{in}}_{\mathbf{B}(z^*)}$ and $\mathcal{S}^{\mathrm{Out}}_{\mathbf{B}(z^*)}$ denote the set of superpixels inside and outside the bounding box $\mathbf{B}(z^*)$, respectively. Noted that in (11) and (12), the voting scheme previously presented is still used to determine whether a superpixel is inside or outside the bounding box.

Fig. 2. Occlusion handling on the *Jogging* sequence. The index is specified in the top-left of each frame, and the two figures between each frame are the corresponding outputs of the pixel-level RFs and labels x_i, respectively. The occlusion is detected from the Frame #049, from which point the RFs stop updating until the target moves out of occlusion.

As discussed in previous works [4,52], it is also important to take occlusions into account during updates, especially when the target is temporarily out-of-view. The pixel labeling provided by our approach also can be used to handle occlusions: a flag of occlusion is trigged if the percentage of foreground pixels inside the found bounding box is less than a predefined threshold θ (0.3). In this case, the RFs are kept unchanged without update. An example of the occlusion handling is shown in Fig. 2. Finally, a systematic view of the whole algorithm is summarized in Algorithm 1.

4 Experiments

In this section, we first present implementation details about important aspects of MQT, including the parameter setting for evaluation. We then present a set of qualitative experimental results as well as quantitative comparison with several state-of-the-art trackers on two benchmarks.

4.1 Implementation Details

Similar to [4,52], the optimal tracking result at each frame is achieved by searching in a region centered around the target bounding box determined from the previous frame, as illustrated in Fig. 1(b). As in [5,17,19], we use a bounding box with a fixed size during the tracking in the current implementation. In order to track objects with different resolutions using the same parameters, we resize the image and let the short side of the target bounding box in the first frame to have a length of 35 pixels. After tracking, the results of MQT are projected back to the original image for fair comparison. To obtain meaningful superpixels of appropriate size, the regularized size of SLIC [1] is set to 17. Regarding the parameters of the proposed model in (7), we set $\sigma = 0.1$, $\alpha = 5$, $\beta = 0.3$, $\lambda = 2$, $\gamma = 0.1$, and $\mu = \frac{w \times h}{100^2}$, where w and h are the width and the height of the target bounding box, respectively. The minimum normalized distance $d(z, i)$ in (6) is computed by measuring minimum distance between the pixel i and the boundary of bounding box $\mathbf{B}(z)$ in a resized coordinate system, in which the size of target bounding box becomes $w' = h' = 1$. The number of trees T is set to 15 for

Table 1. Non-rigid object tracking: percentage of correctly tracked frames

	HT [15]	TLD [25]	PixelTrack [14]	SPT[52]	**MQT**	**P**	**S**	**B**	**P&S**	**P&B**	**S&B**
Cliff-dive 1	**100.00**	69.12	**100.00**	**100.00**	**100.00**	**100.00**	97.06	**100.00**	**100.00**	**100.00**	**100.00**
Motocross 1	**100.00**	15.38	57.69	29.49	43.59	30.13	30.13	40.38	44.87	66.67	35.26
Skiing	**100.00**	6.85	**100.00**	17.28	**100.00**	**100.00**	7.41	9.88	98.77	98.77	9.88
Mountain-bike	**100.00**	81.36	94.55	**100.00**	**100.00**	**100.00**	4.39	39.04	**100.00**	**100.00**	18.86
Cliff-dive 2	**100.00**	8.20	32.79	**100.00**	**100.00**	**100.00**	78.69	50.82	**100.00**	**100.00**	62.30
Volleyball	45.12	42.28	**100.00**	46.55	**100.00**	**100.00**	28.46	25.00	60.16	**100.00**	28.66
Motocross 2	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
Transformer	38.71	33.06	94.35	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
Diving	21.21	24.68	88.74	41.56	**97.84**	13.42	35.06	24.24	96.54	13.85	55.84
High Jump	77.87	35.25	94.26	19.67	**98.36**	84.43	8.20	8.20	90.16	91.80	41.80
Gymnastics	98.87	84.75	99.09	21.90	**100.00**	96.87	95.57	29.47	98.17	97.13	59.06
Average	80.16	45.54	87.41	61.50	**94.53**	84.08	53.18	47.91	89.88	88.02	55.60

Note: The left panel shows the comparison between different trackers. Right panel summaries the results of baseline performance, where P, S, B are the trackers using single quantization of pixel (RF), superpixel (RF), and bounding box (MFT), respectively, and P&S the tracker using the two quantizations from pixel and superpixel, etc.

both pixel- and superpixel-level RFs. Other parameters specific to the sample selection in ORF model training are $\epsilon_p^+ = 0.8$, $\epsilon_p^- = 0.3$, $\epsilon_{sp}^+ = 0.5$, $\epsilon_{sp}^- = 0.5$. It should be noted that we strictly follow the protocols proposed in [14,54] and fix all parameters for all video sequences in the following evaluations. We use the initial bounding box given by the dataset as the starting point for our tracker.

The tracker was implemented using Matlab & C++ without intensive program optimization. The average time cost for all testing sequences is 1.1s per frame on a cluster node (3.4GHz, 8 Cores, 32GB RAM, less than 19% CPU used), consisting of: feature extraction (0.13s), SLIC (0.10s), RF prediction (0.18s), RF update (0.38s), and dynamic graph cuts (0.30s). It should be noted that parallel programming can be easily adopted for some key components (e.g., feature extraction, graph cuts, and ORF) to significantly reduce the run-time.

4.2 Tracking Non-rigid Objects

We first evaluate the performance of our tacker on the non-rigid object tracking dataset, which was first collected by [15] and was recently used to test a state-of-the-art tracking method [14]. This dataset consists of 11 sequences, where significant non-rigid deformation of objects is present. Quantitative results on a set of representative frames are shown in Fig. 3. For quantitative comparison, we compare MQT with a set of state-of-the-art methods[2], including PixelTrack [14], Hough Tracker (HT) [15], TLD [25], and the Superpixel Tracker (SPT) [52], by computing the success rate, defined as the percentage of frames in which the object is successfully tracked. In each frame, the overlap measure (i.e., half of the DICE coefficient) $S_o = \frac{|R_t \cap R_g|}{|R_t \cup R_g|}$ is computed, where R_t is the bounding box output by a tracker and R_g is the ground truth bounding box. The tracking is considered successful is S_o is larger than a given threshold t_o. For a fair comparison, we use the same protocol as in [14] by setting t_o to 0.1. The quantitative results of the comparative trackers are summarized in the left panel of Tab. 1.

[2] The performance of SPT is evaluated by using the code released by Wang *et al.* [52], and the data for the other three competitors' is from [14].

Fig. 3. Tracking results of MQT in the non-rigid object tracking dataset. Frame numbers are shown in the top left of each figure. Each column contains results of three sequences: (A) Cliff-dive 1, Cliff-dive 2, Mountain-bike; (B) Diving, High Jump, Gym.

As pointed out in [14], one of the most difficult videos for the PixelTrack is *Motocross 1*, where the motor-bike does a complete flip, changes its size rapidly, and the background is very cluttered. The rapid size change poses great challenge to our tracker given a fixed size bounding box we adopt. The quantitative evaluation shows that our tracker successfully tracks objects in almost all of the sequences in this dataset, and significantly outperforms the other four methods. Our tracker is demonstrated to be a promising method for tracking non-rigid objects, possessing the advantage of multilevel appearance representation incorporated in a graphical model, compared to the methods (e.g., PixelTrack, SPT, TLD) based on only a single level representation.

Moreover, to better understand the importance of different components in the proposed framework, we also conducted the baseline experiments to evaluate performance on parts of our tracker by switching off some components and summarized the average performance in the right panel of Tab. 1.

4.3 Evaluation on Comprehensive Benchmark

The second experiment is conducted on the CVPR2013 tracking benchmark [54], which is an up-to-date comprehensive benchmark specifically designed for evaluation of tracking performance. The whole dataset consists of 50 fully annotated sequences. Each sequence is tagged with a number of attributes indicating to the presence of different challenges, e.g. Occlusion (OCC), Deformation (DEF). To evaluate the strength and weakness of different methods, the sequences are categorized according to those attributes and 11 challenge subsets are therefore created. In [54], the evaluation is based on two different metrics: the precision plot and success plot. The precision plot shows the percentage of frames on which the Center Location Error (CLE) of a tracker is within a given threshold r, where CLE is defined as the center distance between R_t and R_g, and a representative precision score ($r = 20$) is used for ranking. Another metric is to compute the bounding box overlap S_o introduced in the previous experiment (Section 4.2), and the success plot shows the ratios of successful frames at a given threshold

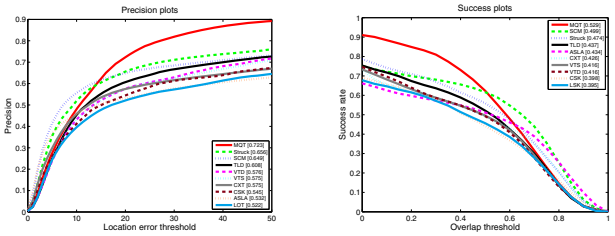

Fig. 4. Quantitative comparison in CVPR2013 benchmark. The performance score for each tracker is shown in the legend. For each figure, only the top 10 trackers are presented. The trackers appearing in the legend are as follows: MQT (ours), Struck [17], SCM [57], TLD [25], VTD [29], VTS [30], CXT [13], CSK [19], ASLA [23], LOT [44], LSK [40].

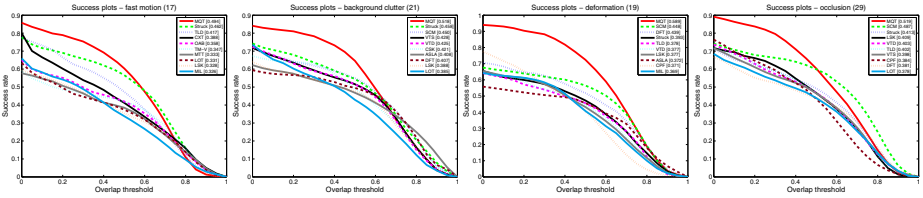

Fig. 5. Success plots for some challenge subsets of CVPR2013 tracking benchmark. The performance score for each tracker is shown in the legend. The value appears in the title is the number of sequences in that subset. Only the top 10 trackers are presented. The trackers appearing in the legend are as follows: OAB [16], TM-V [12], DFT [34], CPF [45], MIL [5].

t_o varied from 0 to 1. In success plot, the ranking is based on the area under curve (AUC) instead of using a specific threshold. For the comparative trackers, it currently includes 29 popular tracking algorithms, and most of them operated on a single choice of quantization. For more details about the benchmark, we refer readers to the original paper [54].

We run the One-Pass Evaluation (OPE) [54] on the benchmark using the proposed MQT. For comparison, we use the online available[3] tracking results and the unified tool provided by [54] to compute the evaluation plots. In this experiment, the proposed MQT achieves overall the best performance using both the metrics, which is shown in Fig. 4. MQT also ranks in the top ten from all 30 trackers over all challenge subsets using either the measurement of precision plots or success plots and takes the first places in the nine out of the eleven challenge subsets when using the success plots as measurement. According to the results, MQT is more robust to background clutter, deformation, occlusion challenges comparing to the other 29 trackers. We show the success plots of the some challenge subsets in Fig. 5, but omit other figures due to the space limits.

[3] `http://visual-tracking.net/`

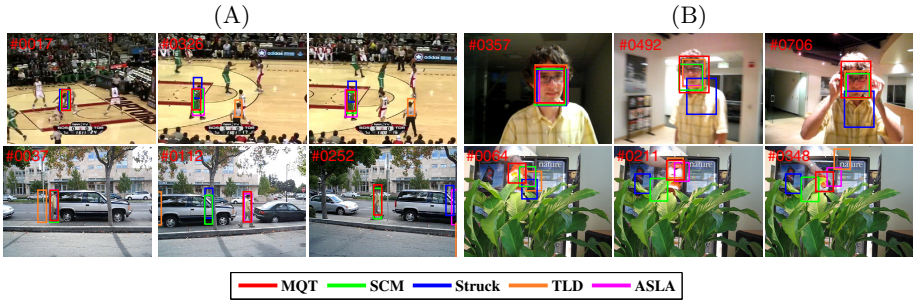

Fig. 6. Tracking results in the CVPR2013 benchmark. Only top five trackers on success plots are presented. Frame numbers are shown in the top left of each figure. Each column contains results of two sequences: (A) Basketball, David3; (B) David, Tiger1.

Finally, qualitative comparison with the top-rank trackers is shown in Fig. 6 for more intuitive demonstration. Note that, due to the adoption of a fixed-size bounding box and the lack of strong holistic-appearance model, the predicted bounding box will only partially capture the target in the presence of heavy occlusion and scale changes, which can be interpreted from Fig. 4 and Fig. 5.

5 Conclusions and Future Work

In this paper, we propose a tracking method based on a hierarchical appearance representation using multilevel quantization. The different levels of the representation are incorporated into a Conditional Random Field model using a coherent framework. By exploiting all the quantization levels, the method utilizes and integrates the information contained at each representation level by explicitly modeling the interactions and constraints between them; this results in significantly improved performance compared to other state-of-the-art tracking methods based on a single quantization. Moreover, Online Random Forests are used to update the appearance model in different levels of the tracker, in order to capture changes in object appearance over time. The experimental results demonstrate that the proposed method is capable of taking advantage of multilevel information and significantly boosting tracking performance. In the future, we will improve our tracker by considering the scale change of the target and extend it by taking more sophisticated high-level information into consideration.

Acknowledgment. This project is supported by Australian Research Council Discovery Projects number FT-130101457 and DP-120103730, and it is also supported by Toyota Research Institute NA collaborative project 2013001793.

References

1. Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., Susstrunk, S.: SLIC superpixels compared to state-of-the-art superpixel methods. TPAMI 34(11), 2274–2282 (2012)
2. Adam, A., Rivlin, E., Shimshoni, I.: Robust fragments-based tracking using the integral histogram. In: CVPR, pp. 798–805 (2006)
3. Aeschliman, C., Park, J., Kak, A.C.: A probabilistic framework for joint segmentation and tracking. In: CVPR, pp. 1371–1378 (2010)
4. Avidan, S.: Ensemble tracking. TPAMI 29(2), 261–271 (2007)
5. Babenko, B., Yang, M., Belongie, S.: Robust object tracking with online multiple instance learning. TPAMI 33(8), 1619–1632 (2011)
6. Bosch, A., Zisserman, A., Muoz, X.: Image classification using random forests and ferns. In: ICCV, pp. 1–8 (2007)
7. Boykov, Y., Funka-Lea, G.: Graph cuts and efficient nd image segmentation. IJCV 70(2), 109–131 (2006)
8. Boykov, Y., Kolmogorov, V.: An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. TPAMI 26(9), 1124–1137 (2004)
9. Breiman, L.: Random forests. Machine Learning 45(1), 5–32 (2001)
10. Brunelli, R.: Template matching techniques in computer vision: theory and practice. John Wiley & Sons (2009)
11. Chockalingam, P., Pradeep, N., Birchfield, S.: Adaptive fragments-based tracking of non-rigid objects using level sets. In: ICCV, pp. 1530–1537 (2009)
12. Collins, R., Liu, Y., Leordeanu, M.: Online selection of discriminative tracking features. TPAMI 27(10), 1631–1643 (2005)
13. Dinh, T.B., Vo, N., Medioni, G.: Context tracker: Exploring supporters and distracters in unconstrained environments. In: CVPR, pp. 1177–1184 (2011)
14. Duffner, S., Garcia, C.: Pixeltrack: a fast adaptive algorithm for tracking non-rigid objects. In: ICCV, pp. 2480–2487 (2013)
15. Godec, M., Roth, P.M., Bischof, H.: Hough-based tracking of non-rigid objects. In: ICCV, pp. 81–88 (2011)
16. Grabner, H., Bischof, H.: On-line boosting and vision. In: CVPR, pp. 260–267 (2006)
17. Hare, S., Saffari, A., Torr, P.H.: Struck: Structured output tracking with kernels. In: ICCV, pp. 263–270 (2011)
18. He, X., Zemel, R.S., Carreira-Perpiñán, M.Á.: Multiscale conditional random fields for image labeling. In: CVPR, pp. 695–702 (2004)
19. Henriques, J.F., Caseiro, R., Martins, P., Batista, J.: Exploiting the circulant structure of tracking-by-detection with kernels. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part IV. LNCS, vol. 7575, pp. 702–715. Springer, Heidelberg (2012)
20. Hong, Z., Mei, X., Prokhorov, D., Tao, D.: Tracking via robust multi-task multi-view joint sparse representation. In: ICCV, pp. 649–656 (2013)
21. Hong, Z., Mei, X., Tao, D.: Dual-force metric learning for robust distracter-resistant tracker. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part I. LNCS, vol. 7572, pp. 513–527. Springer, Heidelberg (2012)
22. Huang, Q., Han, M., Wu, B., Ioffe, S.: A hierarchical conditional random field model for labeling and segmenting images of street scenes. In: CVPR, pp. 1953–1960 (2011)

23. Jia, X., Lu, H., Yang, M.H.: Visual tracking via adaptive structural local sparse appearance model. In: CVPR, pp. 1822–1829 (2012)
24. Kalal, Z., Mikolajczyk, K., Matas, J.: Forward-backward error: Automatic detection of tracking failures. In: ICPR, pp. 2756–2759 (2010)
25. Kalal, Z., Mikolajczyk, K., Matas, J.: Tracking-learning-detection. TPAMI 34(7), 1409–1422 (2012)
26. Kohli, P., Rihan, J., Bray, M., Torr, P.H.: Simultaneous segmentation and pose estimation of humans using dynamic graph cuts. IJCV 79(3), 285–298 (2008)
27. Kohli, P., Torr, P.H.: Dynamic graph cuts for efficient inference in markov random fields. TPAMI 29(12), 2079–2088 (2007)
28. Kwon, J., Lee, K.M.: Tracking of a non-rigid object via patch-based dynamic appearance modeling and adaptive basin hopping monte carlo sampling. In: CVPR, pp. 1208–1215 (2009)
29. Kwon, J., Lee, K.M.: Visual tracking decomposition. In: CVPR, pp. 1269–1276 (2010)
30. Kwon, J., Lee, K.M.: Tracking by sampling trackers. In: ICCV, pp. 1195–1202 (2011)
31. Ladicky, L., Russell, C., Kohli, P., Torr, P.H.: Associative hierarchical crfs for object class image segmentation. In: ICCV, pp. 739–746 (2009)
32. Ladický, L., Sturgess, P., Alahari, K., Russell, C., Torr, P.H.S.: What, where and how many? Combining object detectors and CRFs. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part IV. LNCS, vol. 6314, pp. 424–437. Springer, Heidelberg (2010)
33. Lagarias, J.C., Reeds, J.A., Wright, M.H., Wright, P.E.: Convergence properties of the nelder–mead simplex method in low dimensions. SIAM Journal on Optimization 9(1), 112–147 (1998)
34. Learned-Miller, E., Sevilla-Lara, L.: Distribution fields for tracking. In: CVPR, pp. 1910–1917 (2012)
35. Lepetit, V., Fua, P.: Keypoint recognition using randomized trees. TPAMI 28(9), 1465–1479 (2006)
36. Leung, T., Malik, J.: Representing and recognizing the visual appearance of materials using three-dimensional textons. IJCV 43(1), 29–44 (2001)
37. Levinshtein, A., Sminchisescu, C., Dickinson, S.: Optimal contour closure by superpixel grouping. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part II. LNCS, vol. 6312, pp. 480–493. Springer, Heidelberg (2010)
38. Li, H., Shen, C., Shi, Q.: Real-time visual tracking using compressive sensing. In: CVPR, pp. 1305–1312 (2011)
39. Li, X., Hu, W., Shen, C., Zhang, Z., Dick, A., Hengel, A.V.D.: A survey of appearance models in visual object tracking. ACM Transactions on Intelligent Systems and Technology (TIST) 4(4) (2013)
40. Liu, B., Huang, J., Yang, L., Kulikowsk, C.: Robust tracking using local sparse appearance model and k-selection. In: CVPR, pp. 1313–1320 (2011)
41. Mei, X., Ling, H.: Robust visual tracking and vehicle classification via sparse representation. TPAMI 33(11), 2259–2272 (2011)
42. Mei, X., Ling, H., Wu, Y., Blasch, E., Bai, L.: Efficient minimum error bounded particle resampling L1 tracker with occlusion detection. TIP 22(7), 2661–2675 (2013)
43. Ojala, T., Pietikainen, M., Maenpaa, T.: Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. TPAMI 24(7), 971–987 (2002)
44. Oron, S., Bar-Hillel, A., Levi, D., Avidan, S.: Locally orderless tracking. In: CVPR, pp. 1940–1947 (2012)

45. Pérez, P., Hue, C., Vermaak, J., Gangnet, M.: Color-based probabilistic tracking. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) ECCV 2002, Part I. LNCS, vol. 2350, pp. 661–675. Springer, Heidelberg (2002)
46. Ross, D., Lim, J., Lin, R., Yang, M.: Incremental learning for robust visual tracking. IJCV 77(1), 125–141 (2008)
47. Rother, C., Kolmogorov, V., Blake, A.: Grabcut: Interactive foreground extraction using iterated graph cuts. ACM Transactions on Graphics (TOG) 23(3), 309–314 (2004)
48. Saffari, A., Leistner, C., Santner, J., Godec, M., Bischof, H.: On-line random forests. In: ICCV Workshops, pp. 1393–1400 (2009)
49. Santner, J., Leistner, C., Saffari, A., Pock, T., Bischof, H.: Prost: Parallel robust online simple tracking. In: CVPR, pp. 723–730 (2010)
50. Wang, C., Komodakis, N., Paragios, N.: Markov random field modeling, inference & learning in computer vision & image understanding: A survey. CVIU 117(11), 1610–1627 (2013)
51. Wang, C., de La Gorce, M., Paragios, N.: Segmentation, ordering and multi-object tracking using graphical models. In: ICCV, pp. 747–754 (2009)
52. Wang, S., Lu, H., Yang, F., Yang, M.H.: Superpixel tracking. In: ICCV, pp. 1323–1330 (2011)
53. Wojek, C., Schiele, B.: A dynamic conditional random field model for joint labeling of object and scene classes. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part IV. LNCS, vol. 5305, pp. 733–747. Springer, Heidelberg (2008)
54. Wu, Y., Lim, J., Yang, M.H.: Online object tracking: A benchmark. In: CVPR, pp. 2411–2418 (2013)
55. Yang, H., Shao, L., Zheng, F., Wang, L., Song, Z.: Recent advances and trends in visual tracking: A review. Neurocomputing 74(18), 3823–3831 (2011)
56. Zhang, T., Ghanem, B., Liu, S., Ahuja, N.: Robust visual tracking via multi-task sparse learning. In: CVPR, pp. 2042–2049 (2012)
57. Zhong, W., Lu, H., Yang, M.H.: Robust object tracking via sparsity-based collaborative model. In: CVPR, pp. 1838–1845 (2012)