in Proc. IEEE Int'l Conf. Computer Vision (ICCV), Kyoto, Japan, 2009

Video Object Segmentation by Tracking Regions

William Brendel and Sinisa Todorovic
Oregon State University, Corvallis, OR 97331, USA,
{brendel, sinisa}@eecs.oregonstate.edu

Abstract

This paper presents an approach to unsupervised segmentation of moving and static objects occurring in a video. Objects are, in general, spatially cohesive and characterized by locally smooth motion trajectories. Therefore, they occupy regions within each frame. The shape and location of these regions vary slowly from frame to frame. Thus, video segmentation can be done by tracking regions across the frames such that the resulting tracks are locally smooth. To this end, we use a low-level segmentation to extract regions in all frames. Then, similar regions are transitively matched and clustered across the video. Region similarity is defined with respect to geometric and motion properties of region contours. To match region contours, we formulate a new circular dynamic-time warping (CDTW) algorithm that generalizes DTW to closed contours, without compromising the optimality and low complexity of DTW. Our quantitative evaluation and comparison with the state of the art suggest that the proposed approach is a competitive alternative to currently prevailing point-based methods.

1. Introduction

This paper presents an approach to unsupervised video object segmentation (VOS). Our goal is to delineate the boundaries of all moving and static objects occurring in an arbitrary video. In general, objects are spatially cohesive, and characterized by locally smooth motion trajectories. Therefore, they occupy regions within each video frame. Also, assuming relatively slow camera motions, the shape and location of these regions vary slowly from frame to frame. Thus, VOS can be formulated as tracking regions across the frames, such that the resulting tracks are locally smooth. This will partition the spatiotemporal video volume into tubes that are coherent in space and time. Since region boundaries coincide with object boundaries, a cross section of the tubes and any video frame will delineate all objects present in the frame.

VOS is a prerequisite step of a wide range of higher-level vision algorithms, including activity recognition [2, 9], video summarization and retrieval [6, 8], and nonphotorealistic video rendering [21, 3]. Most prior work focuses on a simplified formulation of VOS – that of motion segmentation [18, 22]. Typically, these methods require that the number of moving objects or layers is pre-specified, and cannot handle long videos [18, 22]. Also, motion segmentation using optical flow rests on the assumption of brightness constancy, which is violated at boundaries that move, resulting in poor estimates of object contours [14, 16].

Currently, the two predominant approaches to VOS are tracking interest points, and perceptual grouping of pixels from all frames. There is a number of unsatisfying aspects about both of them. Point-based approaches group the trajectories of keypoints with similar motions [2, 8]. However, tracking points yields only a confidence map of the objects' vicinity – not segmentation. To improve robustness, multiple points that fall within a fixed size and shape window are jointly tracked along a pre-specified number of consecutive frames. These ad hoc choices increase complexity that is proportional to the product of scales and locations of the scanning windows. As interest points do not capture the spatial cohesiveness of objects, this approach usually suffers from multiple overlapping object detections. These are usually resolved by making heuristic assumptions about the numbers, sizes, and shapes of objects present in the video. In the second approach, VOS is formulated as clustering of pixels from all frames. The pixels are represented in a multidimensional feature space spanned over pixel photometric, 2D space, and motion properties. The clustering gathers evidence of pixel groupings simultaneously in all dimensions of the feature space by using, e.g., Gaussian mixture models [10], meanshift [4, 20], spectral clustering [5], and dominant sets [19]. This, however, becomes infeasible for even moderate size videos. Attempts have been made to heuristically split the video into small parts, and use out-of-sample data clustering [5, 19].

In this paper, we adopt an alternative, hybrid formulation. We initially conduct a perceptual grouping of pixels in the spatial domain, i.e., segment each frame, and then track the resulting regions. This is appealing, since there are many fast and effective algorithms for image segmen-

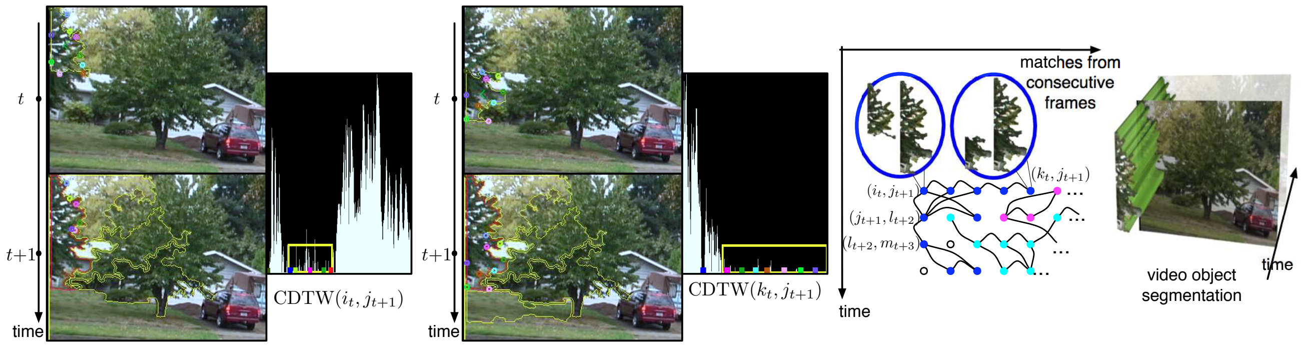

Figure 1. Two example video frames t and $t{+}1$ (shown twice). In frame t, the leftmost tree is split into two segments i_t and k_t (yellow), and in frame $t{+}1$ the tree is one segment j_{t+1} (red). The candidate matches in frame $t{+}1$ of i_t and k_t are marked yellow. The two plots show the cost of matching boundary pixels of (i_t, j_{t+1}) and (k_t, j_{t+1}). CDTW identifies the longest, best matching boundary parts (yellow rectangles and control points). The region pairs are transitively clustered across the video if they are spatially adjacent and have similar intrinsic and motion properties. Each cluster corresponds to a discovered and delineated object occurring in the video.

tation. Also, there are, in general, fewer segments than interest points to be tracked, which will allow us to handle relatively long videos. The higher descriptive power of regions relative to points can be efficiently used to specify more robust tracking algorithms. Since region boundaries delineate objects, tracking the right combination of regions will immediately result in VOS, unlike tracking points.

Despite the aforementioned advantages, there is a very limited work on VOS by tracking regions [7, 15, 3]. This, in part, is due to the well-known irrepeatability of image segmentation across the video sequence. In particular, a low-contrast boundary between two regions in one frame may be undetected in the subsequent frame, resulting in a merger of the low-contrast regions. Conversely, a large region that contains small variations of brightness may be split into a set of smaller, more homogeneous regions in the next frame. The merges and splits cause changes in the number, size, shape, and layout of regions in two consecutive frames showing the same object, as illustrated in Fig. 1. Therefore, any region tracking algorithm that makes the following assumptions: (i) properties of regions remain nearly the same between two consecutive frames, and (ii) for every region in one frame there exists a single corresponding region in another frame (i.e., one-to-one correspondence) will yield poor performance. Existing methods to region tracking are often based on these two assumptions [7, 15, 3].

1.1. Contributions

This paper presents important properties of regions that remain invariant even under the aforementioned merges and splits in segmentation. We make use of these invariances, and thus enable robust region tracking. In particular, the merging and splitting may occur only between spatially adjacent regions (whose boundaries touch). The merging happens along shared, low-contrast boundaries – whereas the remaining, non-shared boundaries remain intact, as shown in Fig. 1. Also, the splitting introduces new boundaries, but does not change the old ones. Therefore, identifying parts of region contours that remain intact from frame to frame will facilitate region tracking.

To meet reasonable runtime requirements, it is critical that the matching of region contours be computationally efficient. To this end, we formulate a new circular dynamic time warping (CDTW) algorithm. CDTW identifies the longest, best matching boundary parts of two regions. It generalizes the well-known DTW to cyclic sequences [17]. CDTW first estimates the optimal position where to break and thus open the two region contours, and then applies DTW to these two sequences of pixels. The optimal breaking points are estimated in terms of the cumulative cost of all pixel matches along the two contours. In particular, we select the pair of points with the maximum likelihood that the associated cumulative cost is minimum and most stable. CDTW preserves all the attractive properties of DTW. That is, CDTW achieves the optimal solution with complexity that is linear in the number of pixels in the two contours.

1.2. Overview of Our Approach

The block-diagram of our approach is shown in Fig. 1. *Step 1*: Given a video, each frame is segmented into regions by using any available low-level segmenter (e.g., mean-shift). *Step 2*: Regions from every two consecutive frames are matched. To avoid matching redundant, unlikely region pairs, we find for each region i in frame t its matching candidates in frame $t+1$. The candidates are those regions that overlap with the estimated area of i's displacement in frame $t{+}1$. We use the standard Lucas-Kanade method to estimate the likely area of i's displacement in the next frame. *Step 3*: Contours of the candidate region pairs are matched by using CDTW. The matched contour parts are interpreted as unaffected by region merging and splitting, and their associated similarity is used as the similarity of photometric and geometric properties of the two regions. We also estimate motion parameters of the region pair from the displacement of their matched boundary parts. *Step 4*: The matched region pairs are transitively clustered across the video if they are spatially adjacent and similar in terms of photometric, geometric, and motion properties. The clustering is formulated as relaxation labeling (RL) over the region pairs, which simultaneously identifies all clusters without requiring any input parameters. Both CDTW and this clustering are aimed at overcoming segmentation instability across the video. Each cluster represents a spatially and temporally coherent tube of the video volume, i.e., a discovered and delineated object present in the video.

Note that our approach is intended to serve as an initial computational step of a wide range of higher-level algorithms. Therefore, it is based purely on low- and mid-level cues. We do not make any specific assumptions about: object intrinsic, layout, and motion properties, number of objects, and camera motion that are common in prior work. In addition, we do not require any input parameters (except those required by image segmentation). As a result, the extracted space-time video tubes may correspond to distinct objects, but also to groups of objects (or parts) if members of the group are adjacent and have similar motions. A top-down inference may subsequently improve our purely bottom-up results with much less effort than if it were to start from all pixels or interest points of all frames. Our approach is capable of handling partial occlusion, scale changes, and in-plane rotation. Complete occlusions and abrupt affine transforms usually require higher-level reasoning which is beyond our scope.

In the sequel, Sec. 2 describes CDTW, Sec. 3 explains region clustering, and Sec. 4 presents experimental results.

2. CDTW

This section presents our Step 3 – region matching – where the goal is to identify the longest, best matching boundary parts of two regions. Such formulation of region matching is aimed at addressing segmentation instability, because parts of region boundaries are invariant to the region merging and splitting. A large volume of work on shape matching is not suitable for our purposes, due to the associated high complexity. Matching point clouds representing region boundary pixels can be computationally efficient [13]. However, this approach ignores an important cue that boundary pixels form an ordered sequence in the image. Therefore, it is typically inferior to methods based on aligning pixel sequences [12, 11]. These methods are based on DTW that guarantees the optimal alignment of two open contours with M and N pixels, with complexity $O(MN)$. Because of these attractive properties, in this paper, we seek to extend DTW to cyclic sequences, and use it for region matching. Below, we first review the classical DTW and some of its generalizations, and then present our CDTW.

Define two sequences of points, $\boldsymbol{b}_i{=}\{b_{i1},..,b_{iM}\}$ and $\boldsymbol{b}_j{=}\{b_{j1},..,b_{jN}\}$. Let $f{=}\{(b_{iu},b_{jv}) : u{=}1..M, v{=}1..N\}$, denote a many-to-many mapping between $\boldsymbol{b}_i$ and $\boldsymbol{b}_j$. Also, let $c(b_{i\cdot},b_{j\cdot})$ denote the cost of matching two points. Given start and end matches, (b_{i1},b_{j1}) and (b_{iM},b_{jN}), DTW finds the optimal many-to-many matching of the remaining points, f^*, that respects their ordering and is characterized by the minimum total cost, $f^* = \arg\min_f \sum_f c(b_{iu},b_{jv})$. DTW first constructs an $M{\times}N$ cumulative cost matrix, C, whose each element $c(b_{iu},b_{jv})$ is recursively computed via its vertical, horizontal, and diagonal neighbors $c(b_{iu},b_{j(v-1)})$, $c(b_{i(u-1)},b_{jv})$, $c(b_{i(u-1)},b_{j(v-1)})$. Then, DTW uses the dynamic programming to find the minimum cumulative-cost path, $\boldsymbol{\pi}^*$, between $c(b_{i1},b_{j1})$ and $c(b_{iM},b_{jN})$ in C, as illustrated in Fig. 2. Elements of C that are included in $\boldsymbol{\pi}^*$ represent the optimal matches in f^*, and the associated minimum cost $c(\boldsymbol{\pi}^*) = \sum_{f^*} c(b_{iu},b_{jv})$. Note that vertical or horizontal moves in $\boldsymbol{\pi}^*$ are the result of many-to-many matching. This allows robust matching of contours with different lengths and deformations.

There are a number of extensions of DTW for matching cyclic sequences that guarantee the optimal solution. For example, a brute-force approach, referred to as BCDTW, recomputes DTW for every cyclic shift of one of the two sequences, increasing complexity to $O(MN^2)$. Also, A*-like algorithms add new source and termination elements to C, and then search for $\boldsymbol{\pi}$ between these two new elements [12, 11]. Their complexity is between $O(MN\log N)$ and $O(MN^2)$. However, for our video object segmentation, this complexity is too prohibitive. There are also a number of extensions that do not guarantee the optimal solution, but have low complexity. For example, ACDTW [1] achieves currently the best accuracy vs. complexity tradeoff in matching MPEG-7 silhouettes. ACDTW concatenates C to itself, $[CC]$, and then searches for the optimal path under a number of ad hoc constraints; e.g., it favors the path slope to be N/M, and penalizes a large number of successive vertical or horizontal moves in the path.

Unlike the related work, our objective is to avoid any heuristic assumptions and preserve complexity of $O(MN)$ in deriving our CDTW. We reason that if the start and end points of $\boldsymbol{b}_i$ and $\boldsymbol{b}_j$ were known, we could directly use DTW with its nice properties. Therefore, we formulate CDTW as the problem of identifying the start and end points of $\boldsymbol{b}_i$ and $\boldsymbol{b}_j$. Note that it only suffices to identify a single true match between any two points b_{iu} and b_{jv} along the optimal path, and then declare them as the start points; b_{iu} and b_{jv} are also the end points, because $\boldsymbol{b}_i$ and $\boldsymbol{b}_j$ are cyclic. Below, we first introduce some notation, and then derive our CDTW.

Let $\boldsymbol{\pi}^*$ denote the optimal path in C of two closed contours $\boldsymbol{b}_i$ and $\boldsymbol{b}_j$. Let $m{=}(b_{i\cdot},b_{j\cdot})$ denote a pair of points from the two contours, and C_m denote all elements of C

in the vicinity of m. By construction of C, for any m in C, we immediately know a bottom-up path, $\boldsymbol{\pi}_{\uparrow m}$, from the bottom row of C to m with the minimum cumulative cost, as well as a top-down path, $\boldsymbol{\pi}_{\downarrow m}$, from the top row of C to m with the minimum cumulative cost. Let $\boldsymbol{\pi}_m$ denote the union of these paths, $\boldsymbol{\pi}_m = \boldsymbol{\pi}_{\uparrow m} \cup \boldsymbol{\pi}_{\downarrow m}$. Note that even if $m \in \boldsymbol{\pi}^*$, in general, $\boldsymbol{\pi}_m \neq \boldsymbol{\pi}^*$, because $\boldsymbol{\pi}^*$ starts and ends at the same element of C, whereas $\boldsymbol{\pi}_{\uparrow m}$ and $\boldsymbol{\pi}_{\downarrow m}$ start from matrix elements different from m.

The main rationale behind our derivation of CDTW is that if m lies on $\boldsymbol{\pi}^*$, or is located in a vicinity of $\boldsymbol{\pi}^*$, then it is very likely that $\boldsymbol{\pi}_m$ and $\boldsymbol{\pi}^*$ will share many elements of C, $\boldsymbol{\pi}_m \sim \boldsymbol{\pi}^*$. This means that the cumulative costs of these paths will be relatively small and close $c(\boldsymbol{\pi}_m) \simeq c(\boldsymbol{\pi}^*)$. As a corollary, if $m \in \boldsymbol{\pi}^*$, for elements of C in the vicinity of m, $n \in C_m$, we expect that $\boldsymbol{\pi}_n \sim \boldsymbol{\pi}^*$, and $c(\boldsymbol{\pi}_n) \simeq c(\boldsymbol{\pi}_m) \simeq c(\boldsymbol{\pi}^*)$. Otherwise, if $\boldsymbol{\pi}_m$ and $\boldsymbol{\pi}_n$ were very different from $\boldsymbol{\pi}^*$ in this case, it would mean that $\boldsymbol{\pi}^*$ consistently discards the minimum cost elements of C (the strict proof of this claim is beyond our scope). Conversely, if m and n lie in the remote areas of C from $\boldsymbol{\pi}^*$ then $c(\boldsymbol{\pi}_m) > c(\boldsymbol{\pi}^*)$ and $c(\boldsymbol{\pi}_n) > c(\boldsymbol{\pi}^*)$, otherwise $\boldsymbol{\pi}^*$ is not optimal.

From the above rationale, to estimate the optimal position where to break and open closed contours $\boldsymbol{b}_i$ and $\boldsymbol{b}_j$, we find an element of C, m^*, with the maximum probability that it lies on the optimal path $\boldsymbol{\pi}^*$. This probability is maximum if the cost of the associated "greedy" path $c(\boldsymbol{\pi}_{m^*})$ is minimum and stable, i.e., if all elements n of C in the vicinity of m^*, $n \in C_m$, also generate similarly minimum costs $c(\boldsymbol{\pi}_n)$. More formally:

$$\begin{aligned} m^* &= \max_{m \in C} P(m \in \boldsymbol{\pi}^*), \\ &= \max_{m \in C} P(\{\boldsymbol{\pi}_m \sim \boldsymbol{\pi}^*\}, \{\forall n \in C_m, \boldsymbol{\pi}_n \sim \boldsymbol{\pi}_m\}), \\ &= \max_{m \in C} P(\{\boldsymbol{\pi}_m \sim \boldsymbol{\pi}^*\}) \prod_{n \in C_m} P(\{\boldsymbol{\pi}_n \sim \boldsymbol{\pi}_m\} | \{\boldsymbol{\pi}_m \sim \boldsymbol{\pi}^*\}). \end{aligned} \tag{1}$$

The probabilities used for computing (1) are specified using the above rationale, as

$$\begin{aligned} P(\{\boldsymbol{\pi}_m \sim \boldsymbol{\pi}^*\}) &\propto \exp(-\mu c(\boldsymbol{\pi}_m)), \\ P(\{\boldsymbol{\pi}_n \sim \boldsymbol{\pi}_m\} | \{\boldsymbol{\pi}_m \sim \boldsymbol{\pi}^*\}) &\propto \exp(-\lambda |c(\boldsymbol{\pi}_n) - c(\boldsymbol{\pi}_m)|), \end{aligned} \tag{2}$$

where $1/\lambda$ and $1/\mu$ are the ML estimates of the mean values of the corresponding exponential distributions over the elements of C.

After identifying m^*, we run the standard DTW on C starting from m^*, and thus obtain $\boldsymbol{\pi}^*$. The associated cost $c(\boldsymbol{\pi}^*)$ is directly used as the cost of matching the two regions. For small, fixed neighborhoods C_m, the complexity of CDTW is $O(MN)$.

In our implementation, we use the following definition of the cost of matching two boundary pixels, $c(b_{iu}, b_{jv})$. At each pixel u of the boundary of region i, we compute the standard log-polar shape-context descriptor with 21 bins. The shape descriptor is aligned with the image's gradient direction at u on the boundary. The size of the descriptor is dynamically adjusted for every region to be a small percentage (1%) of the region size. This allows us a certain degree of scale and in-plane-rotation invariance. The shape descriptor and boundary contrast at u together form vector b_{iu}, and thus we compute $c(b_{iu}, b_{jv}) = \|b_{iu} - b_{jv}\|$. Fig. 2 shows that our CDTW outperforms the method ACDTW of [1], with significantly lower complexity.

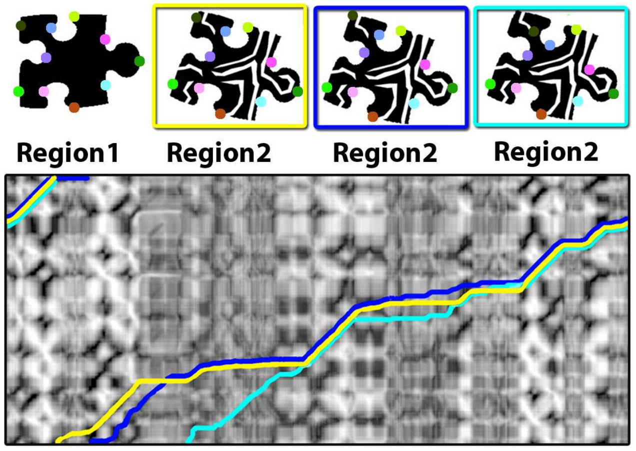

Figure 2. The cost matrix of matches between boundary pixels of region 1 and region 2. The matching results of three algorithms are shown as three folded paths in the cost matrix, and in the three rectangles containing region 2. The corresponding paths and rectangles have the same color. The methods BCDTW (blue path and rectangle) and ACDTW [1] (cyan path and rectangle) yield matching errors (see the control points). Our CDTW (yellow path and rectangle) has no matching errors despite serious challenges.

3. Region Tracking

This section presents our Step 3, where the goal is to cluster similar, adjacent regions across the video. The similarity is defined in terms of region photometric, geometric, and motion properties. In Sec. 2, we explained how to compute the cost of matching region photometric and geometric properties, $c(i,j) = c(\boldsymbol{\pi}^*)$, by using CDTW. The similarity of this matching is defined as $s(i,j) = \exp(-c(i,j))$. Next, we explain how to estimate the similarity of motion parameters characterizing region pairs.

If i moves so that it appears as region j in the next frame, then the matched pairs of boundary pixels of i and j can be used to estimate i's motion parameters. We define these motion parameters as the homography matrix $H(i,j)$. $H(i,j)$ is estimated using the standard RANSAC algorithm on the pixel matches that lie along the optimal path $\boldsymbol{\pi}^*(i,j)$, found by CDTW. The similarity of the motions of region pairs (i,j) and (k,l), denoted as $s(i,j,k,l)$, is estimated by using the Frobenius norm $s(i,j,k,l) = \exp(-\|H(i,j) - H(k,l)\|_F)$.

The similarities $s(\cdot)$ are used for tracking regions across the video. To this end, as mentioned in Sec. 1, we compute $s(\cdot)$ for each region i_t in frame t and its allowed candidate matches $\{j_{t+1}, k_{t+1}, \dots\}$ in frame $t{+}1$. Then, we construct an attributed graph, G, that will allow us to transitively cluster similar and adjacent regions. In G, nodes are the region pairs, and arcs connect the region pairs whose respective members are the same or spatially adjacent (i.e., boundaries touch), as illustrated in Fig. 1. For example, node (i_t, j_{t+1}) is connected to (i_t, l_{t+1}), (k_t, j_{t+1}), (j_{t+1}, m_{t+2}), because regions i_t and j_{t+1} appear in these graph nodes. Also, (i_t, j_{t+1}) is connected to (n_t, o_{t+1}) if regions i_t and n_t are spatially adjacent, or j_{t+1} and o_{t+1} are adjacent. Such graph connectivity facilitates the transitive grouping of only spatially adjacent regions into a single cluster. In G, the attribute of each node (i,j) is $s(i,j)$, and the attribute of each arc $((i,j),(k,l))$ is $s(i,j,k,l)$.

After constructing G, we cluster the regions from all video frames by using relaxation labeling (RL) on G. It is worth noting that we have tried to use other formulations of graph partitioning. For example, we have implemented loopy belief propagation on the MRF, using G and the unary and pairwise potentials defined in terms of $s(\cdot)$. In all our experiments, RL is faster and outperforms the loopy-BP formulation. Recent formulations of graph partitioning based on learning the potential functions are not applicable for our purposes, because, as mentioned in Sec. 1, our approach is aimed at addressing unsupervised object segmentation in arbitrary videos. Below, we explain our RL formulation.

To simplify notation, we will use letters a and b to denote nodes of G (e.g., $a = (i,j)$ and $b = (k,l)$). RL assigns label $\lambda_a \in \{0,1\}$ to each $a \in G$. All nodes that are connected by arcs in G and have $\lambda_a = 1$ are grouped within the same cluster. Nodes that are not connected in G and have $\lambda_a = 1$ belong to different clusters. All nodes with $\lambda_a = 0$ do not belong to any cluster. Thus, RL simultaneously identifies all clusters present in G, without requiring any input parameters. For each $a \in G$, RL iteratively computes the likelihood $p(\lambda_a)$ by collecting evidence on the labels of neighboring nodes of a, $N(a){=}\{b : b{\neq}a,\ a,b \text{ are connected by an arc in } G\}$, as:

$$\begin{aligned} p(\lambda_a) &\leftarrow p(\lambda_a)[1{+}q(\lambda_a)]/\textstyle\sum_{\lambda_a} p(\lambda_a)[1{+}q(\lambda_a)], &(3)\\ q(\lambda_a) &= \textstyle\sum_{b\in N(a)} \sum_{\lambda_b} r(\lambda_a,\lambda_b) p(\lambda_b) &(4)\end{aligned}$$

where $r(\lambda_a,\lambda_b)$ is the correlation between the events that a has label λ_a and b has label λ_b. $r(\cdot)$ is defined as a function of costs $c(\cdot)$ and $h(\cdot)$. Also, we have that $r(\cdot){\in}[-1,1]$.

To define $r(\cdot)$ we use the following intuition. If both shapes and motions of region pairs a and b are similar then a and b should belong to the same cluster, i.e., $r(1,1) = 1$; otherwise, $r(1,1)$ should work to repel a from b so that they do not end up in the same cluster, i.e., $r(1,1) = -1$. Also, if region pair a is a good match and b is a poor match, or vice versa, then a and b should belong to the same cluster only if they move the same, i.e., $r(1,0) = -1$ and $r(0,1) = -1$; otherwise, if a and b move differently then RL should keep them in separate clusters, i.e., $r(1,0) = 1$ and $r(0,1) = 1$. Finally, if a and b are complete mismatches in terms of shapes and motions then they should not be clustered together, i.e., $r(0,0) = 1$. Suppose for the moment that the similarities are Boolean variables, $s(a), s(b), s(a,b) \in \{0,1\}$. By listing exhaustively all 0 and 1 combinations of $s(a), s(b), s(a,b)$, and the resulting suitable values of $r(\cdot)$ we arrive at a disjunctive normal form (DNF) that is equivalent to the following Boolean formula:

$$r(\lambda_a,\lambda_b) = 2\left[[s(a)\cdot s(b)] \text{ XOR } \overline{s(a,b)}\right] - 1\,. \quad (5)$$

Real values of the correlation, $r(\cdot){\in}[-1,1]$, can be obtained by defining $\overline{s(\cdot)}{=}1{-}s(\cdot)$ and using real values of $s(\cdot)$ in (5).

The likelihoods $p(\lambda_a)$, $\forall a{\in}G$, are initialized to random values. After convergence, RL assigns the maximum-likelihood label to each a in G. Each obtained cluster of the region pairs represents a spatially and temporally coherent subvolume of the 3D space-time video volume. CDTW of two regions whose boundary length is 10^4 pixels takes about 0.2sec in our C-implementation on a 3GHz 2GB RAM PC. All steps of our VOS (excluding low-level segmentation) on 100 frames, each of size $512{\times}512$, take about 30sec in the same implementation.

4. Results

This section presents our quantitative and qualitative evaluation on 94 videos. For quantitative evaluation, we use the well-known *Activity* database [9] that consists of 90 videos of 10 distinct human activities (e.g., walking, jumping-jack, hand-waving, bending, etc.). Each activity has 9 videos, and each video has between 30 and 120 frames. The videos are manually segmented into the foreground (person) and background to obtain the ground truth VOS. In *Activity* videos, the person moves in front of a fairly uniform, static background. This dataset tests our invariance to object articulation. We also use the following four videos for qualitative evaluation and comparison: (1) *Kwan* ice-skater video; (2) *House* sequence from CMU motion database; (3) *Coastguard* video; and (4) *Garden* video. These four videos introduce additional challenges: articulated object motions amidst the cluttered background (in *Kwan*), affine transforms (in *House*), presence of dynamic texture and non-static camera (water surface in *Coastguard*), camera motion and layered motions (in *Garden*). Also, the videos contain shadows and motion blur that negatively affect low-level segmentation.

Quantitative results: Most prior work on VOS presents only qualitative evaluation. Note that estimating the accuracy of tracking of points associated with a moving object does not evaluate VOS. We use *Activity* videos to estimate: (i) the average VOS error ϵ; and (ii) how ϵ varies as a function of the total number of segments per frame. We also consider two common low-level segmentation algorithms – the meanshift and N-cuts – in order to evaluate our performance vs. the quality of the low-level segmenter used. ϵ is computed for each object occurring in the 3D video volume. To this end, we consider the ground-truth (manually annotated) subvolume of the object and all subvolumes extracted by our approach. Let G denote all pixels from all frames that belong to the ground-truth subvolume representing the object. Also, let D denote all pixels from all frames that belong to a subvolume extracted by our approach. Given G, we automatically select D that has the largest overlap with G. If the ground-truth subvolume G extends over more frames than the selected D, then we repeat the selection of the best overlap with G over the remaining time intervals. This is repeated until the entire G is covered. Note that at each time instance only one D is selected. Then, we compute $\epsilon(G) = \sum_{\text{intervals}} \min_D \frac{\text{XOR}(G,D)}{G \cup D}$.

Tables 1 and 2 show ϵ estimates on *Activity* videos, computed in the following four cases: the frames are only segmented by the meanshift and N-cuts, and these low-level segmentations are processed by our approach. Note that Tables 1 and 2 do not serve to compare our approach with meanshift and N-cuts, but to quantify how much we improve object segmentation performance when we account for the region splits and mergers, and motion information. The manual annotation of *Activity* videos labels 1 object in the foreground (moving person) and 3 objects in the background (ground, wall, window in the wall). As can be seen, our approach succeeds in producing VOS that is spatially and temporally coherent, despite a large degree of region merging and splitting in the two base low-level segmentations. Averaged over all 90 *Activity* videos, the meanshift (resp. N-cuts) produces ϵ=65% (resp. ϵ=12.5%) on the foreground, whereas our algorithm yields only ϵ=3.5% (resp. ϵ=0.30%) on the foreground.

Fig. 3 shows how ϵ varies as the number of regions present in the N-cuts segmentation increases. As can be seen, oversegmentation has negative effects on our performance. The main reason is that N-cuts produces more unstable segmentations across the video frames as this number increases, because new regions do not correspond to new objects, and thus do not have repeatable boundaries.

Qualitative evaluation: Fig. 4 illustrates that our approach gives coherent VOS on example videos from *Activity* database and *Coastguard* video, despite a large degree of instability of the meanshift segmentation across the frames. The camera motion in *Coastguard* video does not affect our results, because the superposition of the camera and boat motions still produces a spatially and temporally coherent subvolume in the video.

Fig. 5 shows comparison with the approach of [13] that uses loopy-BP based many-to-many matching of meanshift regions across the video, but ignores region motion parameters. For fair comparison, our results presented in Fig. 5 are also obtained by ignoring the motion information in the expression (5), simplifying our approach only to CDTW and RL on adjacent pairs of regions. As can be seen, our simplified approach gives more coherent VOS on these examples. Fig. 6 shows comparison with the approach of [19] that uses dominant sets to conduct out-of-sample, space-time clustering of all pixels from the video. As expected, we are more successful in delineating the boundaries of the entire tree in each frame of *Garden* video, whereas these contours flicker in the dominant-sets based VOS. The meanshift experiences difficulties in segmenting the image area occupied by flower

	Background		Foreground	
Videos	MeanShift	Ours	MeanShift	Ours
Jack	14.03%	0.98%	59.11%	0.51%
Run	30.70%	0.35%	73.10%	5.39%
Skip	14.59%	0.53%	73.21%	5.52%
Walk	8.18%	0.68%	54.76%	2.51%
10 activities	**16.88%**	**0.64%**	**65.04%**	**3.48%**

Table 1. The VOS error ϵ averaged over the 9 videos for each human activity, and over 90 videos of all 10 activities from *Activity* database, using: (1) the meanshift only; and (2) our approach with the meanshift.

	Background		Foreground	
Videos	NCut	Ours	NCut	Ours
Bend	14.67%	3.92%	18.52%	0.03%
Jump	20.57%	9.24%	16.34%	0.05%
PJump	10.93%	2.27%	0.30%	0.30%
Side	21.93%	7.09%	12.92%	0.73%
Wave-1	15.95%	7.57%	3.89%	0.42%
Wave-2	12.71%	7.36%	23.14%	0.28%
10 activities	**16.13%**	**6.24%**	**12.52%**	**0.30%**

Table 2. (Same as Tab 1): (1) N-cuts only; and (2) our approach with N-cuts.

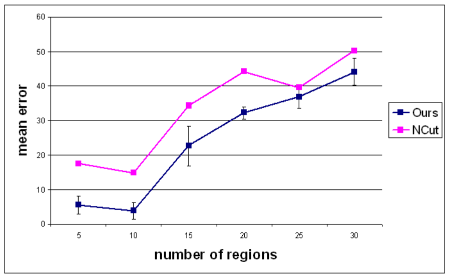

Figure 3. The VOS error ϵ on the foreground averaged over the 90 *Activity* videos vs. the total number of regions per frame, using: (1) N-cuts only and (2) our approach with N-cuts.

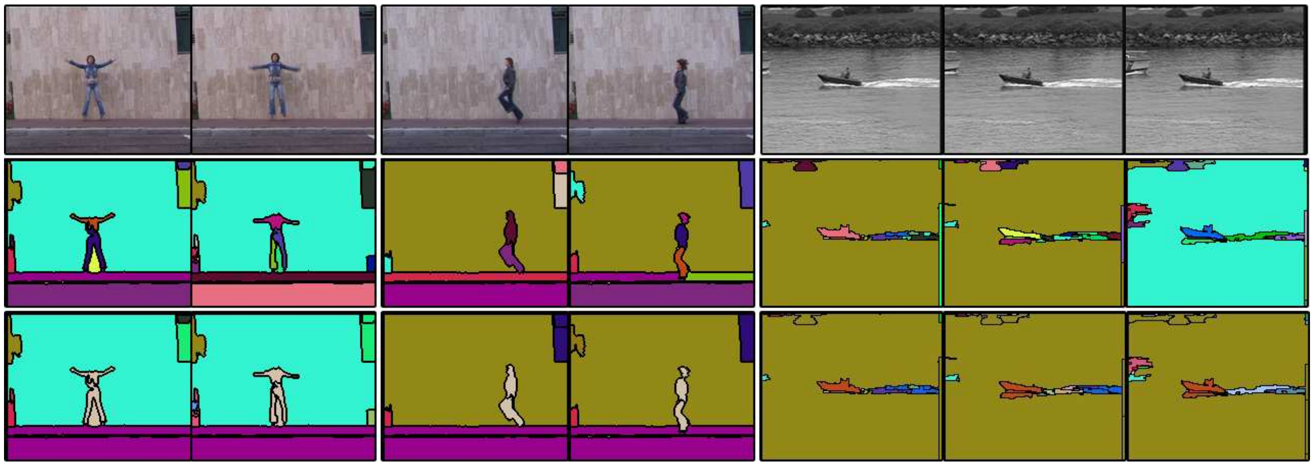
Figure 4. VOS of jumping-jack and skip videos from *Activity* database, and *Coastguard* video: (top row) original sequence; (middle row) VOS using only the meanshift of each frame; (bottom row) our approach with the meanshift. The extracted subvolumes are marked with unique colors. We succeed in identifying that regions occupied by objects (person or coastguard boat) and split in the meanshift should be tracked together. We also improve VOS on dynamic-texture areas, e.g., the trace of the boat's motion on the water surface.

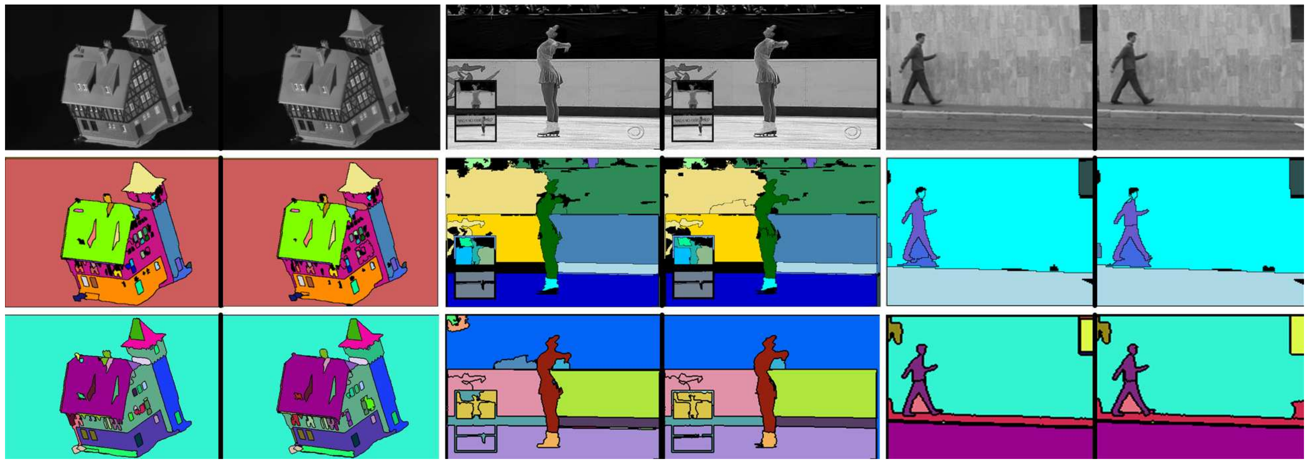
Figure 5. Comparison with [13] on *House*, *Kwan*, and the walk video from *Activity* database without accounting for the motion information: (top row) original sequence; (middle row) results of [13]; (bottom row) results of our simplified approach (no motion parameters) with the meanshift. The extracted subvolumes are marked with unique colors. We identify and track fewer subvolumes in *House* and *Kwan* than [13]. Also, in the example frames from the walk video, the approach of [13] merged the person's head with the background, whereas we successfully delineate the entire person in each frame.

texture. It produces numerous, unstable regions that are too small to have any characteristic shape differences. These small, blobby regions are easily confused by CDTW as similar, which in turn produces errors in their tracking. We anticipate that the use of a more sophisticated image segmentation algorithm than the meanshift will improve our VOS on such highly textured image areas.

5. Conclusion

We have argued that video object segmentation (VOS) by tracking regions has many fundamental advantages over the approaches based on tracking points or jointly clustering of all pixels from all video frames. Despite these advantages, one of the major obstacles toward successful region-based VOS seems to be a high degree of irrepeatability of image segmentation characterizing currently available low-level segmentation algorithms. We have presented important region invariances to this irrepeatability – namely, that parts of region boundaries remain intact under the region merging and splitting. This observation has been used to formulate matching regions across the frames as identifying parts of region boundaries that match. We have generalized DTW to CDTW that is capable of optimally matching closed region contours with linear complexity, without resorting to heuristic assumptions. Tracking regions by using our CDTW has been shown to produce competitive results

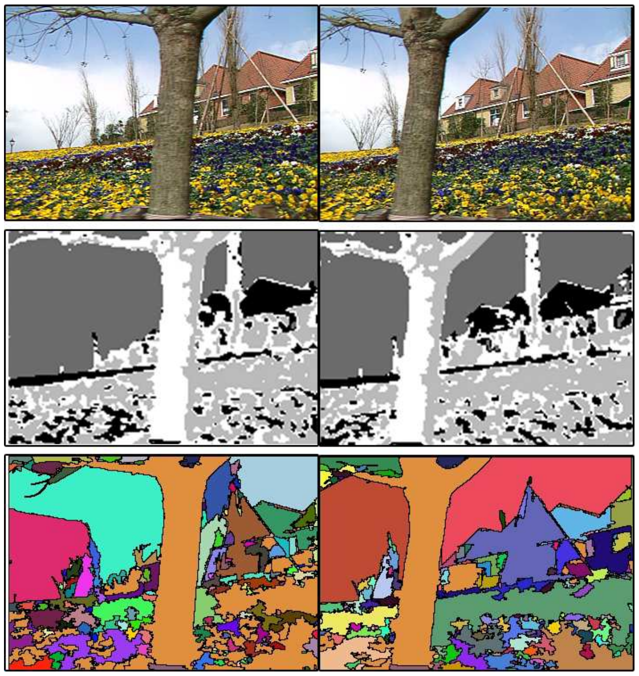

Figure 6. Comparison with [19] on *Garden* sequence: (top row) original sequence; (middle row) results of [19]; (bottom row) results of our approach with the meanshift. The extracted subvolumes are marked with unique colors. While the approach of [19] merges the right part of the tree with the background, and suffers from unreliable (flickering) detection of the tree contours, we successfully delineate the entire tree in each frame. We fail to track the textured surface of flowers as a whole, because the meanshift is very unstable in this area.

in comparison with the state of the art.

Acknowledgment

We would like to thank Prof. Alan Fern and the anonymous reviewers whose thoughtful comments and suggestions improved the quality of the paper.

References

[1] N. Arica. Cyclic sequence comparison using dynamic warping. In *CIVR*, 2005. 3, 4

[2] M. Brand and V. Kettnaker. Discovery and segmentation of activities in video. *IEEE TPAMI*, 22(8):844–851, 2000. 1

[3] J. P. Collomosse, D. Rowntree, and P. M. Hall. Stroke surfaces: Temporally coherent artistic animations from video. *IEEE Trans. Visualization Computer Graphics*, 11(5):540–549, 2005. 1, 2

[4] D. DeMenthon and R. Megret. Spatio-temporal segmentation of video by hierarchical mean shift analysis. In *ECCV Workshop Statistical Methods in Video Processing*, 2002. 1

[5] C. Fowlkes, S. Belongie, F. Chung, and J. Malik. Spectral grouping using the Nystrom method. *IEEE TPAMI*, 26(2):214–225, 2004. 1

[6] E. Galmar and B. Huet. Analysis of vector space model and spatiotemporal segmentation for video indexing and retrieval. In *CIVR*, pages 433–440, 2007. 1

[7] M. Gelgon and P. Bouthemy. A region-level motion-based graph representation and labelling for tracking a spatial image partition. In *EMMCVPR*, 1997. 2

[8] D. B. Goldman, C. Gonterman, B. Curless, D. Salesin, and S. M. Seitz. Video object annotation, navigation, and composition. In *UIST*, pages 3–12, 2008. 1

[9] L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri. Actions as space-time shapes. *IEEE TPAMI*, 29(12):2247–2253, 2007. 1, 5

[10] H. Greenspan, J. Goldberger, and A. Mayer. A probabilistic framework for spatio-temporal video representation. In *ECCV*, 2002. 1

[11] J. Gregor and M. Thomason. Efficient dynamic-programming alignment of cyclic strings by shift elimination. *Pattern Recognition*, 29(7):1179–1185, July 1996. 3

[12] J. Gregor and M. G. Thomason. Dynamic programming alignment of sequences representing cyclic patterns. *IEEE TPAMI*, 15(2):129–135, 1993. 3

[13] V. Hedau, H. Arora, and N. Ahuja. Matching images under unstable segmentations. In *CVPR*, 2008. 3, 6, 7

[14] M. Irani, B. Rousso, and S. Peleg. Computing occluding and transparent motions. *Int. J. Comput. Vision*, 12(1):5–16, 1994. 1

[15] F. Moscheni, S. Bhattacharjee, and M. Kunt. Spatiotemporal segmentation based on region merging. *IEEE TPAMI*, 20(9):897–915, 1998. 2

[16] M. Nicolescu and G. Medioni. A voting-based computational framework for visual motion analysis and interpretation. *IEEE TPAMI*, 27(5):739–752, 2005. 1

[17] H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Trans. Acoustics, Speech, Signal Processing*, 26(1):43–49, 1978. 2

[18] J. Shi and J. Malik. Motion segmentation and tracking using normalized cuts. In *ICCV*, pages 1154–1160, Jan 1998. 1

[19] A. Torsello, M. Pavan, and M. Pelillo. Spatio-temporal segmentation using dominant sets. In *EMMCVPR*, 2005. 1, 6, 8

[20] J. Wang, B. Thiesson, Y. Xu, and M. Cohen. Image and video segmentation by anisotropic kernel mean shift. In *ECCV*, 2004. 1

[21] J. Wang, Y. Xu, H. Shum, and M. Cohen. Video tooning. In *SIGGRAPH*, 2004. 1

[22] Y. Weiss. Smoothness in layers: Motion segmentation using nonparametric mixture estimation. *CVPR*, 1997. 1