Superpixel-based Tracking-by-Segmentation using Markov Chains

Donghun Yeo[†] Jeany Son Bohyung Han Joon Hee Han
Dept. of Computer Science and Engineering, POSTECH, Korea
[†]donghun.yeo@stradvision.com {[†]hanulbog, jeany, bhhan, joonhan}@postech.ac.kr

Abstract

We propose a simple but effective tracking-by-segmentation algorithm using Absorbing Markov Chain (AMC) on superpixel segmentation, where target state is estimated by a combination of bottom-up and top-down approaches, and target segmentation is propagated to subsequent frames in a recursive manner. Our algorithm constructs a graph for AMC using the superpixels identified in two consecutive frames, where background superpixels in the previous frame correspond to absorbing vertices while all other superpixels create transient ones. The weight of each edge depends on the similarity of scores in the end superpixels, which are learned by support vector regression. Once graph construction is completed, target segmentation is estimated using the absorption time of each superpixel. The proposed tracking algorithm achieves substantially improved performance compared to the state-of-the-art segmentation-based tracking techniques in multiple challenging datasets.

1. Introduction

Visual tracking is a traditional topic in computer vision, but remains as a challenging task since target appearances involve significant variations and high-level scene understanding is often required to handle exceptions. Tracking-by-detection [14, 2, 4, 20, 16, 42] is one of the common strategies to deal with these challenges. However, they typically depend on bounding boxes for target representations, and often suffer from drifting problem when a target involves substantial non-rigid or articulated motions. Recently, segmentation-based tracking algorithms have been investigated actively [3, 13, 6, 10, 34], but most of them rely only on pixel-level information that is not sufficient to model semantic structure of target, or utilize external segmentation algorithms such as *Grabcut* [33]. Compared to the information from bounding boxes or pixels, mid-level cues such as superpixels may be effective to model both feature- and semantic-level information of target.

Superpixels have been used for various computer vision tasks, *e.g.*, object segmentation and recognition [31, 12, 28, 8, 32], background subtraction [24], and multi-target tracking [26, 29] due to its effectiveness in representation based on mid-level cues. In addition, the use of superpixels greatly reduces the complexity of sophisticated image processing and computer vision tasks since the number of superpixels is much smaller than the number of pixels obviously.

Visual tracking techniques often employ superpixels. Segmentation-based tracking algorithms relying on superpixels have been proposed to handle non-rigid and deformable targets [36, 37, 18, 40]. Wang *et al*. [36] uses superpixels for discriminative appearance modeling by mean-shift clustering, and they incorporate particle filtering to find the optimal target state. Instead of representing each object with a single holistic model, dynamic Bayesian network tracking [37] adopts a superpixel-based constellation model to deal with non-rigid deformations. However, since both methods categorize each superpixel into foreground or background independently based on low-level features, semantic relations between superpixels are not considered properly for segmentation. To overcome the limitation induced from the flat representations, Hong *et al*. [18] proposed a tracking method based on a hierarchical appearance representation using multiple quantization levels such as pixel, superpixel and bounding box. Xiao *et al*. [40] also presented a dynamic multi-level appearance modeling technique for tracking, which maintains an adaptive clustered decision tree using the information obtained from three different levels—pixel, superpixel, and bounding box. Recently, a tracking-by-segmentation algorithm that combines the information from pixels with bounding boxes has been proposed [34]. Note that [34, 18] require an external segmentation technique such as *Grabcut* [33].

We propose a novel tracking-by-segmentation framework using Absorbing Markov Chain (AMC) on superpixel segmentation, where the estimated target segmentation is propagated to subsequent frames in a recursive manner. To obtain target segmentation in the current frame, we first construct a graph for AMC using the superpixels in the previous and current frames, where a vertex corresponds to a superpixel and the weight of each edge is given by the scores learned from Support Vector Regression (SVR). Once the graph is constructed, target segmentation is obtained from

absorption time of each superpixel in the AMC, and the final tracking result is given by identifying connected components of superpixels corresponding to target. Our algorithm naturally estimates target segmentation through AMC on a graph defined in a spatio-temporal domain.

Our algorithm has several interesting features compared to the existing methods as summarized below:

- We propose a novel and principled tracking-by-segmentation framework well-suited for non-rigid and articulated objects using AMC. Our algorithm obtains initial segmentation masks as well as target segmentations naturally within the proposed framework.

- The proposed algorithm distinguishes foreground and background superpixels accurately based on the scores from a support vector regressor, which learns discriminative features more efficiently than metric learning.

- Our algorithm outperforms the state-of-the-art techniques substantially in challenging benchmark datasets for non-rigid and deformable object tracking.

The rest of this paper is organized as follows. We first review AMC in Section 2, and overview the proposed algorithm in Section 3. Section 4 presents the details about graph construction for AMC in the proposed algorithm, and Section 5 describes the procedure of our segmentation-based tracking. We illustrate experimental results including comparison with existing methods in Section 6.

2. Absorbing Markov Chain (AMC)

AMC is a specific kind of Markov chain that has at least one *absorbing state*, which can be reached from other states but may not be escaped from once entered since all the outgoing transition probabilities are zeros. Non-absorbing states in AMC are referred to as *transient states*. Each vertex has its *absorption time*, which is the expected number of steps from itself to any absorbing state by random walk. We employ AMC to estimate and propagate target segmentations in a spatio-temporal domain. AMC has been studied for several computer vision tasks, which include image matching [7], image segmentation [15], co-activity detection [41] and saliency detection [19].

Denote a graph for AMC by $\mathsf{G} = (\mathsf{V}, \mathsf{E})$, where V and E indicate a set of vertices (states) and edges, respectively. The vertex set can be further divided into transient and absorbing vertex sets, denoted by V^{T} and V^{A}, respectively, where $M_t = |\mathsf{V}^{\mathsf{T}}|$ and $M_a = |\mathsf{V}^{\mathsf{A}}|$.

To compute the absorption time of a vertex in an AMC, we first define the canonical transition matrix as follows:

$$\mathbf{P} = \begin{pmatrix} \mathbf{Q} & \mathbf{R} \\ \mathbf{0} & \mathbf{I} \end{pmatrix}, \tag{1}$$

where $\mathbf{Q} \in \mathbb{R}^{M_t \times M_t}$ is the transition probability matrix for all pairs of transient states and $\mathbf{R} \in \mathbb{R}^{M_t \times M_a}$ contains transition probabilities from transient states to absorbing states. Each row in $\mathbf{P}$ is normalized to sum to one. All transition probabilities from absorbing states to transient states are zeros, and all absorbing states only have single edges, which are self-loops; the corresponding transition submatrices are given by the zero matrix $\mathbf{0}$, and the identity matrix $\mathbf{I}$.

Suppose that q_{ij}^T is the probability of transition from $v_i \in \mathsf{V}^{\mathsf{T}}$ to $v_j \in \mathsf{V}^{\mathsf{T}}$ in T steps. Then, a random walk starting from v_i would visit v_j several times before arriving at one of absorbing states, and the expected number of visits to each transient node is given by the summation of q_{ij}^T, where $T \in [0, \infty)$. This procedure is simplified even further by a single matrix inversion as

$$\mathbf{F} = \sum_{T=0}^{\infty} \mathbf{Q}^T = (\mathbf{I} - \mathbf{Q})^{-1}, \tag{2}$$

where $\mathbf{F}$ is referred to as the fundamental matrix. The absorption time y_i of a random walk from v_i is given by the summation of the elements in the i^{th} row of $\mathbf{F}$.

3. Algorithm Overview

Our tracking algorithm is a combination of the bottom-up and top-down procedures, and Figure 1 illustrates the overall framework of the proposed algorithm.

We first construct a graph for AMC using all superpixels within the regions of interest in two consecutive frames. The vertices corresponding to background superpixels in the previous frame create absorbing states while all other superpixels are regarded as transient vertices. Edges connect two adjacent superpixels, where motion information is incorporated to determine temporal adjacency between superpixels in two different frames. The weight of each edge is given by the similarity of the predicted scores of the end superpixels, where the score is obtained by learning a support vector regressor that maximizes differences between the superpixels with different labels while minimizing differences between superpixels with same labels.

In the next step, an initial binary segmentation mask is identified by simply computing absorption times of transient vertices in the AMC. Since the initial segmentation mask may be noisy due to unexpected feature similarity between foreground and background superpixels and/or potential feature dissimilarity between foreground superpixels, we identify the final foreground segment corresponding to target by extracting multiple connected components of foreground superpixels within two hops in the AMC graph and selecting the most similar connected component to the global target appearance model based on color histogram.

In the first frame, given the target bounding box annotation, we set superpixels outside the bounding box as ab-

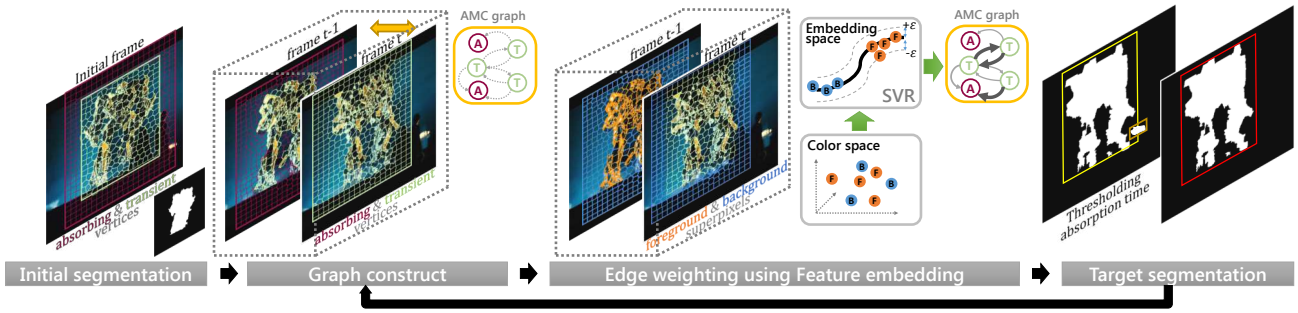


Figure 1: The overall framework of the proposed tracking algorithm. We first identify background superpixels in frame $t-1$ and set them as absorbing vertices in AMC. The AMC graph is constructed with two consecutive frames, $t-1$ and t, where inter-frame edges are created by spatial proximity found with motion information. Edge weights are determined by similarity of connected superpixels in the embedded space, which is learned by SVR using the representations of superpixels. The final tracking results for segmentation are obtained by evaluating connected components of superpixels based on a holistic appearance model after thresholding individual vertices using the absorption time.

sorbing vertices and obtain the initial segmentation mask by thresholding absorption times. Note that we have no inter-frame edges in the graph corresponding to the first frame.

4. Graph Construction for AMC

This section describes the details about the graph construction procedure for the AMC including discriminative edge weighting based on support vector regression.

4.1. Graph Topology

A graph for our AMC, $\mathsf{G} = (\mathsf{V}, \mathsf{E})$, is constructed based on the superpixels, which correspond to vertices in the graph. SLIC [1] superpixel segmentation algorithm is incorporated to obtain a set of superpixels from the region of interest (ROI) in each frame. Note that the ROI in the current frame is given by an enlarged bounding box surrounding the foreground propagated from the segmentation in the previous frame using optical flows. The formal definitions of vertices and edges are described below.

Vertices The vertices are divided into two subsets; one is a transient vertex set and the other is an absorbing vertex set, which are denoted by V^{T} and V^{A}, respectively. All superpixels have corresponding transient vertices in the graph while background superpixels create absorbing vertices additionally. This setting is particularly effective to handle false negative segments in the previous frame since the mislabeled background superpixels can be recovered depending on the features in the superpixels.

Edges There are two types of edges in the graph: intra-frame and inter-frame edges. The intra-frame edges connect the superpixels within 2 hops based on vertex adjacency in the same frame. A vertex within 1 hop means a direct neighbor while a vertex within 2 hops indicates a neighbor of neighbor. If a single superpixel is associated with both transient and absorbing vertices, they are 1 hop neighbors to each other. The inter-frame edges connect superpixels in two consecutive frames based on their temporal adjacency, which is determined by spatial overlap of superpixels after warping a superpixel by the motion vectors. We employ EPPM [5] to obtain pixelwise optical flow.

All edges are bi-directional and have symmetric weights, except the ones going to the absorbing vertices; such edges are unidirectional to satisfy the *absorbing* property. We define two edge types for convenience; transient edges connect two transient vertices, and the edges from transient vertices to absorbing ones are referred to as absorbing edges.

4.2. Embedding Features using Regression

The weight of each edge is given by the similarity of the scores associated with individual vertices, which are learned with features in superpixels. Edge weights between superpixels with the same labels should be larger than those between superpixels with the different labels. Therefore, we learn contrastive scores, which maximize differences between foreground and background samples while minimizing differences between examples with the same labels.

For the purpose, we adopt a support vector regression, where regressor learns a score by projecting the original feature of each superpixel onto an embedded space. To train the regressor, the superpixels within the target segment in the previous frame and the first frame are treated as foreground examples. Background examples consist of background superpixels, which do not correspond to the target in the previous and the first frame, and superpixels at the ROI boundary in the current frame, which are used to represent unseen backgrounds. Note that the information from the first frame is exploited to avoid drift problem.

Let $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ denote training dataset,

where $\mathbf{x}_i \in \mathbb{R}^d$ is a feature vector of sample i and $y_i \in \mathbb{R}$ is the label of each example. Note that the labels are real numbers conceptually but annotated as either +1 or -1 for foreground and background samples, respectively, since it is difficult to provide individual examples with real-numbered labels in practice. We employ simple features for representing superpixels, mean colors in LAB space, to learn the SVR efficiently. Optionally, a feature descriptor from convolutional neural network [27] is also used to learn SVR.

Then the objective function is defined as follows:

$$\begin{aligned} \arg\min_{\mathbf{w},\xi,\hat{\xi}} \ & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}(\xi_i + \hat{\xi}_i) \\ \text{s.t.} \ & y_i - \langle \mathbf{w}, \Phi(\mathbf{x}_i)\rangle - b \leq \varepsilon + \xi_i, \ \xi_i \geq 0, \\ & \langle \mathbf{w}, \Phi(\mathbf{x}_i)\rangle + b - y_i \leq \varepsilon + \hat{\xi}_i, \ \hat{\xi}_i \geq 0, \end{aligned} \tag{3}$$

where C is a constant, and $\Phi : \mathbb{R}^d \rightarrow \mathbb{R}^{d^*}$ $(d < d^*)$ denotes a nonlinear feature mapping function. We employ a radial basis function as a kernel,

$$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j)\rangle = \exp\left(\frac{||\mathbf{x}_i - \mathbf{x}_j||^2}{\gamma^2}\right), \tag{4}$$

for implicit nonlinear feature mapping, where γ is a constant. After training, the regression score of an arbitrary input $\mathbf{x}_i$ is given by,

$$r_i = f(\mathbf{x}_i) = \langle \mathbf{w}, \Phi(\mathbf{x}_i)\rangle. \tag{5}$$

The weight of an edge is given by the similarity of the regression scores associated with two end vertices as

$$w_{ij} = \exp\left(-\frac{|r_i - r_j|}{\sigma_r}\right), \tag{6}$$

where r_i and r_j are the regression scores of two adjacent vertices, $v_i \in \mathsf{V}$ and $v_j \in \mathsf{V}$, and σ_r is a constant.

5. Tracking-by-Segmentation using AMC

This section describes the procedure of our tracking-by-segmentation algorithm, which is mainly about how to combine the bottom-up and top-down estimations for robust target tracking. We also discuss how to initialize the segmentation at the first frame from the bounding box annotation.

5.1. Segmentation using Modified Absorption Time

The initial target segmentation mask in each frame is obtained by computing modified absorption times in the constructed AMC graph and thresholding the absorption times of transient vertices. To compute the absorption times in the standard AMC, we typically employ a canonical transition matrix, $\mathbf{P}$, which is constructed based on the weights of the edges in the graph given by Eq. (6). However, we slightly adjust the weight to increase discriminativeness of the absorption times of foreground and background superpixels. Let q_{ij} and r_{ij} denote the adjusted edge weights and correspond to the elements of $\mathbf{Q}$ and $\mathbf{R}$ at (i, j), respectively. Then, they are given by

$$q_{ij} = \frac{\pi_t w_{ij}}{\sum_{l=1}^{|\mathsf{V}|} \pi_{il} w_{il}} \quad \text{and} \quad r_{ik} = \frac{\pi_a w_{ik}}{\sum_{l=1}^{|\mathsf{V}|} \pi_{il} w_{il}}, \tag{7}$$

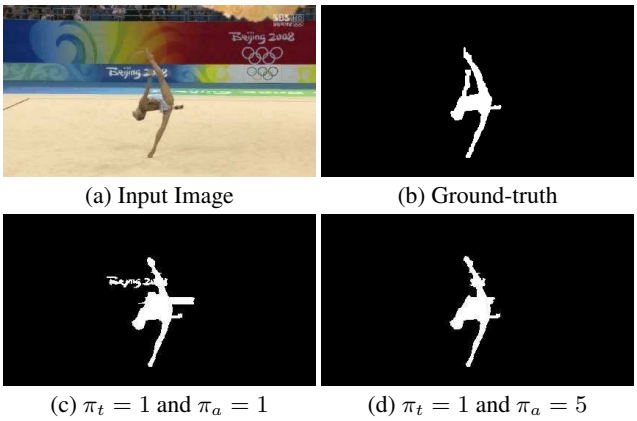
where $v_i, v_j \in \mathsf{V}^{\mathsf{T}}$, $v_k \in \mathsf{V}^{\mathsf{A}}$, and

$$\pi_{il} = \begin{cases} \pi_t, & \text{if } v_l \in \mathsf{V}^{\mathsf{T}} \\ \pi_a, & \text{if } v_l \in \mathsf{V}^{\mathsf{A}} \end{cases}. \tag{8}$$

Note that different coefficients are multiplied by the original weights depending on the type of edge, $\pi_t < \pi_a$, which facilitates fast absorption of a random walk starting from background superpixels and results in more discriminative absorption times. Figure 2 illustrates the impact of parameter setting of the coefficients. When the transition probability to absorbing and transient nodes are equally weighted, *i.e.* $\pi_a = \pi_t$, background superpixels are often labeled as foreground as illustrated in Figure 2(c).

Once the transition matrix is constructed based on the adjusted edge weights, we compute the absorption time of each superpixel using the fundamental matrix in Eq. (2). However, instead of the standard absorption time, we employ a modified version, which is the expected number of visits at the vertices corresponding to the foreground in the previous frame, for classification between foreground and background superpixels. The modified absorption time is formally defined as

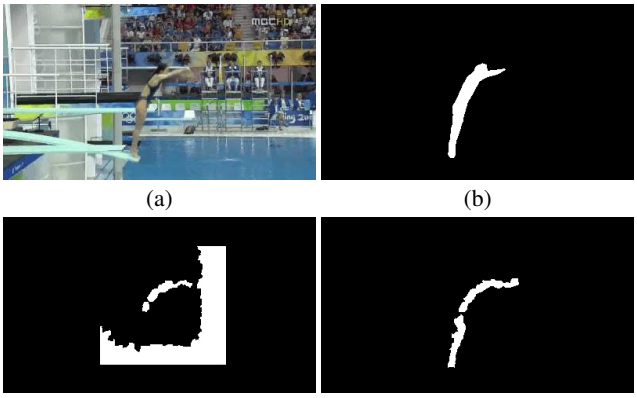
$$y_i^{\text{new}} = \sum_{v_j \in \mathsf{V}_{t-1}^{\mathsf{F}}} f_{ij}, \tag{9}$$



(a) Input Image (b) Ground-truth

(c) $\pi_t = 1$ and $\pi_a = 1$ (d) $\pi_t = 1$ and $\pi_a = 5$

Figure 2: The impact of the weight adjustment coefficients on the segmentation results. The coefficient tends to increase discriminativeness of absorption times of foreground and background superpixels.



(a) (b) (c) (d)

Figure 3: Comparison of segmentation results with the original and the modified absorption times: (a) input image, (b) ground-truth, (c) result with the original absorption time, and (d) result with the modified absorption time.

where $\mathsf{V}^{\mathsf{F}}_{t-1}$ is a set of vertices corresponding to foreground superpixels in frame $t-1$. The original absorption time computes the time spent on every transient vertex until a random walk reaches any absorbing vertex. In this formulation, the superpixels corresponding to the unseen backgrounds often have the large absorption times. The impact of the modified absorption time is demonstrated in Figure 3, where our modified absorption time is more effective to handle unseen background regions. Note that the threshold for classification is given by the average absorption time of the all transient vertices within the ROI in the current frame.

5.2. Target Detection by Global Appearance Model

The target segmentation mask generated by the pure bottom-up approach described in the previous subsection may be fragmented due to missing foreground superpixels and contain false-positive superpixels. To alleviate the target fragmentation problem, we group together the foreground segments connected within 2 hops in the AMC graph to construct target region candidate. Figure 4 illustrates an example of the foreground superpixel grouping. Since there can be multiple target region candidates after superpixel merges within 2 hops, we select the most similar connected component to the holistic target appearance model, which is based on the normalized color histogram of the pixels in the foreground segmentation mask. The dissimilarity is defined by the Bhattacharyya distance between two histograms. Once the target is identified, the histogram, h_t^{new}, is updated recursively as

$$h_t^{\text{new}} = (1 - w_c) \cdot h_t + w_c \cdot h_c, \tag{10}$$

where h_t is the current appearance model, h_c is the appearance model of the candidate, and $w_c = 0.1$ is the learning rate. Figure 5 presents the benefit of the holistic target modeling to identify the true target connected component.

Figure 4: Comparison of target superpixel merge with (left) 1-hop and (right) 2-hop neighborhood systems of surperpixel adjacency. Bounding boxes denote target candidates, and the magenta indicates the identified target using our holistic model. The 2-hop neighborhood system is effective to merge split superpixels belonging to target.

Figure 5: Benefit of holistic appearance model. There are two target region candidates in both frames, and our algorithm chooses the true target (magenta) using the holistic model based on color histogram.

5.3. Initial Segmentation at the First Frame

We apply a similar approach to initialize the target segmentation at the first frame. Since the optical flows are not available in the first frame, the initial segmentation is obtained by thresholding absorption time on the AMC graph using only intra-frame edges. The transient vertices are given by the superpixels overlapped with the initial ground-truth bounding box of target more than 50%. The superpixels in the extended target bounding boxes, which do not correspond to transient vertices, create absorbing vertices. The weight of each edge is computed based on the L_2-norm of the mean colors of two superpixels as in

$$w_{ij}^0 = \exp\left(-\frac{\|\mathbf{c}_i - \mathbf{c}_j\|}{\sigma_c}\right), \tag{11}$$

where $\mathbf{c}_i$ and $\mathbf{c}_j$ are the mean colors in LAB space of the superpixels corresponding to $v_i \in \mathsf{V}$ and $v_j \in \mathsf{V}$, respectively, and σ_c is a constant. We construct the fundamental matrix by multiplying different constant factors to the weights depending on edge types as in Eq. (7), and categorize individual superpixels using the modified absorption times.

6. Experiments

This section describes the details about datasets and our evaluation protocols, and presents the quantitative and qual-

itative results of our algorithm.

6.1. Datasets

We evaluate our algorithm in five independent datasets: non-rigid object tracking dataset (NR) [34], generalized background subtraction dataset (GBS) [22, 25, 24], video saliency dataset (VS) [11], SegTrack v2 dataset (ST2) [23], and DAVIS dataset [30][1]. The targets are annotated with both pixelwise binary segmentation masks and axis-aligned minimum bounding boxes. NR consists of 11 videos, which contain deformable and articulated objects, and has been used to evaluate segmentation-based tracking algorithms including [34]. The other datasets are constructed for other kinds of tasks such as foreground and background segmentation [25, 24, 23] and video saliency detection [11]. Some videos in GBS and ST2 have multiple foreground objects in each frame so we construct separate sequences for individual objects. Hence, GBS and ST2 initially contain 13 and 14 sequences, respectively, while now having 15 and 24 targets, respectively. Note that the targets in these two datasets also involve large deformations, occlusions, and low-resolution. VS consists of 10 sequences with target scale variations, where one video in VS contains multiple objects but we choose to track only one of them because the others are too small throughout the video. DAVIS contains 50 high quality videos for segmentation, where each video contains a single deformable and articulated object.

6.2. Algorithms for Comparison

Our tracking algorithm, denoted by AMC Tracking (AMCT), is compared with four recent segmentation-based trackers and a few bounding box trackers. The tested segmentation-based tracking techniques include Online Gradient Boosting Decision Tree Tracker (OGBDT) [34], HoughTrack (HT) [13], Superpixel Tracker (SPT) [36] and PixelTrack (PT) [10]. We choose DSST [9], MUSTer [17] and MEEM [42] among regular bounding box trackers.

In addition to the comparison with these external algorithms, we implement variations of our approach, which include AMCT without SVR (AMCT-NR), AMCT without holistic appearance model (AMCT-NA) and AMCT with CNN feature descriptor (AMCT+CNN). For AMCT-NR, we replace SVR scores with Lab colors to define an edge weight as in Eq. (11). AMCT+CNN integrates the feature descriptors from CNN as well as Lab color to learn SVR. We employ the CNN pre-trained for semantic segmentation [27], which is already used for online video segmentation in [35]. For these three internal variations, the rest of the implementation is identical to our full algorithm. The performance is measured based on pixel-level masks for segmentation accuracy while we employ bounding box annotations to compare with regular trackers.

We also apply AMCT to online video segmentation task, which aims to propagate segmentation masks given initial segmentation annotations at the first frame. This task is differentiated from our main problem, tracking-by-detection, which has bounding box annotations at the first frame. We select the two state-of-the-art video segmentation algorithms for this comparison: Joint Online Tracking and Segmentation (JOTS) [38] and Object Flow (OF) [35]. For fair comparison, we provide the initial ground-truth segmentation masks to all methods. Among the variations of our algorithm, we choose AMCT+CNN for this experiment, which is denoted by AMCT+CNN*, where the asterisk means the use of ground-truth segmentation annotation at the first frame.

6.3. Evaluation Metrics

There are two main criteria to evaluate tracking algorithm quantitatively, tracking success rate and precision, which are given by online tracking benchmark protocol [39]. Each tracker is evaluated using tracking success rate based on overlap ratios (intersection over union) between ground-truths and tracking results. We employ both bounding boxes and segmentation masks to compute success rates. The representative success rate of each algorithm is given by area under curve (AUC) of success plot, for which tracking success rates are computed at various overlap ratio thresholds. Another criterion, precision, is based on the center location errors between ground-truths and tracking results. The center location is given by the centroid of a bounding box, and precision is estimated with the Euclidean distances between the center locations of two bounding boxes. Typically, the precisions of tracking algorithms are compared using 20 pixel center location errors. In addition to the standard evaluation measures, we present the average overlap ratios between ground-truths and tracking results additionally.

6.4. Implementation Details

We employ SLIC [1] and EPPM [5] for superpixel segmentation and dense optical flow computation, respectively. These two algorithms are run with the publicly available source codes using the default parameters. The number of superpixels is proportional to the size of ROI in each frame with maximum 600. In the first frame, we start with 600 superpixels and ground-truth bounding box annotation to estimate the initial segmentation mask as described in Section 5.3.

There are several parameters in our algorithm. For SVR, C in Eq. (3) and γ in Eq. (4) are set to 10 and 1, respectively. The two free parameters for edge weights computation, σ_r

[1]The primary goal of our algorithm is pixel-level segmentation with tracking, so it is not directly comparable to the bounding box tracking methods evaluated on the online tracking benchmark [39] and visual object tracking benchmark [21].

Table 1: Average overlap ratio of segmentation masks for tracking-by-segmentation algorithms.

	AMCT	AMCT+CNN	AMCT-NR	AMCT-NA	OGBDT [34]	HT [13]	SPT [36]	PT [10]
NR	58.6	66.3	23.1	49.3	53.3	41.1	29.7	28.3
GBS	74.8	77.1	53.0	70.4	59.7	40.4	45.9	35.3
VS	84.1	82.3	71.4	83.8	79.8	51.2	61.0	73.9
ST2	58.8	71.3	47.2	60.7	47.6	43.0	26.3	21.2
DAVIS	59.2	65.1	41.2	56.9	44.9	33.1	27.1	26.1

Table 2: Average overlap ratio of bounding boxes for tracking-by-segmentation algorithms.

	(a) Segmentation-based trackers								(b) Regular trackers		
	AMCT	AMCT+CNN	AMCT-NR	AMCT-NA	OGBDT [34]	HT [13]	SPT [36]	PT [10]	DSST [9]	MUSTer [17]	MEEM [42]
NR	66.9	73.3	25.7	50.8	60.8	40.9	35.7	16.1	35.4	36.2	33.1
GBS	80.0	81.9	53.7	71.4	61.2	43.0	55.2	44.7	62.9	59.4	52.6
VS	88.2	88.7	75.4	88.1	78.8	57.6	61.5	51.9	66.9	64.1	60.3
ST2	64.8	76.3	50.3	64.3	50.2	44.9	53.5	32.2	62.0	58.8	59.5
DAVIS	60.9	67.8	44.5	60.1	50.0	35.8	43.2	41.6	58.4	25.9	52.7

in Eq. (6) and σ_c in Eq. (11) are set to 0.1 and 0.05, respectively. The parameters to define transition probability in Eq. (7) are $\pi_t = 1$ and $\pi_a = 3$. All the parameters are fixed throughout the evaluation.

6.5. Results

We now present and analyze comparative evaluation results on five different datasets for tracking-by-segmentation including NR, GBS, VS, ST2 and DAVIS. The overall performance of all compared algorithms in the five datasets are summarized in Table 1 and 2, where the best and second best trackers are highlighted with red and blue, respectively. We mark the best accuracy of online segmentation methods in bold in Table 3.

The proposed algorithms, AMCT and AMCT+CNN demonstrates outstanding performance in both segmentation mask and bounding box overlap ratios compared to the other tracking methods including OGBDT [34], which is the current state-of-the-art in tracking-by-segmentation. Note that the accuracy of AMCT is improved by incorporating CNN features. AMCT without support vector regression (AMCT-NR) works poorly in most datasets while AMCT without holistic appearance model (AMCT-NA) is competitive but still worse than our full algorithm in average. These results illustrate the contribution of two additional components in our algorithm; in particular, support vector regression plays very important role for robust tracking. The proposed methods, AMCT and AMCT+CNN, also outperform the state-of-the-art tracking algorithms for bounding box prediction in all tested datasets by large margins. This is mainly because bounding box tracking is not effective to follow highly articulated or deformable objects. Success and precision plots in all five datasets are illustrated in Figure 6.

Figure 7 illustrates target segmentation results of all compared tracking-by-segmentation algorithms in the five datasets. The second column presents the ground-truths in

Table 3: Average overlap ratio of segmentation masks for online video segmentation algorithms.

	AMCT+CNN*	JOTS [38]	OF [35]
NR	**66.4**	22.3	41.6
GBS	**79.0**	47.2	76.5
VS	**89.7**	79.2	88.8
ST2	**74.5**	51.5	69.5
DAVIS	**73.2**	47.6	70.7

segmentation and bounding box. Our algorithms extracts target object boundaries more accurately even with low contrast between foreground and background in *high-jump* sequences thanks to use of SVR. AMCT and AMCT+CNN track the target successfully in *board* sequence while other algorithms generate unreasonable segmentations in their results, and identify the non-convex object shape accurately in *humming bird2* sequence. AMCT+CNN tends to capture more precise target segmentation in *high-jump*, *dunk*, *humming bird2* and *motocross-jump* sequences by incorporating high-level semantic representations produced by CNN [27].

Although AMCT is originally designed for tracking-by-segmentation but it also achieves excellent performance in online video segmentation task as shown in Table 3. Especially, AMCT+CNN* outperforms OF by a significant margin in NR, which contains videos with deformable objects.

The proposed algorithm, AMCT, runs at about 4 fps, which includes time for EPPM optical flow computation (0.11 sec), SLIC superpixel segmentation (0.02 sec), graph construction (0.10 sec) and absorption time computation (0.01 sec). Compared to AMCT based only on LAB color descriptors, AMCT+CNN is slow since SVR training is substantially slow due to the high dimensionality of CNN features. AMCT+CNN takes 15 seconds per frame but is still much faster than JOTS and OF, which need about 80 seconds and 150 seconds per frame. The algorithms are tested with Intel Core i7-5930K CPU@3.50 GHz in MATLAB. We use a single NVIDIA Titan-X PASCAL GPU for

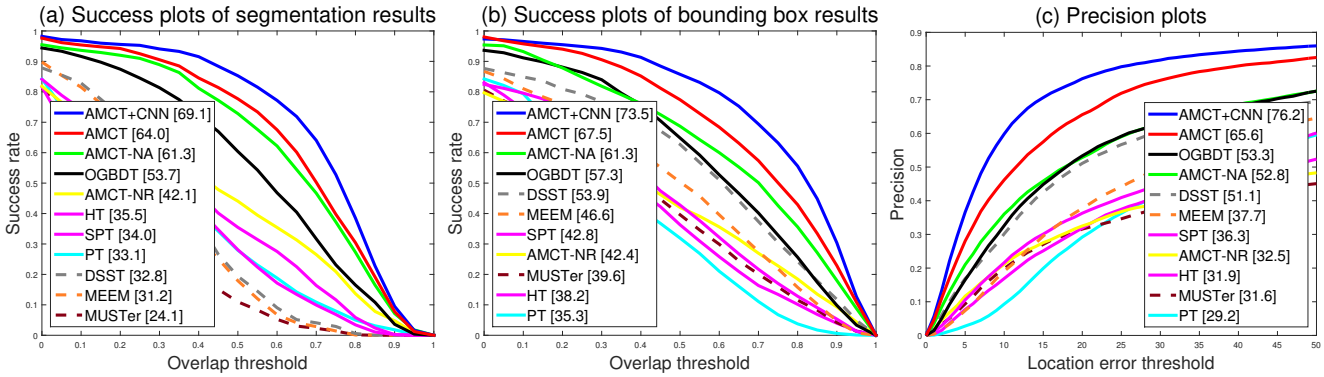


Figure 6: Success and precision plots in all five datasets: (a) success plots in terms of segmentation overlap ratio (b) success plots in terms of bounding box overlap ratio (c) precision plots

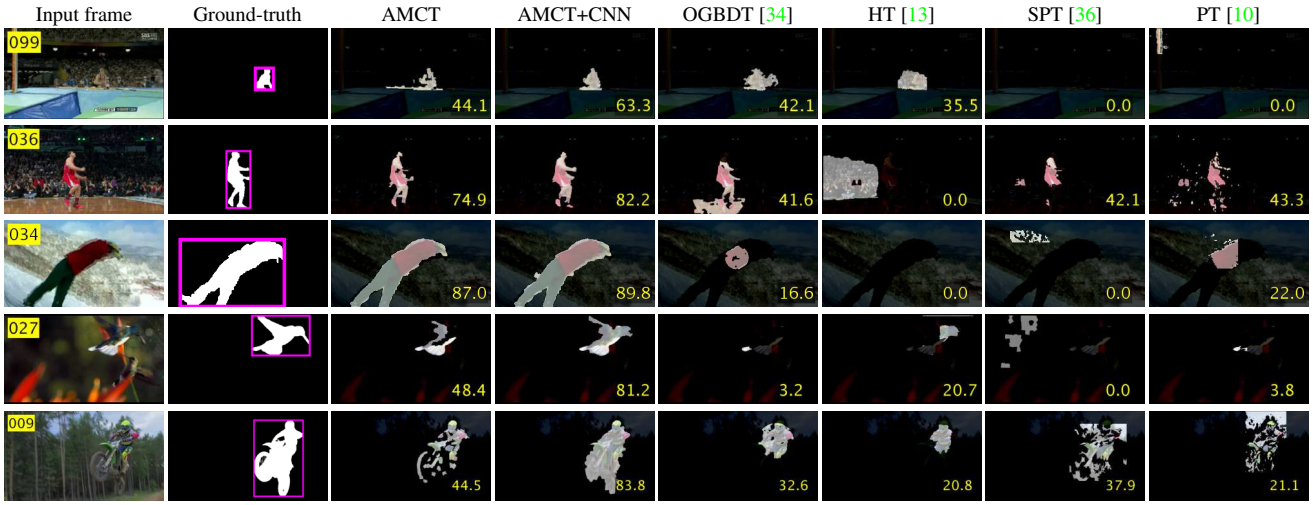


Figure 7: Qualitative performance evaluation results. The number in each image denotes segmentation overlap ratio. From top to bottoms, we present results of *high-jump* in NR, *dunk* in GBS, *board* in VS, *humming bird2* in ST2 and *motocross-jump* in DAVIS.

CNN feature computation.

Refer to our project page[2] for more detailed results. We plan to release source codes and all results from our experiment to facilitate reproduction of our algorithm.

7. Conclusion

We have proposed a simple but powerful superpixel-based tracking-by-segmentation algorithm using AMC. The proposed algorithm has a few interesting features, which include the application of AMC to visual tracking, edge weight learning using SVR, and accurate initial segmentations from bounding box annotations. Since our algorithm estimates target segmentations instead of target bounding boxes, it is more effective to track non-rigid and deformable objects. We compared our algorithm with the existing techniques related to tracking-by-segmentation and the state-of-the-art regular trackers in multiple challenging datasets, and achieved substantially better performance.

[2] http://cvlab.postech.ac.kr/research/AMCT/

Acknowledgements

This work was partly supported by the ICT R&D program of MSIP/IITP [2014-0-00059, Development of Predictive Visual Intelligence Technology (DeepView); 2016-0-00563, Research on Adaptive Machine Learning Technology Development for Intelligent Autonomous Digital Companion] and by the NRF grant [NRF-2011-0031648, Global Frontier R&D Program on Human-Centered Interaction for Coexistence].

References

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk. Slic superpixels compared to state-of-the-art

superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34:2274–2282, 2012. 3, 6

[2] A. Adam, E. Rivlin, and I. Shimshoni. Robust fragments-based tracking using the integral histogram. In *CVPR*, pages 798–805, 2006. 1

[3] C. Aeschliman, J. Park, and A. C. Kak. A probabilistic framework for joint segmentation and tracking. In *CVPR*, pages 1371–1378, 2010. 1

[4] B. Babenko, M.-H. Yang, and S. Belongie. Robust object tracking with online multiple instance learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33:1619–1632, 2011. 1

[5] L. Bao, Q. Yang, and H. Jin. Fast edge-preserving patchmatch for large displacement optical flow. *IEEE Transactions on Image Processing*, 23:4996–5006, 2014. 3, 6

[6] V. Belagiannis, F. Schubert, N. Navab, and S. Ilic. Segmentation based particle filtering for real-time 2d object tracking. In *ECCV*, pages 842–855, 2012. 1

[7] M. Cho, J. Lee, and K. M. Lee. Reweighted random walks for graph matching. In *ECCV*, pages 492–505, 2010. 2

[8] T. Cour and J. Shi. Recognizing objects by piecing together the segmentation puzzle. In *CVPR*, pages 1–8, 2007. 1

[9] M. Danelljan, G. Häger, F. Khan, and M. Felsberg. Accurate scale estimation for robust visual tracking. In *BMVC*, 2014. 6, 7

[10] S. Duffner and C. Garcia. Pixeltrack: a fast adaptive algorithm for tracking non-rigid objects. In *ICCV*, pages 2480–2487, 2013. 1, 6, 7, 8

[11] K. Fukuchi, K. Miyazato, A. Kimura, S. Takagi, and J. Yamato. Saliency-based video segmentation with graph cuts and sequentially updated priors. In *ICME*, pages 638–641, 2009. 6

[12] B. Fulkerson, A. Vedaldi, and S. Soatto. Class segmentation and object localization with superpixel neighborhoods. In *ICCV*, pages 670–677, 2009. 1

[13] M. Godec, P. M. Roth, and H. Bischof. Hough-based tracking of non-rigid objects. In *ICCV*, pages 81–88, 2011. 1, 6, 7, 8

[14] H. Grabner, M. Grabner, and H. Bischof. Real-time tracking via on-line boosting. In *BMVC*, page 6, 2006. 1

[15] P. He, X. Xu, and L. Chen. Constrained clustering with local constraint propagation. In *ECCV*, pages 223–232, 2012. 2

[16] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. Exploiting the circulant structure of tracking-by-detection with kernels. In *ECCV*, pages 702–715, 2012. 1

[17] Z. Hong, Z. Chen, C. Wang, X. Mei, D. Prokhorov, and D. Tao. Multi-store tracker (muster): a cognitive psychology inspired approach to object tracking. In *CVPR*, pages 749–758, 2015. 6, 7

[18] Z. Hong, C. Wang, X. Mei, D. Prokhorov, and D. Tao. Tracking using multilevel quantizations. In *ECCV*, pages 155–171, 2014. 1

[19] B. Jiang, L. Zhang, H. Lu, C. Yang, and M.-H. Yang. Saliency detection via absorbing markov chain. In *ICCV*, pages 1665–1672, 2013. 2

[20] Z. Kalal, K. Mikolajczyk, and J. Matas. Tracking-learning-detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34:1409–1422, 2012. 1

[21] M. Kristan et al. The visual object tracking vot2014 challenge results. In *ECCVW*, 2014. 6

[22] S. Kwak, T. Lim, W. Nam, B. Han, and J. H. Han. Generalized background subtraction based on hybrid inference by belief propagation and bayesian filtering. In *ICCV*, pages 2174–2181, 2011. 6

[23] F. Li, T. Kim, A. Humayun, D. Tsai, and J. M. Rehg. Video segmentation by tracking many figure-ground segments. In *ICCV*, pages 2192–2199, 2013. 6

[24] J. Lim and B. Han. Generalized background subtraction using superpixels with label integrated motion estimation. In *ECCV*, pages 173–187, 2014. 1, 6

[25] T. Lim, S. Hong, B. Han, and J. H. Han. Joint segmentation and pose tracking of human in natural videos. In *ICCV*, pages 833–840, 2013. 6

[26] L. Liu, J. Xing, H. Ai, and S. Lao. Semantic superpixel based vehicle tracking. In *ICPR*, pages 2222–2225, 2012. 1

[27] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, pages 3431–3440, 2015. 4, 6, 7

[28] A. Lucchi, Y. Li, K. Smith, and P. Fua. Structured image segmentation using kernelized features. In *ECCV*, pages 400–413, 2012. 1

[29] A. Milan, L. Leal-Taixé, K. Schindler, and I. Reid. Joint tracking and segmentation of multiple targets. In *CVPR*, pages 5397–5406, 2015. 1

[30] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *CVPR*, pages 724–732, 2016. 6

[31] X. Ren and J. Malik. Learning a classification model for segmentation. In *ICCV*, pages 10–17, 2003. 1

[32] A. Rosenfeld and D. Weinshall. Extracting foreground masks towards object recognition. In *ICCV*, pages 1371–1378, 2011. 1

[33] C. Rother, V. Kolmogorov, and A. Blake. Grabcut: interactive foreground extraction using iterated graph cuts. *SIGGRAPH*, 23:309–314, 2004. 1

[34] J. Son, I. Jung, K. Park, and B. Han. Tracking-by-segmentation using online gradient boosting decision tree. In *ICCV*, pages 3056–3064, 2015. 1, 6, 7, 8

[35] Y.-H. Tsai, M.-H. Yang, and M. J. Black. Video segmentation via object flow. In *CVPR*, pages 3899–3908, 2016. 6, 7

[36] S. Wang, H. Lu, F. Yang, and M.-H. Yang. Superpixel tracking. In *ICCV*, pages 1323–1330, 2011. 1, 6, 7, 8

[37] W. Wang and R. Nevatia. Robust object tracking using constellation model with superpixel. In *ACCV*, pages 191–204, 2013. 1

[38] L. Wen, D. Du, Z. Lei, S. Z. Li, and M.-H. Yang. Jots: Joint online tracking and segmentation. In *CVPR*, pages 2226–2234, 2015. 6, 7

[39] Y. Wu, J. Lim, and M.-H. Yang. Online object tracking: A benchmark. In *CVPR*, pages 2411–2418, 2013. 6

[40] J. Xiao, R. Stolkin, and A. Leonardis. Single target tracking using adaptive clustered decision trees and dynamic multi-level appearance models. In *CVPR*, pages 4978–4987, 2015. 1

[41] D. Yeo, B. Han, and J. H. Han. Unsupervised co-activity detection from multiple videos using absorbing markov chain. In *AAAI*, pages 3662–3668, 2016. 2

[42] J. Zhang, S. Ma, and S. Sclaroff. Meem: Robust tracking via multiple experts using entropy minimization. In *ECCV*, pages 188–203, 2014. 1, 6, 7