Interactive Video Segmentation Using Occlusion Boundaries and Temporally Coherent Superpixels

Radu Dondera, Vlad Morariu, Yulu Wang and Larry Davis
University of Maryland
College Park, MD USA
{rdondera, morariu, lsd}@cs.umd.edu, ylwang@umd.edu

Abstract

We propose an interactive video segmentation system built on the basis of occlusion and long term spatio-temporal structure cues. User supervision is incorporated in a superpixel graph clustering framework that differs crucially from prior art in that it modifies the graph according to the output of an occlusion boundary detector. Working with long temporal intervals (up to 100 frames) enables our system to significantly reduce annotation effort with respect to state of the art systems. Even though the segmentation results are less than perfect, they are obtained efficiently and can be used in weakly supervised learning from video or for video content description. We do not rely on a discriminative object appearance model and allow extracting multiple foreground objects together, saving user time if more than one object is present. Additional experiments with unsupervised clustering based on occlusion boundaries demonstrate the importance of this cue for video segmentation and thus validate our system design.

1. Introduction

Video segmentation is the problem of determining precise support regions for the objects in a video. A satisfactory solution would benefit important tasks like object detection from video and human action recognition, and this has motivated much research on unsupervised methods. However, the progress of these methods is fundamentally limited because video segmentation is an underconstrained problem. Motion cues typically reduce ambiguity compared to pure image segmentation, but objects can move too slowly or too quickly or can have too many articulated parts for an automated procedure to correctly separate them. It is clear that guidance from a human operator is needed, but it remains to be seen how to optimally specify it and combine it with techniques relevant to unsupervised video segmentation. Interactive video segmentation work in the Computer Graphics community [21] [3] focused on collecting user annotations to obtain perfect object boundaries in all frames. There is no guarantee that the tradeoff between segmentation quality and user effort is optimal and this suggests exploring solutions that incur a small decrease in quality for a large reduction in effort. While applications like video editing could not use the output of such a system directly, weakly supervised learning of object appearance models from video is highly likely to improve if temporally consistent image segments are available. High level video analyses, e.g. video content description, can also build on less-than-perfect segmentations.

A number of papers [3] [2] [14] cast the interactive segmentation problem as assisted contour tracking, in which the user traces the initial object contour and in subsequent frames the system propagates it and the user corrects it. This scheme does not exploit the global spatio-temporal structure of the video and as a result may require excessive supervision when resolving local ambiguities. We propose a system that incorporates user constraints in a superpixel clustering framework that works on long time intervals and accounts for occlusion boundaries. When an object moves, it typically generates occlusion boundaries and the system will likely produce segments that follow these boundaries without needing supervision. On the other hand, if an object is stationary, unsupervised clustering can fail, but a user defined object mask will likely propagate reliably with optical flow, which will be close to zero and have little noise. In both cases, it is reasonable to predict that our system will collect supervision efficiently: dynamic frames only need minor corrections and static frames, while potentially requiring more interaction, indirectly constrain the labels of many superpixels in nearby video frames.

1.1. Related Work

In the interest of space, we limit our review to recent papers in the video segmentation literature. In unsupervised segmentation, Brox and Malik [4] clustered long term point trajectories, providing a basic framework that other

authors built on. Another important direction was pursued by Grundmann et al. [8], who adapted Felzenszwalb's agglomerative image segmentation [6] to video and produced a segmentation hierarchy. Authors such as [10] built on image segmentation by aggregating the image regions output by generic object segmentation proposal methods like [5].

Some unsupervised methods oversegmented each frame into superpixels and ran clustering on all the superpixels obtained. Levinshtein et al. [11] generated temporally coherent superpixels, built a superpixel graph, and hypothesized multiple superpixel groupings with parametric maxflow. We adopt Levinshtein et al.'s method to generate superpixels, but take occlusion boundaries into account in the affinities of edges between superpixels. Galasso et al. [7] evaluated superpixel affinity measures to be used in a spectral clustering framework. They obtained good performance on the Berkeley moseg dataset, but since only low level information is available when computing the affinity of two neighboring superpixels in the same frame, this affinity cannot reflect more than the presence or absence of an occlusion boundary. Without user supervision, our segmentation method is similar to that of [7], the crucial difference being that it explicitly changes the superpixel graph according to occlusions. Directly incorporating this essential signal makes our system more likely to work on more general videos. Vazquez-Reina et al. [19] computed multiple superpixel segmentations for each frame, generated plausible superpixel tracks and finally labeled individual superpixels as belonging to one of the tracks using a higher order CRF. Their method can oversegment severely, as there is no mechanism to restrict splitting a single object into arbitrarily many superpixel tracks. In contrast, our system upper bounds the number of resulting objects by construction.

At the border between unsupervised and interactive methods is work that assumes the existence of a predefined set of pixel labels. Badrinarayanan et al. [1] modeled the evolution of both the image data and the labels in a video frame with a common set of latent patches. Vijayanarasimhan and Grauman's work [20] is closer to interactive video segmentation: they estimated label propagation errors to select the video frames whose complete annotation minimizes the total user effort. This selection procedure cannot be directly added to our system because the effort to label a frame can vary significantly, according to the quality of the previous segmentation. Furthermore, automatic frame selection would offer little benefit because in our system annotating **any** frame is likely to be efficient.

The interactive video segmentation has received the most attention from the Computer Graphics community. Wang et al. [21] employed hierarchical segmentation to improve system response time and extended a 2D alpha matting scheme to video volumes. Bai et al. [3] used multiple local color and shape classifiers to discriminate between foreground and background pixels in small neighborhoods of the object contour. Their system updated the contour by translating the neighborhood regions according to optical flow, applying the classifiers and then retraining them from new masks; user corrections were integrated via classifier retraining. Later work by authors in the same group [2] modeled foreground and background appearance in a scale adaptive manner, updating color space Gaussians for individual pixels using probabilistic optical flow. Price et al. [14] proposed additional types of local classifiers, such as color adjacency relations, but more importantly they dynamically weighted them to adapt to change. Although in [3] [2] and [14] updating the object contour frame by frame with help from the user is intuitive, such a short temporal neighborhood may require significant avoidable supervision in more complex videos (e.g., with moderate motion or with locally similar foreground and background appearance). Our system enables the use of long temporal contexts by running spectral clustering on superpixels from long video intervals. It is not designed to produce pixel perfect segmentations, but to obtain good results quickly.

1.2. Contribution

Our main contribution is to propose a system for interactive video segmentation that leverages long term motion information to reduce annotation effort. On videos used to evaluate interactive segmentation, we obtain satisfactory results significantly faster than existing systems [14]. Two additional advantages of our system are that it does not assume the objects have discriminative appearance and that it allows extracting multiple objects at the same time, saving annotation work. The secondary contribution of this work is to demonstrate the effectiveness of occlusion cues in video segmentation. On a standard dataset, superpixel clustering using occlusion cues only (no user supervision) has performance comparable to a state-of-the-art unsupervised method based on superpixels [7].

2. System Description

Our system first performs unsupervised clustering in a superpixel graph and then repeated supervised clusterings with more and more annotation from the user. In the preprocessing stage, optical flow is computed [17] forward and back and then temporally coherent superpixel segmentations for each video frame [11] (the target number of superpixels per frame is fixed, S). The next preprocessing operation is to run a simple occlusion detector that fires for forward flow inconsistent with backward flow (inequation 1) or large flow gradient magnitude (inequation 2) [18]. If at least one of the two inequations holds for a pixel, then the pixel is deemed occluded (there is an OR between the conditions). With respect to [18], we change the multiplicative constants, requiring more consistency/allowing for larger

gradient magnitudes:

$$||\mathbf{w}(\mathbf{p})+\mathbf{w}'(\mathbf{p}')||_2^2 > 0.01(||\mathbf{w}(\mathbf{p})||_2^2+||\mathbf{w}'(\mathbf{p}')||_2^2)+0.01 \quad (1)$$

$$||\nabla u(\mathbf{p})||_2^2+||\nabla v(\mathbf{p})||_2^2 > 0.01||\mathbf{w}(\mathbf{p})||_2^2+0.01 \quad (2)$$

where $\mathbf{p}$, $\mathbf{p}'$ are pixel coordinates, $\mathbf{w}$ and $\mathbf{w}'$ are the forward and backward optical flow with components u and v ($\mathbf{p}' = \mathbf{p}+\mathbf{w}(\mathbf{p})$).

2.1. Graph Construction

We divide the input video into consecutive time intervals of F frames (or less, at the end of the video) and run the system independently on each of them. Each interval is represented with a graph of superpixels with two types of edges: within-frame edges that link superpixels in the same frame and between-frame edges that link superpixels in temporally adjacent frames. There are within-frame edges only between superpixels that share a pixel border in the image. Each superpixel has at most one between-frame edge linking it to a superpixel in the next frame, depending on the output of the occlusion detector. If the superpixel has at least one non-occluded pixel, then its center is shifted using optical flow and it is linked to the closest superpixel in the next frame:

$$successor(i, f+1) = \arg\min_{j \in f+1} ||\mathbf{c}_j - (\mathbf{c}_i + \overline{\mathbf{w}}_i)||_2 \quad (3)$$

where $\mathbf{c}_i$ is the center of superpixel i (in frame f) and $\overline{\mathbf{w}}_i$ is the mean optical flow of its non-occluded pixels. If all the pixels of the superpixel are occluded, then there is no link the next frame (note that it will still be linked with neighbors in the current frame and possibly with a superpixel in the previous frame). We do not consider two layer spatial neighborhoods or temporal edges across multiple video frames, like [7]. The segmentation results are slightly more robust when these types of edges are included, but the computation is noticeably slower because the affinity matrix used in spectral clustering becomes more dense. With our choice of F and S, the graph for an interval of analysis can exceed $100,000$ vertexes, so it is computationally infeasible to work with a complete affinity matrix. However, the more important problem is that it is unclear how to define a meaningful distance for superpixels far away in time or image space based just on low level cues such as color or optical flow.

We define the distance between the two superpixels of a within-frame edge in a similar way to [11]:

$$d_{i,j}^w = \min\left(1, \frac{||\overline{\mathbf{w}}_i - \overline{\mathbf{w}}_j||_2}{1+\max(||\overline{\mathbf{w}}_i||_2, ||\overline{\mathbf{w}}_j||_2)}\right) \quad (4)$$

If all pixels in a superpixel are occluded then the mean flow is arbitrarily set to 0, but the graph modifications described at the end of this subsection ensure the distance values are still meaningful. The difference from [11] is the 1 in the denominator, which makes the distance tend to 0 if both motions are small. We define the distance between the two superpixels of a between-frame edge as the average of an overlap distance and a color distance:

$$d_{i,j}^b = \frac{d_{i,j}^o + d_{i,j}^c}{2} \quad (5)$$

The overlap distance $d_{i,j}^o$ is defined using the support of the earlier superpixel shifted by its mean flow and the support of the later superpixel:

$$d_{i,j}^o = 1 - \frac{|\{\mathbf{p}+round(\overline{\mathbf{w}}_i)|\mathbf{p} \in \mathbf{s}_i\} \cap \mathbf{s}_j|}{|\{\mathbf{p}+round(\overline{\mathbf{w}}_i)|\mathbf{p} \in \mathbf{s}_i\} \cup \mathbf{s}_j|} \quad (6)$$

The color distance $d_{i,j}^c$ is defined using the normalized 8-bin rgb histograms of the superpixels:

$$d_{i,j}^c = 1 - \sum_{r,g,b} h_i(r,g,b)h_j(r,g,b) \quad (7)$$

In most situations, the overlap distance is by itself adequate for the purpose of d^b, which is to characterize confidence in a superpixel's most likely temporal successor. The color distance was included to prevent superpixels with erroneous mean optical flow from matching random similar shaped superpixels in the next frame, which can lead to superpixel labels leaking across objects. Both d^w and d^b are in the range $[0,1]$ and are converted to affinities with a Gaussian kernel. Note that the upper bound on the distance effectively caps the affinities from below and avoids noisy superpixel sets with arbitrarily low normalized cut value.

Graph modifications according to occlusion boundaries. Since occlusion boundaries often indicate objects moving separately, the affinities of edges across these boundaries should be small, but this might not be the case in the graph constructed so far. We set the affinities of within-frame edges "cut" by an occlusion boundary to $\epsilon > 0$, overriding the distance defined in Equation 4. To detect "cut" situations, we examine the connected components formed by the non-occluded pixels of the two superpixels of an edge. The edge is defined to be "cut" if, for any of its two superpixels, all the components reaching the common border are less than half the superpixel size, see Figure 1. If a superpixel is completely occluded, then all its within-frame edges are "cut". The criterion is not perfect (Figure 1d), but it allows occlusion boundaries not coinciding exactly with superpixel borders to still influence the graph structure.

2.2. Clustering with Constraints

We use the spectral clustering method of Ng et al. [13], which partially eigendecomposes the symmetrically normalized graph laplacian, row-normalizes the eigenvectors

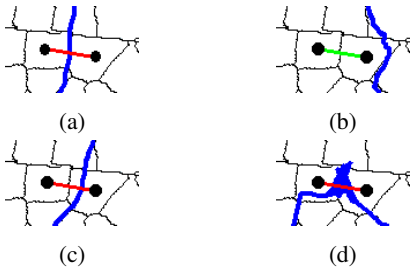


Figure 1: Four representative occlusion cases for a within-frame edge (colored red if it is "cut" by the occlusion boundary and green otherwise). (a) Ideally, the occlusion boundary (blue) includes all the pixels of the superpixel border and the connected components of non-occluded pixels do not contain any. (b) The boundary splits the right superpixel into a large component neighboring the left superpixel and a small component away from it, so the edge is not "cut". (c) The component on the border is small and the occlusion boundary is close to that in (a), so the edge is "cut". (d) Errors of the occlusion boundary detector near the superpixel border lead to the edge being considered "cut".

of the smallest eigenvalues and runs k-means in this space. We incorporate user input in the usual way for constrained clustering, via must-link and cannot-link constraints, but impose these on pairs of nodes already linked by edges, essentially projecting all the annotation information on the original graph structure. The constraints are implemented by changing edge affinities to a large value θ and to 0 for must-link and cannot-link, respectively. This makes spectral clustering strongly prefer to cut the cannot-link edges, as all the other edges in the original graph have cost $\epsilon > 0$, and to avoid cutting must-link edges. Kamvar et al.'s method [9] similarly modifies affinities to 0 or 1; however, we empirically observed the segmentations it returns are not so strongly influenced by the constraints, which tends to reduce the efficiency of the interactive segmentation system. Many other constrained spectral clustering methods have been developed [22, 12, 16, 23], but they target settings in which the graph is relatively small (thousands of nodes), complete and there are few constraints; also, many work for 2 class tasks only. For our video segmentation problem, the graph is large (up to hundreds of thousands of nodes) and sparse, there can be multiple classes (background and more than one object) and the user indirectly specifies a large number of constraints (e.g. on 10% of the graph edges) at once. Rangapuram and Hein's method [15] does work on large graphs, but preliminary tests demonstrated that it cannot be employed in a real time system – on a sample graph for which we computed a segmentation in roughly 7 seconds, their code took 54 minutes.

We set the number of clusters k comparable to but larger

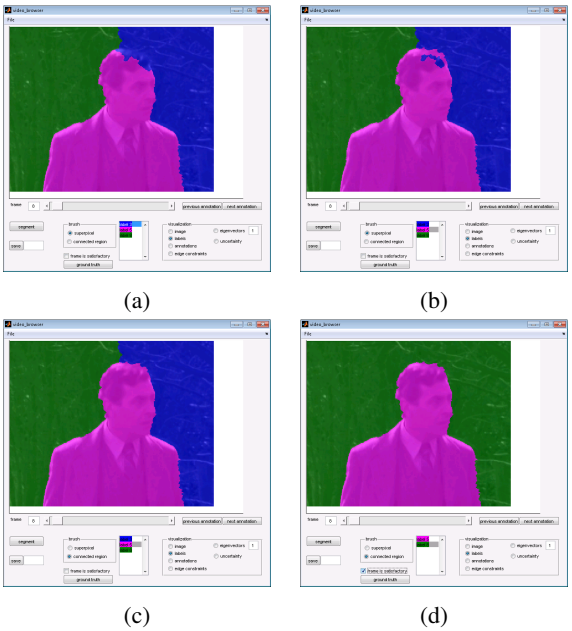


Figure 2: (a) The user browses through the video and finds a frame that is relatively easy to correct. Then, s/he selects a target label and changes superpixels to this label, in one of two modes. In "superpixel" mode, only the clicked superpixel changes; in "connected regions" mode, all the superpixels in a connected component (same label as the clicked superpixel) change. (b) The head contour is completely traced. What is left is to switch to "connected regions" mode and relabel the hole in the head (c) and the right background region (d). A checkbox allows to inform the system that the current frame labeling is satisfactory.

than the number of actual objects in a video. Since the graphs we encounter in practice have noisy small node subsets with normalized cut value lower than for actual objects, clustering with the correct number of segments would return the noisy subsets and an undersegmentation of the rest of the graph, which does not lead to efficient annotation. Instead of extracting the correct number of segments, we prompt the user to assign the over-segmented clusters to objects at the end. It would have been possible to allow the user to change the number of clusters during the interaction, but this can be unintuitive for a non-computer vision/machine learning expert and the total interaction time would have been strongly user-dependent.

2.3. User Interface

The interface displays the current proposed segmentation of the video, the completely unsupervised one in the beginning or one of the constrained ones, following user input. The user navigates through the video to decide which frame(s) to correct, makes the corrections and reruns the

segmentation with an incrementally larger set of constraints, iterating these steps until a satisfactory result is achieved for the entire video. Corrections are specified as new superpixel labels: starting with the previous labels, the user clicks to change the labels of either individual superpixels or of entire connected regions of superpixels with the same label. In the example in Figure 2, the user traces the contour of the head, which was wrongly segmented by the system, fills the small hole and finally "unions" the two background segments. Once the frame labeling becomes satisfactory, the user informs the system (Figure 2d, bottom center of the window); s/he can then request a segmentation given the labels entered so far, or can proceed to correct other frames.

2.4. Temporal Label Propagation

A key feature of our system is that it generates constraints for clustering by temporally propagating the labels of the superpixels in satisfactory frames. Only the between-frame edges of the graph with distance less than d_{max} are used; in the graph with this reduced set of edges, if a labeled node A is the closest among an unlabeled node B's neighbors, then B receives A's label. Starting from a frame marked as satisfactory, the propagation proceeds frame by frame, first forward and then backward in time. The propagation is done independently for each satisfactory frame and then constraints are derived on the original graph edges without having to establish correspondences between labels from different satisfactory frames. There are three possible relations between the two labels assigned to the nodes of an edge as a result of one propagation: same, different and unknown (at least one node does not receive a label). If all satisfactory frames agree in the relation they determine at an edge – at least one 'same' and possibly some 'unknown', but no 'different'; or at least one 'different' and possibly some 'unknown', but no 'same' –, then that edge generates a must-link or cannot-link constraint, respectively. Clearly, the set of constraints depends on d_{max}, but there is a value range that produces large amounts of correct constraints for the videos tested, see Figure 5 and the explanation at the end of Section 3.2.

3. Experimental Results

3.1. Parameter Values

The target number of superpixels per frame is $S = 1,200$ and the number of frames in an interval is $F = 100$. On current PCs, spectral clustering can manage graphs with roughly a hundred thousand vertexes and hundreds of thousands of edges and this choice of S and F was a good compromise between respecting object boundaries and having long temporal context. The Gaussian kernel used to obtain superpixel affinities has $\sigma = 0.4$ and the affinity value for within-frame edges "cut" by occlusion boundaries is $\epsilon = 0.01$. The parameters related to constraints and constraint propagation are $\theta = 1000$ and $d_{max} = 0.25$. In the evaluation videos, with $d_{max} = 0.25$, on average 11.3% of a superpixel graph's edges are constrained after a frame is annotated. The number of clusters is $k = 10$; since the number of objects is from 1 to 3 in the evaluation videos, the amount of extra work for assigning clusters to actual objects is small.

Video	Frames	LiveCut user time (mins)	Our user time (mins)
ballerina	150	74	36
elephant	100	38	13
bass guitar	72	38	14
lemurs	86	36	15
flamingo	76	30	12
manincap	150	23	12
stairs	63	13	8
cat	56	5	5

Table 1: Comparison of interaction times using LiveCut [14] versus using our system. Note that we extract both foreground objects in the "lemurs" sequence while LiveCut extracts just one. Our system is roughly 2-3 times faster, except for the relatively short "cat" sequence, in which long term motion information is not too effective.

3.2. Interactive Video Segmentation

We ran our interactive video segmentation system on eight of the videos used by Price et al. [14] [1]. Our method currently needs preprocessing time on the order of hours for a video, mostly because of optical flow computation. However, this could be easily reduced to minutes by running on a computer cluster or by using faster optical flow implementations. Furthermore, if results are needed for a batch of videos, computationally heavy preprocessing can be run on different machine(s) in parallel with interactive segmentation for different videos. Running spectral clustering on video intervals of up to 100 frames takes from 3 to 45 seconds on an Intel Xeon E5@3GHz equipped with 8GB of RAM (it can take less time if more constraints are available). To minimize the user's wait, the initial unconstrained spectral clustering is done in the preprocessing stage and the user corrects small batches of 2 or 3 frames well spaced out in time before asking the system to re-segment.

In Table 1 we compare our total user time, which includes the re-segmentation wait and the time for the final segment-object assignment, with that obtained with the LiveCut system [14]. For most of the sequences (except

[1] For 2 of the 10 evaluation videos in [14] it was not clear what the start and end frames were.

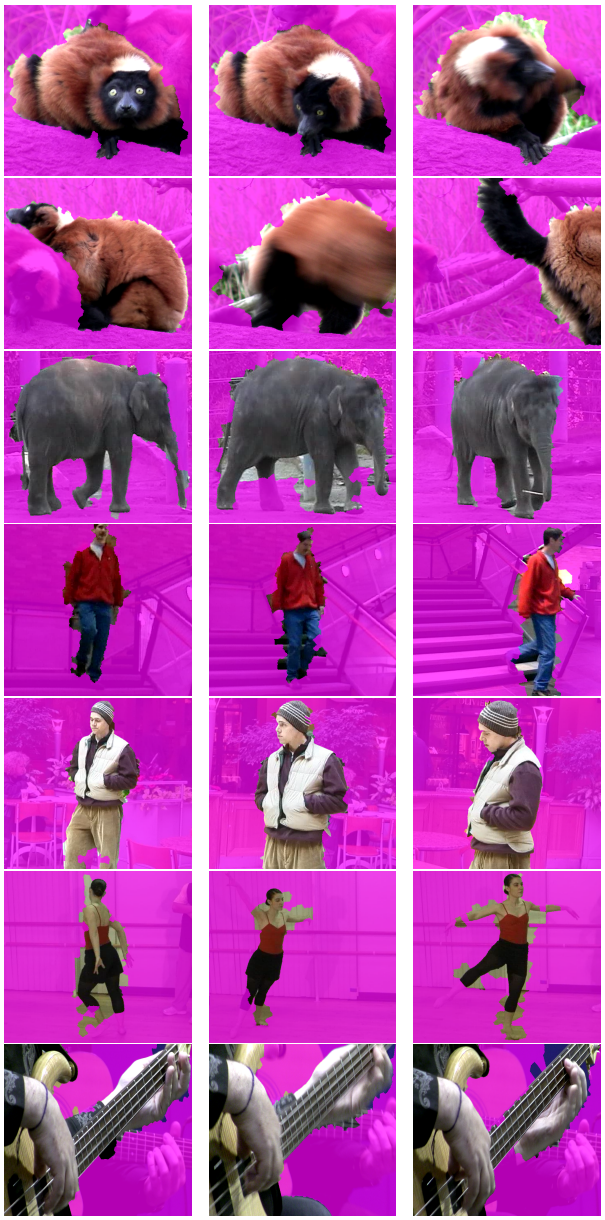


Figure 3: Interactive video segmentation results. The "lemurs" sequence contains two foreground objects (first and second row). While the masks are not perfect, they are obtained much faster than with state of the art systems [14]. Reasons for errors include heavy motion blur (second row, second column), very fast articulated motion (sixth row, second and third columns), and contours with inlets (third row, second and third columns).

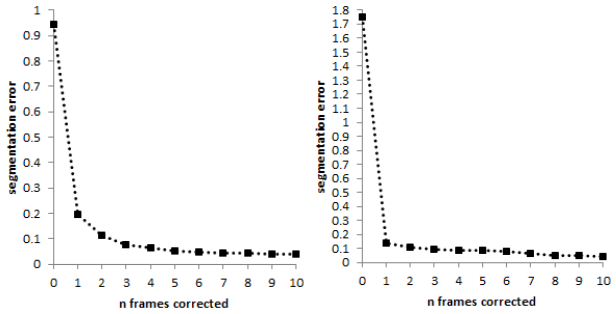
"cat", the shortest), our system is roughly 2-3 times faster than [14]. Figure 3 shows sample object masks for videos "lemurs", "elephant", "stairs", "manincap", "ballerina" and "bass guitar". The supplementary material includes visualizations of these masks for complete videos. Note that we could have continued refining the masks, but we stopped when they became reasonably good in all the frames. The segmentation results we obtained required labeling between 4 and 8 frames in each video interval of analysis.

To gauge the sensitivity to frame choice and the increase in segmentation quality as more annotations become available, we simulated system behavior for random configurations of corrected frames. These were chosen at least 5 video frames apart from each other and re-segmentation was done at each corrected frame using all the frames corrected so far. In the simulation, correcting a frame was implemented by completely replacing its current labeling with the ground truth labeling. We created ground truth object masks at the superpixel level for all the frames of two LiveCut sequences, "manincap" (1 object) and "lemurs" (2 objects). To evaluate a proposed video segmentation, we assigned its segments to the ground truth segments with the highest intersection over union and we computed the segmentation error as the ratio (number of superpixels covered by the wrong segment as per the assignment) to (total number of object superpixels). Note that the denominator does not include superpixels belonging to the background, which can be fairly numerous, but we do count errors made on background superpixels in the numerator. Also note that multiple segments may cover an object, but as long as a segment does not contain superpixels from different objects, it does not contribute to the segmentation error. We generated 10 random configurations per video interval and averaged the 10 errors obtained after each frame correction. The results for the first 100 frames of "manincap" and for the entire "lemurs" sequence are visualized in Figure 4. The segmentation error generally decreases with respect to the number of frames corrected, more so when this number is small.

With a similar setup, we examined the influence of d_{max}, the distance threshold for temporal label propagation, on the segmentation results. In Figure 5 we plot the segmentation error curves for values of d_{max} in the set $\{0.1, 0.2, 0.3, 0.4, 0.5\}$. Except for 0.1 and 0.5, a very conservative/permissive threshold for label propagation, the curves descend quite consistently, suggesting that in the range $[0.2, 0.4]$ the exact value of d_{max} has little impact on segmentation quality.

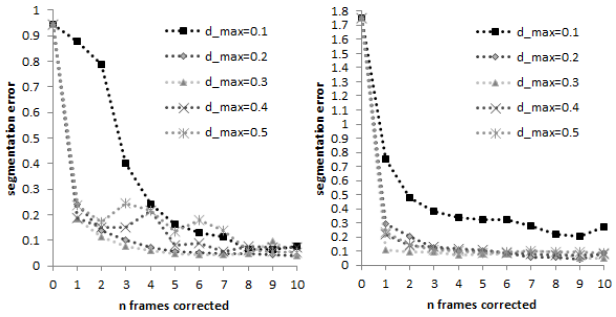
3.3. Role of Occlusion Boundaries

To validate our choice to include occlusion boundary cues in the interactive segmentation system, we ran spectral clustering without user supervision, modifying the superpixel graph only based on occlusion boundaries. We compared this unsupervised version of our method against Galasso et al.'s [7] on the Berkeley motion segmentation dataset [4]. This dataset consists of 26 sequences showing relatively simple traffic scenes (10 videos), short clips from the movie "Ms. Marple" (13 videos), 2 pedestrian



(a) "manincap" (b) "lemurs"

Figure 4: Simulation of interactive video segmentation for random configurations of corrected frames in two LiveCut sequences. The starting point of the segmentation error curves is unsupervised segmentation (0 corrected frames). To ensure the frame selection is neither extremely favorable nor extremely unfavorable, each curve displayed is the average of 10 curves for 10 random configurations.



(a) "manincap" (b) "lemurs"

Figure 5: Simulation of interactive video segmentation for different values of d_{max}, the threshold for temporal label propagation. The test setup is the same as in Figure 4.

scenes, and a video of a tennis player. Challenges include large camera motion, occlusions, objects sweeping across the screen, objects being still for long periods of time and highly non-rigid motion. We report results obtained with the evaluation software accompanying the dataset. Given the proposed segmentation of a video, the software matches its segments with the ground truth segments available in a few manually annotated frames and then computes five metrics reflecting the quality of the proposal: density, oversegmentation, overall error, average error and number of extracted objects. The density is the fraction of voxels covered by segments (the proposed segmentation is allowed to be incomplete). The oversegmentation is the average number of proposed segments assigned to the same ground truth segment (ideally 1). The overall error is the fraction of voxels covered by the wrong segment, given the assignment of proposed segments to ground truth segments. The average error

method	density	over-segm.	overall error	average error	# objects
Galasso et al. [7]	100%	6.77	9.92%	16.52%	17
Ours	99.31%	6.65	12.35%	23.38%	19
Ours w/o occ. bnd.	99.71%	6.54	22.36%	44.17%	9

Table 2: Results obtained with Galasso et al.'s method [7] versus ours. While [7] use complex combinations of superpixel distance measures, we employ simple distances but crucially integrate the output of an occlusion boundary detector in the superpixel graph construction. These simple distances perform poorly by themselves (last row), highlighting the importance of occlusion boundaries.

is also related to voxels covered by the wrong segment, but it is an average fraction over ground truth segments instead of over the entire video. The number of extracted objects is the number of proposed segments matching ground truth segments (except the background) with less than 10% error.

The evaluation software was run on the first 100 frames of each video, or the length of the video if it was shorter, to match the setting in [7]. Results are shown in Table 2. Our method's density is slightly less than 100% because we eliminate small groups of superpixels that have small affinity with the rest of the superpixels in the graph (in theory, they should not affect spectral clustering, but in practice this step always improves results). At virtually the same oversegmentation rate, our method has somewhat higher overall and average error, but also somewhat higher number of extracted objects, putting its performance close to [7]. The last row of the table shows an almost 2-fold increase in both overall and average error when the affinities of edges "cut" by occlusion boundaries are not modified. This demonstrates that (1) occlusion boundaries are a strong cue for video segmentation and (2) it is correct to design an interactive system in which user annotations complement the object contours found using occlusion boundaries.

Note that we do not aim to outperform [7] on unsupervised clustering, since in the interactive system the user still has to reduce error. Instead, we show that occlusion boundaries capture much of the performance of more complicated and likely less general approaches.

3.4. Limitations and Remarks on Applicability

Our system does not produce pixel perfect contours, as it operates on superpixels whose boundaries do not always align with object contours. The most significant errors in the unsupervised segmentation occur when the occlusion boundary detector misfires in large portions of the image, e.g. for motion blur or very fast and/or articulated motion.

More sophisticated occlusion detectors could be swapped in, but our interactive system still allows to correct errors efficiently even with the simple detector presented. Occlusion detectors will not fire in the absence of motion, but in this case user annotations typically propagate long term.

Our system does not do as well as LiveCut [14] if the video is short and the object has discriminative appearance – longer temporal context brings little benefit. However, visual clutter poses no additional problem to our system because most of the low level information comes from motion. Note that it is far from trivial to extend [14] to trade off segmentation quality for user time because their system assumes perfect segmentation in the previous frame. The user could correct the object mask less, but errors can accumulate quickly for moderately complex videos. Also, since [14] works frame by frame, it cannot produce a segmentation for the whole video given temporally incomplete annotations, which is possible in our system.

4. Conclusion

We built a system for interactive video segmentation on the basis of occlusion and long term spatio-temporal structure cues. With respect to other systems, ours provides a large reduction in the amount of user supervision for a small degradation of the segmentation results. Our system has the advantages that it does not rely on a discriminative object appearance model and it allows extracting multiple foreground objects together (in other systems, extracting each object is a separate task). Additional experiments with unsupervised clustering based on occlusion boundaries demonstrate the importance of this cue for video segmentation and validate the design of our system, which essentially complements occlusion boundaries with user annotation of the object contours.

References

[1] V. Badrinarayanan, F. Galasso, and R. Cipolla. Label propagation in video sequences. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[2] X. Bai, J. Wang, and G. Sapiro. Dynamic color flow: a motion-adaptive color model for object segmentation in video. In *European Conference on Computer Vision*, 2010.

[3] X. Bai, J. Wang, D. Simons, and G. Sapiro. Video snapcut: Robust video object cutout using localized classifiers. In *ACM Transactions on Graphics*, 2009.

[4] T. Brox and J. Malik. Object segmentation by long term analysis of point trajectories. In *European Conference on Computer Vision*, 2010.

[5] I. Endres and D. Hoiem. Category independent object proposals. In *European Conference on Computer Vision*, 2010.

[6] P. Felzenszwalb and D. Huttenlocher. Efficient graph-based image segmentation. In *International Journal of Computer Vision*, 2004.

[7] F. Galasso, R. Cipolla, and B. Schiele. Video segmentation with superpixels. In *Asian Conference on Computer Vision*, 2012.

[8] M. Grundmann, V. Kwatra, M. Han, and I. Essa. Efficient hierarchical graph based video segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[9] S. Kamvar, D. Klein, and C. Manning. Spectral learning. In *International Joint Conference On Artificial Intelligence*, 2003.

[10] Y. Lee, J. Kim, and K. Grauman. Key-segments for video object segmentation. In *IEEE International Conference on Computer Vision*, 2011.

[11] A. Levinshtein, C. Sminchisescu, and S. Dickinson. Spatiotemporal closure. In *Asian Conference on Computer Vision*, 2010.

[12] Z. Lu and M. Carreira-Perpinan. Constrained spectral clustering through affinity propagation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[13] A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Neural Information Processing Systems*, 2001.

[14] B. Price, B. Morse, and S. Cohen. Livecut: Learning-based interactive video segmentation by evaluation of multiple propagated cues. In *IEEE International Conference on Computer Vision*, 2009.

[15] S. Rangapuram and M. Hein. Constrained 1-spectral clustering. In *Journal of Machine Learning Research*, 2012.

[16] A. Sharma, E. von Lavante, and R. Horaud. Learning shape segmentation using constrained spectral clustering and probabilistic label transfer. In *European Conference on Computer Vision*, 2010.

[17] D. Sun, S. Roth, and M. Black. Secrets of optical flow estimation and their principles. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[18] N. Sundaram, T. Brox, and K. Keutzer. Dense point trajectories by gpu-accelerated large displacement optical flow. In *European Conference on Computer Vision*, 2010.

[19] A. Vazquez-Reina, S. Avidan, H. Pfister, and E. Miller. Multiple hypothesis video segmentation from superpixel flows. In *European Conference on Computer Vision*, 2010.

[20] S. Vijayanarasimhan and K. Grauman. Active frame selection for label propagation in videos. In *European Conference on Computer Vision*, 2012.

[21] J. Wang, P. Bhat, R. Colburn, M. Agrawala, and M. Cohen. Interactive video cutout. In *ACM Transactions on Graphics*, 2005.

[22] X. Wang and I. Davidson. Flexible constrained spectral clustering. In *International conference on knowledge discovery and data mining*, 2010.

[23] L. Xu, W. Li, and D. Schuurmans. Fast normalized cut with linear constraints. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.