2016 23rd International Conference on Pattern Recognition (ICPR)
Cancún Center, Cancún, México, December 4-8, 2016

Online RGB-D Tracking via Detection-Learning-Segmentation

Ning An
Institute of Automation
Chinese Academy of Sciences
Beijing, China
ning.an@ia.ac.cn

Xiao-Guang Zhao
Institute of Automation
Chinese Academy of Sciences
Beijing, China
xiaoguang.zhao@ia.ac.cn

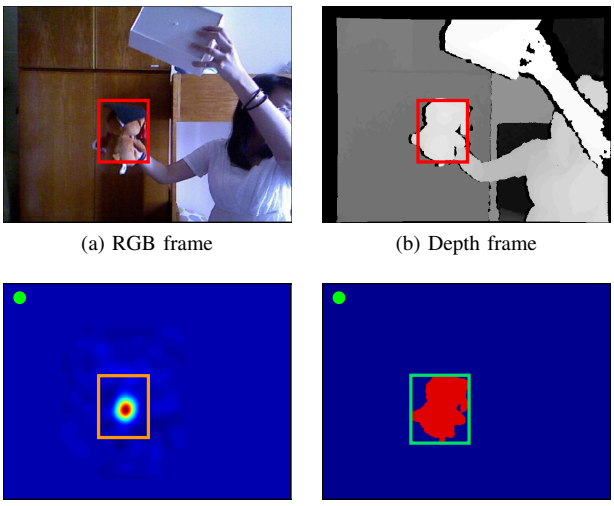
Zeng-Guang Hou
Institute of Automation
Chinese Academy of Sciences
Beijing, China
zengguang.hou@ia.ac.cn

***Abstract*—In this paper, we address the problem of online RGB-D tracking where the target object undergoes significant appearance changes. To sufficiently exploit the color and depth cues, we propose a novel RGB-D tracking framework (DLS) that simultaneously builds the target 2D appearance model and 3D distribution model. The framework decomposes the tracking task into detection, learning and segmentation. The detection and segmentation components locate the target collaboratively by using the two target models. An adaptive depth histogram is proposed in the segmentation component to efficiently locate the target in depth frames. The learning component estimates the detection and segmentation errors, updates the target models from the most confident frames by identifying two kinds of distractors: potential failure and occlusion. Extensive experimental results on a large-scale benchmark dataset show that the proposed method performs favourably against state-of-the-art RGB-D trackers in terms of efficiency, accuracy, and robustness.**

I. Introduction

Object tracking is one of the fundamental problems in computer vision with many applications. The most general type of object tracking is single-object model-free online tracking, in which an object is initialized in the first frame and tracked in the subsequent frames with no prior knowledge. Despite significant progresses in the last decades [1], [2], object tracking is still challenging due to deformation, occlusion, background clutter, and abrupt motion. Nowadays the RGB-D sensors have been used in computer vision tasks and demonstrate great performance improvement [3]. Therefore, in this paper we consider to exploit the color (RGB) and depth cues in RGB-D frames to overcome the tracking challenges mentioned above.

Recently, the RGB-D trackers [4], [5], [6], [7], [8] have shown superior performance against state-of-the-art RGB trackers. However, two prominent issues occur in these approaches. The first issue is the insufficient target modeling method. The depth channel of a RGB-D frame is merely regarded as a gray image for further feature extraction and model learning. This method ignores the physical property of the depth cue and thus the learned model lacks target presentation power. The second issue lies in the inefficient occlusion handling method. Some trackers estimate the occlusion state in an implicit way, e.g., the occlusion state is valid whenever the tracking confidence is below a threshold. False estimations may happen in clutter scenes since the target appearance is similar to the background and the tracking confidence is less reliable. Other trackers locate the target by using traditional color image segmentation methods during occlusion, which may not be effective in the depth frames.



(a) RGB frame (b) Depth frame (c) Detection (d) Segmentation

Fig. 1: Outline of our approach. The DLS framework tracks the target in the RGB-D frames (1a, 1b) via decomposing the task into detection (1c), learning, and segmentation (1d). The green dot indicates no learning distractor occurs in detection and segmentation results respectively. Best viewed in color.

In this work, we propose a novel detection-learning-segmentation (DLS) tracking framework to address the above two issues by: (i) simultaneously learning 2D appearance model and 3D distribution model of the target; (ii) efficiently segmenting target by using an adaptive depth histogram. The outline of the DLS framework is shown in Figure 1. The RGB-D tracking task is decomposed into detection, learning and segmentation by the DLS. First, the target location is estimated by detection and segmentation independently and simultaneously in the current frame. The detection component uses a kernelized correlation filter to locate the target by 2D appearance model and the segmentation component uses an adaptive depth histogram to locate the target by 3D distribution

978-1-5090-4846-5/16/$31.00 ©2016 IEEE

model. Then, the learning component identifies two kinds of distractors from the estimated locations: potential failure and occlusion. Afterwards, the estimated locations are integrated into the final target location according to the distractor state. Finally, the learning component updates the appearance model and the distribution model at the integrated target location if no distractor occurs.

The main contributions of this paper are:

(i) A novel RGB-D tracking framework is designed, where 2D appearance cue and 3D distribution cue are simultaneously exploited to precisely model the target.

(ii) Target segmentation is carried out efficiently by building and analyzing an adaptive depth histogram.

(iii) A distractor-aware learning method is proposed to effectively alleviate model drift problem.

(iv) We evaluate the proposed method on a large-scale RGB-D tracking benchmark dataset [4] with 100 challenge videos and demonstrate its efficiency, accuracy, and robustness against state-of-the-art trackers.

II. Related Work

Significant progresses have been made in the RGB object tracking [2], [9]. On the contrary, RGB-D tracking is a new research branch of the object tracking and a few works have been done on it. We review the works closely related to our method from three aspects in this section.

Tracking by Detection. Object tracking is regarded as a detection process in this framework. During tracking, a set of positive and negative samples are collected to train the target classifier. Many advanced techniques have been applied to the classifier updating, including multiple-instance learning [10], boosting [11], [12], P-N learning [13], structured output SVM [14], and deep learning [15], [16], [17]. The correlation filter based trackers [18], [19], [20], [21], [22] are proposed rencetly to settle the sampling ambiguity problem in binary classifiers. They use all circular-shift samples to learn a regression model and use fast Fourier transforms to perform high computational efficiency. However, these trackers are prone to drift due to their relatively risky model update methods. In our work, we propose a distractor-aware learning method that updates the model in most confident frames to alleviate the model drift problem.

Tracking by Segmentation. A pixel-level probabilistic framework is proposed to jointly solve segmentation and tracking in [23]. *GrabCut* [24] is applied to segment each sample in a particle filter framework in [25]. Structural information captured in superpixels is used to build a robust target appearance model in [26]. PixelTrack [27] uses pixel-based descriptors and a probabilistic segmentation method to track the objects. In [28], an online gradient boosting decision tree operating on individual patches is intergraded and the final result is provided by segmentation masks. The above methods may not be effective in depth frame segmentation since the property of the depth frame is essentially different with the color frame. We propose an adaptive depth histogram in this work to efficiently segment the target.

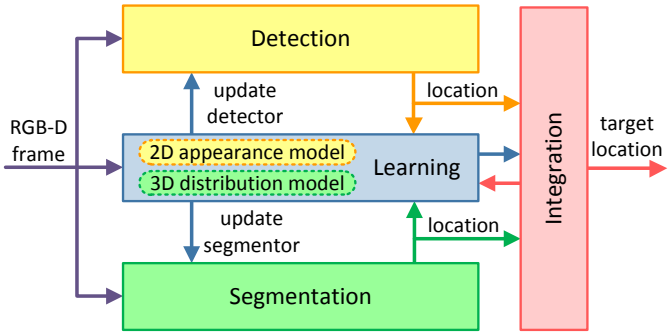


Fig. 2: Block flowchart of the DLS framework. Best viewed in color.

RGB-D Tracking. In tracker [7], depth features are fused with grayscale and color features in a feature pool to train a boosting classifier. In [4], a SVM classifier is trained using HOG features extracted in color and depth frames, large displacement optical flow is integrated and occlusion is handled by assuming that the target is the closest object in the bounding box when there is no occlusion. The particle filter tracking framework is applied in [5], each particle is equipped with a latent occlusion flag variable to estimate occlusion. In [6], a kernelized correlation filter with combined color and depth features is applied for fast tracking, the depth cue is exploited to estimate scale change and handle occlusion. However, in the occlusion handle procedure, the estimation is implicit in [7], [5], and the bounding box aspect ratio is fixed in [7], [6], which will lead to an inaccuracy tracking result. Meanwhile, trackers are inefficiency due to massive appearance feature fusion [7], [5], extra occlusion tracking [4], [6] or computationally expensive classifier and optical flow algorithm [4]. Unlike the above methods, our proposed DLS framework uses an adaptive depth segmentation method to handle occlusion and uses a distribution model besides the appearance model to precisely and effectively model the target.

III. Detection-Learning-Segmentation

DLS is a framework designed for single-object model-free online tracking in RGB-D frames. The framework simultaneously builds two target models: (i) the 2D appearance model is built upon the commonly used image features extracted from color and depth frames; (ii) the 3D distribution model is built according to the point cloud distribution on the target surface. Detailed block flowchart of the DLS is shown in Figure 2. First, *Detection* and *Segmentation* independently and simultaneously estimate the target location. Second, *Learning* identifies distractors from the estimated locations. Third, *Integration* fuses the location estimations into the final target location according to the distractor state. Finally, *Learning* updates the two target models at the target location if no distractor occurs.

A. Correlation Detection

For robust and efficient performance, we use the kernelized correlation filter (KCF) [19], [20] in the detection component.

The KCF uses a filter $\mathbf{w}$ to model the target appearance. The filter is trained on a $M \times N$ image patch $\mathbf{x}$, which is l_t times larger than the target to contain background information. All the circular shift versions of $\mathbf{x}$ are used as training samples and each shifted sample $\mathbf{x}_{m,n}$, $(m,n) \in \{0,1,\ldots,M-1\} \times \{0,1,\ldots,N-1\}$, is labeled by a Gaussian function $y_{m,n} = \exp(-\frac{(m-M/2)^2+(n-N/2)^2}{2\sigma_l^2})$. The convolution response of $\mathbf{w}$ on the sample $\mathbf{x}_{m,n}$ is given by $f(\mathbf{x}_{m,n}) = \mathbf{w} \cdot \phi(\mathbf{x}_{m,n})$, where ϕ denotes the mapping to a kernel space. The filter is trained by minimizing the L^2-error:

$$\mathbf{w} = \underset{\mathbf{w}}{\operatorname{argmin}} \sum_{m,n} (f(\mathbf{x}_{m,n}) - y_{m,n})^2 + \lambda \|\mathbf{w}\|^2, \tag{1}$$

where λ is the regularization parameter. Using a kernel $\kappa(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x}) \cdot \phi(\mathbf{x}')$, the filter can be computed as $\mathbf{w} = \sum_{m,n} \alpha_{m,n} \phi(\mathbf{x}_{m,n})$, where $\boldsymbol{\alpha}$ is the dual variable of $\mathbf{w}$. According to [19], if the used kernel is shift invariant, $\boldsymbol{\alpha}$ can be computed efficiently by using the discrete Fourier transformation (DFT) in the Fourier domain:

$$\hat{\boldsymbol{\alpha}} = \frac{\hat{\mathbf{y}}}{\hat{\mathbf{k}}^{\mathbf{xx}} + \lambda}, \tag{2}$$

where hat $\hat{}$ denotes the DFT operator, $\mathbf{k}^{\mathbf{xx}}$ is a matrix whose element (m,n) is $\kappa(\mathbf{x}_{m,n}, \mathbf{x})$. We use the shift invariant Gaussian kernel in this work.

During the target detection stage, a $M \times N$ image patch $\mathbf{z}$ is extracted at the last target location as the search window. The detection response map can be obtained by evaluating all cyclic shift patches of $\mathbf{z}$:

$$\mathbf{f}(\mathbf{z}) = \mathcal{F}^{-1}(\hat{\mathbf{k}}^{\mathbf{xz}} \odot \hat{\boldsymbol{\alpha}}), \tag{3}$$

where $\mathbf{x}$ and $\boldsymbol{\alpha}$ denote the learned target appearance model. The target bounding box is estimated by searching the location of the maximal value of $\mathbf{f}(\mathbf{z})$. To cope with scale changes, we apply the filter in multiple search window sizes $\mathbf{s} = \{s_1, \ldots, s_p\}$ similar to [29]. We extract search window patches $\{\mathbf{z}_r\}$, $r \in \mathbf{s}$, at the scale r relative to the last target scale. The optimal scale $\tilde{s}$ of the target is:

$$\tilde{s} = \underset{r}{\operatorname{argmax}}\, \mathbf{f}(\mathbf{z}_r). \tag{4}$$

To sufficiently model the target appearance, we use HOG descriptors [30] in both color and depth frames. Moreover, we extract color names [31], [20] in color frames to encode the color informations.

B. Segmentation with Adaptive Depth Histogram

We propose an adaptive depth histogram to efficiently segment the target in depth frames. The histogram is built according to the target 3D distribution model $x_d \sim \mathcal{N}(\mu_d, \sigma_d^2)$, where x_d denotes the depth value of cloud point on the target surface, μ_d and σ_d are the mean and standard deviation of x_d. By analyzing the histogram, the target depth μ_d^t in the current frame is estimated straightforwardly and the target region is obtained by the shifted distribution model $x_d \sim \mathcal{N}(\mu_d^t, \sigma_d^2)$.

The segmentation task is carried out in the 3D search window centering on the last target location. For implementing simplicity, we project cloud points into a depth image patch, which is l_s times larger than the target in the last frame. Target is segmented by analyzing the adaptive depth histogram h_d extracted on the image patch. The histogram h_d has two parameters: bin width b_w and bin range $[0, b_r]$. It only considers pixels in the foreground, therefore pixels with depth values larger than b_r are left out. During the histogram extraction, parameters b_w and b_r are determined adaptively according to the target distribution model:

$$b_w = q_w \sigma_d, \tag{5a}$$

$$b_r = \mu_d + q_r \sigma_d, \tag{5b}$$

where q_w and q_r are proportion factors. Once the histogram h_d is extracted, peaks are computed on it. We assume each peak corresponds an object region in the foreground. The peak closest to μ_d is regarded as the target depth μ_d^t in the current frame. Then we only remain pixels potentially belong to the target in the search window, whose depth value is within $[\mu_d^t - 3\sigma_d, \mu_d^t + 3\sigma_d]$ according to the centrality of the distribution model. The connected component with the maximum size is regraded as the target region. The target location is estimated by computing the enclosing bounding box over the target region.

C. Distractor-Aware Learning

The learning component updates the target models from most confident frames in a distractor-aware manner. We define two kinds of distractors: (i) *potential failure* and (ii) *occlusion*. During tracking, the model updating is turned off whenever the distractor occurs to avoid the model drift problem.

Potential failure estimation. Potential failure is estimated under the assumption that the appearance of the target should not change drastically between consecutive frames. To quantize the appearance change, we regularize the response of the tracking result in frame t as follows:

$$g(t) = \left| 1 - \frac{\max f(\mathbf{x}^t)}{\max f(\mathbf{x}^{t-1})} \right|, \tag{6}$$

where $\mathbf{x}^t$ and $\mathbf{x}^{t-1}$ denote the search patch in frame t and $t-1$ respectively. Note that the response maps of $\mathbf{x}^t$ and $\mathbf{x}^{t-1}$ should be computed under the same appearance model obtained in frame $t-1$. The potential failure state G_f is set as:

$$G_f = \begin{cases} \text{True}, & g(t) > \lambda_{fail} \\ \text{False}, & \text{otherwise.} \end{cases} \tag{7}$$

Occlusion estimation. Occlusion can be effectively estimated by using the target segmentation result. Let p_f denote the peak with smaller depth value than the target in depth histogram h_d, N_{p_f} denote the number of pixels belong to peak p_f, N_S denote the total number of pixels in the search window. The occlusion state G_o can be obtained by considering whether there are other objects in front of the target:

$$G_o = \begin{cases} \text{True}, & N_{p_f}/N_S > \lambda_{occ} \\ \text{False}, & \text{otherwise.} \end{cases} \tag{8}$$

When no distractor occurs in the current frame, *i.e.* $G_f \vee G_o = \text{False}$, the tracking result is judged realiable. Then the learning component updates the appearance model and the distribution model simultaneously:

$$\tilde{\mathbf{x}}^t = (1-\gamma_a)\tilde{\mathbf{x}}^{t-1} + \gamma_a \mathbf{x}^t, \tag{9a}$$

$$\tilde{\boldsymbol{\alpha}}^t = (1-\gamma_a)\tilde{\boldsymbol{\alpha}}^{t-1} + \gamma_a \boldsymbol{\alpha}^t, \tag{9b}$$

$$\tilde{\sigma}_d^t = (1-\gamma_d)\tilde{\sigma}_d^{t-1} + \gamma_d \sigma_d^t, \tag{9c}$$

where t is the frame index, $\tilde{\mathbf{x}}$ and $\tilde{\boldsymbol{\alpha}}$ denote the target appearance model, $\tilde{\sigma}_d$ denotes the target distribution model, γ_a and γ_d are the learning rate respectively.

D. Integration

The detection and segmentation components use essential different models to locate the target. To sufficiently exploit the complementarity of them, the integration component fuses the detection bounding box $\mathbf{b}_t^a$ and the segmentation bounding box $\mathbf{b}_t^s$ into the final output $\mathbf{b}_t$ according to the distractor state G_o and G_f:

$$\mathbf{b}_t = \begin{cases} (1-\beta_i)\mathbf{b}_t^a + \beta_i \mathbf{b}_t^s, & G_o = \text{False} \\ \mathbf{b}_t^s, & G_o = \text{True} \wedge G_f = \text{True} \\ \mathbf{b}_t^a, & G_o = \text{True} \wedge G_f = \text{False}. \end{cases} \tag{10}$$

During the normal stage ($G_o = \text{False}$), $\mathbf{b}_t^a$ and $\mathbf{b}_t^s$ are fused in a linear manner; during the occlusion stage ($G_o = \text{True}$), $\mathbf{b}_t^s$ is more reliable and $\mathbf{b}_t^a$ is used to correct $\mathbf{b}_t^s$ when the appearance of $\mathbf{b}_t^a$ has a high similarity to the target before occlusion ($G_f = \text{False}$). We summarize the overall algorithm in Algorithm 1.

Algorithm 1 Detection-Learning-Segmentation algorithm

Input: RGB-D frames $\{I_t^c, I_t^d\}_1^T$, initial bounding box $\mathbf{b}_1$
Output: Bounding box predictions $\{\mathbf{b}_t\}_2^T$,
Target 2D appearance model $\{\tilde{\mathbf{x}}, \tilde{\boldsymbol{\alpha}}\}$,
Target 3D distribution model $\tilde{\sigma}_d$.

for $t = 2 : T$ **do**
  Crop out the searching window in frame t;
  // Detection
  Compute the correlation maps $\mathbf{f}(\mathbf{z}_r)$ using $\{\tilde{\mathbf{x}}, \tilde{\boldsymbol{\alpha}}\}$ and Eq. (3), estimate the new bounding box $\mathbf{b}_t^a$;
  // Segmentation
  Compute the adaptive depth histogram h_d using $\tilde{\sigma}_d$ and Eq. (5), estimate the new bounding box $\mathbf{b}_t^s$;
  // Learning—distractor estimation
  Estimate the potential failure state G_f using Eq. (7) and the occlusion state G_o using Eq. (8);
  // Integration
  Predict $\mathbf{b}_t$ by integrating $\mathbf{b}_t^a$ and $\mathbf{b}_t^s$ using Eq. (10);
  // Learning—model update
  if $G_f \vee G_o = \text{False}$ **then**
    Update the appearance model $\{\tilde{\mathbf{x}}, \tilde{\boldsymbol{\alpha}}\}$ and the distribution model $\tilde{\sigma}_d$ using Eq. (9);
  end if
end for

IV. Experimental Results

We evaluate the proposed method on a large RGB-D tracking benchmark dataset [4]. First, we decompose and evaluate the proposed RGB-D tracking framework. Next, we compare the proposed DLS tracker with state-of-the-art RGB-D trackers in both overall and attribute-based manners.

A. Experimental Setups

Our approach is implemented in native Matlab without optimization. The experiments are carried out on an Intel I5-2400 3.10 GHz CPU with 4GB RAM.

Parameters. In detection, the padding size l_t is set to 1.5, the regularization parameter is set to $\lambda = 10^{-4}$, the width of the Gaussian kernel is set to $\sigma = 0.5$, the scaling pool is set to $\mathbf{s} = \{0.94, 0.96, 0.98, 1.00, 1.02, 1.04, 1.06\}$. In segmentation, the padding size l_s is set to 0.5, the proportion factors q_w and q_r is set to 3.0 and 6.0 respectively. In learning, the thresholds for distractor estimation are set to $\lambda_{occ} = 0.1$ and $\lambda_{fail} = 0.8$, the learning rate γ_a and γ_d are set to 0.02 and 0.3 respectively. In integration, the ratio β_i is set to 0.5. All Parameters are fixed for all videos in the experiments.

Datasets. We evaluate our method on the Princeton Tracking Benchmark (PTB) [4]. The dataset contains 100 RGB-D videos which are annotated with 11 attributes to indicate different challenges in tracking task. Among the 100 videos, ground truth of 5 validation videos are released for parameter setting. Oppositely ground truth of the rest 95 evaluation videos are reserved to prevent data-specific parameter tuning. We run our algorithm on all evaluation videos in PTB and submit our tracking results to the website of the dataset for evaluation.

Evaluation Methodology. To validate the performance of our approach, we follow the evaluation metric used in [4]. Results are evaluated by the *average success rate*, which can be obtained by calculating the area-under-the-curve (AUC) of the tracker's success plot. The success plot is defined as the percentage of success frames where the overlap between result bounding box and ground truth exceeds a threshold $t_o \in [0, 1]$.

B. Component analysis

To demonstrate the sufficiency and efficiency of the proposed DLS framework, we implement two more algorithms using individual components of the DLS. First, we implement a tracker DT merely using the detector of the DLS. In addition, we implement a tracker DLT with the detection and learning components of DLS, DLT can update the model selectively by considering distractors during tracking. We report the results on the 95 benchmark evaluation videos using the average success rate metric. As shown in Table I and Figure 3, the DLT tracker outperforms the DT tracker in all attributes since the additional learning component helps the tracker to update the target model correctly and avoid model drift. The proposed DLST tracker (use all components) outperforms the DLT tracker, performances are improved in a large margin

TABLE I: Experimental results on the Princeton Tracking Benchmark: average successful rates (SR) and rankings (in parentheses) are presented under different attributes. The best and the second best results are in red and blue respectively.

Algorithm	all SR	target type			target size		movement		occlusion		motion type	
		human	animal	rigid	large	small	slow	fast	yes	no	passive	active
DLST	0.74(1)	0.77(1)	0.69(2)	0.73(4)	0.80(1)	0.70(4)	0.73(5)	0.74(1)	0.66(2)	0.85(2)	0.72(5)	0.75(1)
PrinT [4]	0.73(2)	0.74(2)	0.63(3)	0.78(1)	0.78(2)	0.70(2)	0.76(2)	0.72(2)	0.72(1)	0.75(4)	0.82(1)	0.70(3)
OAPF [5]	0.73(3)	0.64(4)	0.85(1)	0.77(2)	0.73(3)	0.73(1)	0.85(1)	0.68(3)	0.64(3)	0.85(1)	0.78(3)	0.71(2)
DS-KCF [6]	0.69(4)	0.67(3)	0.61(4)	0.76(3)	0.69(4)	0.70(3)	0.75(3)	0.67(4)	0.63(4)	0.78(3)	0.79(2)	0.66(4)
PrinT-PC [4]	0.59(5)	0.51(5)	0.52(5)	0.73(5)	0.63(5)	0.56(5)	0.74(4)	0.53(5)	0.55(5)	0.64(6)	0.75(4)	0.53(5)
SAMFD[29]	0.54(6)	0.45(6)	0.50(6)	0.67(6)	0.52(6)	0.55(6)	0.65(6)	0.49(6)	0.41(6)	0.72(5)	0.66(6)	0.49(6)
DT	0.57	0.49	0.57	0.67	0.56	0.58	0.68	0.53	0.43	0.77	0.67	0.54
DLT	0.63	0.55	0.63	0.74	0.59	0.67	0.72	0.60	0.49	0.84	0.73	0.60

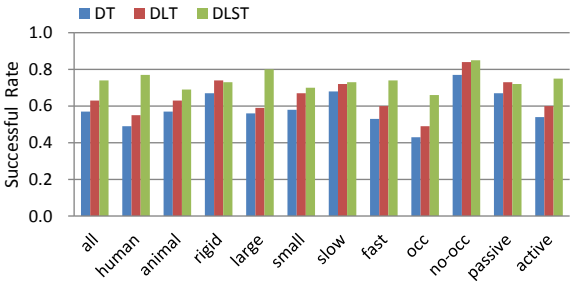


Fig. 3: Comparison of the trackers equipped with different components of the DLS framework. The DLST tracker equipped with all the components performs best.

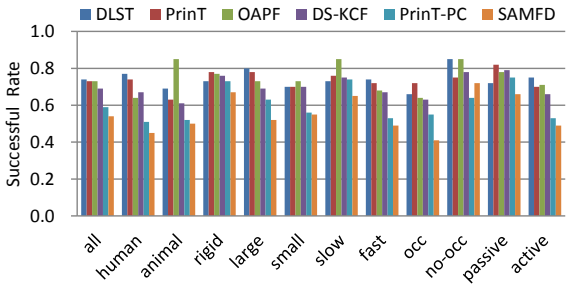


Fig. 4: Comparison with state-of-the-art RGB-D trackers. The proposed DLST tracker performs favorably against other RGB-D tracking algorithms.

in *human*, *large*, *fast*, *occ*, and *active* attributes due to the superior power of segmentation in handle deformation and occlusion. The analysis demonstrates that components in the DLS framework collaborate with each other in a effective and complementary way.

C. State-of-the-art Comparison

We compare our DLST algorithm with five state-of-the-art RGB-D trackers on the benchmark. The trackers are from four different categories of tracking algorithms: (i) correlation trackers (DS-KCF [6], SAMFD [29]); (ii) binary discriminative trackers (PrinT [4]); (iii) generative trackers (OAPF [5]); and (iv) point cloud trackers (PrinT-PC [4]). We report both the overall and attribute-based results using the average success rate metric in Table I and Figure 4.

Overall performance. The *all SR* column in Table I shows that our DLST tracker preforms favorably against state-of-the-art RGB-D trackers, since it sufficiently exploits both appearance cue and distribution cue to learn a more accurate target model. To evaluate the efficiency of the considered algorithms, we report the average frame rate of the top three trackers over the whole benchmark in Figure 5. Our algorithm is much faster than the other two top rank trackers, which demonstrates an excellent performance-speed ratio due to the use of simpler features and efficient depth segmentation method. Figure 6 shows the qualitative evaluation of the proposed algorithm against state-of-the-art trackers. Our algorithm is able to locate the target accurately in challenging scenarios such as deformation (row *1*, *3*, *4*), occlusion (row *1*, *2*, *3*, *4*), background clutter (row *3*, *4*), and abrupt motion (row *5*).

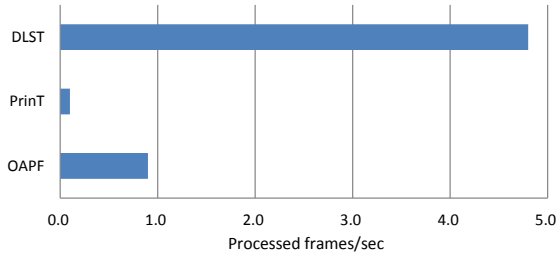


Fig. 5: Average frame rate of the top three RGB-D trackers on PTB. Our DLST tracker achieve superior computational efficiency.

Attribute-based performance. To further analyze the performance of the proposed algorithm in different categorizations, we report the result comparisons for all 11 attributes in Table I and Figure 4. Among state-of-the-art methods, our DLST algorithm performs the best results in *human*, *large*, *fast*, and *active* categorizations, performs the second best results in *animal*, *occlusion*, and *no occlusion* categorizations. The results show that our algorithm achieves comparable performance in almost all categorizations and superior performance in handle targets with deformation and fast motion, since in the DLST the distribution model is robust against deformation and the learning component estimate distractors effectively in scenes with drastic variations.

V. Conclusions

In this paper, we propose a novel framework (DLS) for online RGB-D tracking. The DLS framework simultaneously

Fig. 6: Qualitative evaluation of the proposed tracking algorithm, the PrinT [4], DS-KCF [6], SAMFD [29] trackers on five challenging videos (from top to down are *bdog_occ2*, *bear_back*, *computerbar1*, *new_ex_occ2*, and *zball_no2*, respectively). Our algorithm performs consistently against state-of-the-art RGB-D trackers. Best viewed in color.

builds target 2D appearance model and 3D distribution model, effectively taking advantage of the color and depth cues. In the DLS, tracking task is decomposed in to detection, learning, and segmentation: the detection component uses a correlation filter to locate the target by 2D appearance model, the segmentation component uses an adaptive histogram to locate the target by 3D distribution model, the learning component updates the target models in a distractor-aware manner. Extensive experimental results on a large-scale RGB-D benchmark demonstrate that components of the framework complement with each other, and the proposed method performs favorably against the state-of-the-art trackers in terms of efficiency, accuracy, and robustness.

ACKNOWLEDGMENT

This work is supported by the National Natural Science Foundation of China under Grant 61271432.

REFERENCES

[1] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *Acm computing surveys*, vol. 38, no. 4, 2006.

[2] A. W. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah, "Visual tracking: An experimental survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 7, pp. 1442–1468, 2014.

[3] J. Han, L. Shao, D. Xu, and J. Shotton, "Enhanced computer vision with microsoft kinect sensor: A review," *IEEE Transactions on Cybernetics*, vol. 43, no. 5, pp. 1318–1334, 2013.

[4] S. Song and J. Xiao, "Tracking revisited using rgbd camera: Unified benchmark and baselines," in *ICCV*, 2013.

[5] K. Meshgi, S. Maeda, S. Oba, H. Skibbe, Y. Li, and S. Ishii, "Occlusion aware particle filter tracker to handle complex and persistent occlusions," *Computer Vision and Image Understanding*, 2015 to appear.

[6] M. Camplani, S. Hannuna, M. Mirmehdi, D. Damen, A. Paiement, L. Tao, and T. Burghardt, "Real-time rgb-d tracking with depth scaling kernelised correlation filters and occlusion handling," in *BMVC*, 2015.

[7] G. M. García, D. A. Klein, J. Stückler, S. Frintrop, and A. B. Cremers, "Adaptive multi-cue 3d tracking of arbitrary objects," *Pattern Recognition*, pp. 357–366, 2012.

[8] Q. Wang, J. Fang, and Y. Yuan, "Multi-cue based tracking," *Neurocomputing*, vol. 131, pp. 227–236, 2014.

[9] Y. Wu, J. Lim, and M.-H. Yang, "Object tracking benchmark," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 9, pp. 1834–1848, 2015.

[10] B. Babenko, M.-H. Yang, and S. Belongie, "Robust object tracking with online multiple instance learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 8, pp. 1619–1632, 2011.

[11] H. Grabner, M. Grabner, and H. Bischof, "Real-time tracking via on-line boosting." in *BMVC*, 2006.

[12] H. Grabner, C. Leistner, and H. Bischof, "Semi-supervised on-line boosting for robust tracking," in *ECCV*, 2008.

[13] Z. Kalal, K. Mikolajczyk, and J. Matas, "Tracking-learning-detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 7, pp. 1409–1422, 2012.

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

[15] N. Wang and D.-Y. Yeung, "Learning a deep compact image representation for visual tracking," in *NIPS*, 2013.

[16] C. Ma, J.-B. Huang, X. Yang, and M.-H. Yang, "Hierarchical convolutional features for visual tracking," in *ICCV*, 2015.

[17] L. Wang, W. Ouyang, X. Wang, and H. Lu, "Visual tracking with fully convolutional networks," in *ICCV*, 2015.

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

[19] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2015.

[20] M. Danelljan, F. Khan, M. Felsberg, and J. Weijer, "Adaptive color attributes for real-time visual tracking," in *CVPR*, 2014.

[21] C. Ma, X. Yang, C. Zhang, and M.-H. Yang, "Long-term correlation tracking," in *CVPR*, 2015.

[22] M. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg, "Learning spatially regularized correlation filters for visual tracking," in *ICCV*, 2015.

[23] C. Aeschliman, J. Park, and A. C. Kak, "A probabilistic framework for joint segmentation and tracking," in *CVPR*, 2010.

[24] C. Rother, V. Kolmogorov, and A. Blake, "Grabcut: Interactive foreground extraction using iterated graph cuts," *ACM Transactions on Graphics*, vol. 23, no. 3, pp. 309–314, 2004.

[25] V. Belagiannis, F. Schubert, N. Navab, and S. Ilic, "Segmentation based particle filtering for real-time 2d object tracking," in *ECCV*, 2012.

[26] S. Wang, H. Lu, F. Yang, and M. H. Yang, "Superpixel tracking," in *ICCV*, 2011.

[27] S. Duffner and C. Garcia, "Pixeltrack: a fast adaptive algorithm for tracking non-rigid objects," in *ICCV*, 2013.

[28] J. Son, I. Jung, K. Park, and B. Han, "Tracking-by-segmentation with online gradient boosting decision tree," in *ICCV*, 2015.

[29] Y. Li and J. Zhu, "A scale adaptive kernel correlation filter tracker with feature integration," in *ECCV Workshop*, 2014.

[30] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *CVPR*, 2005.

[31] J. van de Weijer, C. Schmid, J. Verbeek, and D. Larlus, "Learning color names for real-world applications," *IEEE Transactions on Image Processing*, vol. 18, no. 7, pp. 1512–1523, 2009.