Robust Video Segment Proposals with Painless Occlusion Handling

Zhengyang Wu(1), Fuxin Li(1), Rahul Sukthankar(2), James M. Rehg(1)
(1) Computational Perception Lab - Georgia Tech. (2) Google Research
[zwu66, fli, rehg]@cc.gatech.edu, sukthankar@google.com

Abstract

We propose a robust algorithm to generate video segment proposals. The proposals generated by our method can start from any frame in the video and are robust to complete occlusions. Our method does not assume specific motion models and even has a limited capability to generalize across videos. We build on our previous least squares tracking framework, where image segment proposals are generated and tracked using learned appearance models. The innovation in our new method lies in the use of two efficient moves, the merge move and free addition, to efficiently start segments from any frame and track them through complete occlusions, without much additional computation. Segment size interpolation is used for effectively detecting occlusions. We propose a new metric for evaluating video segment proposals on the challenging VSB-100 benchmark and present state-of-the-art results. Preliminary results are also shown for the potential use of our framework to track segments across different videos.

1. Introduction

Following the successes of image segmentation and segment-based object recognition, there has been increasing interest in video segmentation. Video offers an excellent opportunity for utilizing motion and appearance cues to potentially reduce the ambiguity in image segmentation. Progress in video segmentation could enable new approaches to building non-rigid 3D object models from video, understanding dynamic scenes, analyzing robot-object interactions, and other high-level vision tasks.

The increasing interest in video segmentation has also led to a diverse set of problem definitions and different assumptions based on these definitions. Video segmentation has been defined by different researchers as separating foreground from background [4, 28, 37, 45], identifying moving objects [12, 29, 34], creating segmentation proposals [3, 28, 31, 45], computing hierarchical sets of coarse-to-fine video segments [19, 24, 36, 42], or generating motion segmentations [24, 35]. Each of these definitions has its own merits and applications. However, inherent ambiguities exist, particularly in the definition of what is an object in video. One could identify a man wearing a suit and belt as a consistently moving object and segment him using motion segmentation. Alternatively, his belt and suit could each independently be an object of interest. There are also some objects that do not move but are still of interest in a video, e.g., a person leaning against a wall. A further complication is occlusion: a person can enter the scene, be occluded by others for a significant amount of time, and then re-appear in the middle of the scene without changing his or her identity (see Figure 1). In the face of this complexity, should video segmentation methods maintain persistent identities, and how can they do so?

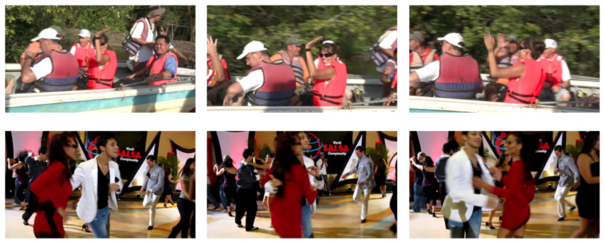
Figure 1: Complete occlusion happens frequently in video, but no existing solution addresses the problem efficiently.

In order to handle the diverse video segmentation needs of different applications, we argue that a stage for creating *video segment proposals* is necessary, where a pool of overlapping volumetric segments is generated, including segments that satisfy requirements for different applications. There have been some previous frameworks for creating video segmentation proposals, e.g., [3, 28, 31, 45]. However, they all have different restrictions, such as objects have to be moving independently [28], or segments have to start from the first frame [31]. Importantly, none of them handles the complete occlusion problem where a segment is fully occluded and then re-enters the scene. The restrictive assumptions of these previous methods make them unsuitable for a broad spectrum of applications.

In this paper, we introduce a new method which targets the limitations of existing approaches. We propose an approach that generates a pool of video segment proposals starting from any frame, identifies both moving and still objects as well as parts of an object, handles both partial and complete occlusions, and is free of any specific motion model (e.g., linear, smooth, etc.). It is our belief that such a proposal method can provide the necessary preprocessing for many subsequent algorithms.

Our approach trains long-term holistic appearance models on image segment proposals based on least squares, similar to [31]. Thousands of appearance models are efficiently trained on a pool of image segments, and tracked segments are gradually filtered in subsequent frames using appearance and motion constraints. There are two major differences: One is lifting the restrictive assumption as in [31] that all segments must start from the first frame. This is implemented via a series of forward and backtracking moves within the algorithm, without greatly increasing the time and space complexities. The second difference is that we handle complete occlusion, by an occlusion detection routine that automatically detects the onset of complete occlusions, maintains persistent identities of the occluded segments, and detects them when they re-enter the scene.

Both enhancements are implemented using efficient moves on the least squares regression. We store the sufficient statistics of the least squares model and propose two moves: a merge move that merges two models without significant time complexity, and a free addition move that tracks occluded segments and improves their appearance models without any additional operation. These moves are the main technical contributions of the paper.

The algorithm is tested on the challenging VSB-100 dataset which is currently the only large-scale video segmentation dataset that includes complete occlusions. However, the original evaluation metric is not suitable for evaluating proposals that can potentially overlap each other. Therefore we propose a new metric based on overlap and evaluate our approach along with others on the dataset and illustrate our state-of-the-art performance.

In addition, because our approach maintains persistent, long-term appearance models of video segment proposals, it can locate the same object when it appears in other videos. Preliminary video retrieval results are shown for such cross-video generalization. We believe this creates a path to potential novel applications such as video search.

2. Related Work

Video segmentation has seen quite some interest in recent years. However the problem definitions have been diverse. Many approaches focus on separating one foreground object from the background, using semi-supervised or unsupervised methods [4, 18, 28, 37, 38, 41, 45]. [2] is a multi-label semi-supervised approach, [28, 32, 37] are potentially multi-label, albeit they mainly aim to find a single moving object.

Among unsupervised multiple segmentation methods that generate an exhaustive list of video segments, agglomerative or spectral clustering on superpixels/supervoxels has been popular [10, 15, 19, 22, 24, 27, 36, 40, 43]. Some approaches utilize tracked feature points [5, 6, 14, 29, 34]. Approaches that track a pool of image segment proposals are most related to our approach [3, 31], however, these algorithms are unable to handle complete occlusions, and some have restrictions on where segment tracks may start.

Holistic image segment proposals [1, 8, 23, 26, 39] have been very popular in recent years as a preprocessing step for many successful object recognition systems on semantic segmentation and object detection problems (e.g., [7, 17, 20]). Holistic segment proposals can often cover entire objects in the image with excellent accuracy, hence shape and other global appearance features can be extracted from them, leading to better recognition accuracy. Currently, most state-of-the-art semantic segmentation systems utilize segment proposals in some stage.

Occlusion handling has been a major focus in the tracking literature for a long time [21, 44, 46] but research on occlusion handling for video segmentation is limited. There are works that handle occlusion in video segmentation under some assumptions, either only segmenting humans [13, 25] or utilizing multi-view cameras [33]. However, we do not know of any work that handles complete occlusions for generic video object segmentation.

3. The Least Squares Tracker and Moves

Our system is built on the flexible least squares tracker utilized in our previous work [31], which adopts the following regularized least squares formulation:

$$\min_{\mathbf{W}} \|\mathbf{X}\mathbf{W} - \mathbf{V}\|_F^2 + \lambda \|\mathbf{W}\|_{\mathrm{F}}^2, \tag{1}$$

where the goal is to recover the $d \times t$ weight matrix $\mathbf{W}$, given $n \times d$ input matrix $\mathbf{X}$ and $n \times t$ output matrix $\mathbf{V}$. n is the number of examples, d the dimensionality and t the number of distinct targets. $\|\mathbf{W}\|_{\mathrm{F}}$ is a Frobenius norm regularization used for convenience, but in principle any convex norm regularizer can be used. Designate $\mathbf{H} = \mathbf{X}^\top \mathbf{X} = \sum_{i=1}^n \mathbf{x}_i \mathbf{x}_i^\top$ and $\mathbf{C} = \mathbf{X}^\top \mathbf{V} = \sum_{i=1}^n \mathbf{x}_i^\top \mathbf{v_i}$, where $\mathbf{x}_i$ and $\mathbf{v}_i$ are columns of $\mathbf{X}$ and $\mathbf{V}$, respectively. The solution of the least squares is given by the linear system:

$$(\mathbf{H} + \lambda \mathbf{I})\mathbf{W} = \mathbf{C}. \tag{2}$$

The pair $(\mathbf{H}, \mathbf{C})$ is the sufficient statistics of the model. We denote $L = (\mathbf{H}, \mathbf{C})$ as a **least squares object** (LSO). Note each column in $\mathbf{C}$ corresponds to a distinct target (with the output a column in $\mathbf{V}$).

The least squares tracker is an online tracker since online **addition** updates can be made on the least squares object which adds examples and retains optimality. Suppose

$L_{t-1} = (\mathbf{H}_{t-1}, \mathbf{C}_{t-1})$ is the LSO from frame 1 to $t-1$, one can always perform:

$$\mathbf{H}_t = \mathbf{H}_{t-1} + \mathbf{X}_t^\top \mathbf{X}_t, \quad \mathbf{C_t} = \mathbf{C}_{t-1} + \mathbf{X}_t^\top \mathbf{V}_t \qquad (3)$$

to add examples described by $(\mathbf{X}_t, \mathbf{V}_t)$ to the regression problem. In addition, in order to **remove targets**, one only needs to remove the corresponding column in $\mathbf{C}_t$.

In [31], the least squares tracker has been used to track a pool of segment proposals. The main idea is to use the spatial overlap between segments as regression targets, and use appearance features in $\mathbf{X}$ to learn regression models $\mathbf{W}$ that serve as appearance models for each track: the overlap of every segment to the track can be predicted as regression outputs using $\mathbf{W}$. Such an overlap prediction regime has been used in many successful semantic segmentation systems starting from our work in [30]. The algorithm proceeds via the following steps:

1. A pool of segment proposals $\mathbf{A}_t$ is generated for each frame t.
2. Appearance features $\mathbf{X}_t$ are extracted for all segments and a spatial overlap matrix $\mathbf{V}_t$ is computed, containing overlaps between all pairs of segments within a single frame.
3. A least squares object L_1 is built from frame 1 by using $\mathbf{X}_1$ as input and $\mathbf{V}_1$ as output. The regression weight matrix $\mathbf{W}_1$ is obtained by solving the least squares problem (1), as the appearance model for all targets.
4. In frame 2, $\hat{\mathbf{V}}_2 = \mathbf{X}_2 \mathbf{W}_1$ is computed as the predicted overlaps for all targets.
5. Each segment in frame 2 is assigned to the target that has the highest predicted overlap to it ($\arg\max_{\mathbf{A}_2} \hat{\mathbf{V}}_2$) and satisfies simple motion constraints (not moving too far from the previous segment). New tracks are started on segments that are unassigned, with a new LSO L_2. Those targets that do not match any segment are no longer tracked subsequently.
6. Update L_1 using (3) with $\mathbf{X}_2$, $\mathbf{V}_2$ (with columns in $\mathbf{V}_2$ rearranged to reflect the matching). The output of each target is the overlap between every segment and the segment assigned to the target in frame 2.
7. Repeat steps 3–6 in subsequent frames till the end of the video. Stop starting new tracks from frame 6 because of memory/speed issues and only track previously generated tracks till the end of the sequence.

In this tracking framework, more than $1,000$ initial segments can be generated for each frame. The appearance models of all targets are built simultaneously using the efficient least squares framework. When the tracking continues, the greedy matching step removes many tracks that do not match well with any segment in terms of appearance. Therefore, after tracking for an extended period, the number of targets is reduced significantly but meaningful objects remain tracked. The approach is also robust to drifting, since segments from all previous frames are jointly utilized as training examples. As a consequence, the model is not overly biased to the appearance in the most recent frame.

However, our previous method [31] only identifies segments that start in the first few frames due to speed/memory issues, and the framework also does not handle complete occlusions: targets that are not matched to any segment are considered lost and never recovered. We believe that the framework has not been utilized to its full potential and propose to handle these two problems by innovative manipulations of the least square objects.

Mainly, two new moves on the least squares objects are proposed. The first is a **merge** move, which merges two adjacent LSOs differing by one frame. Suppose $L_1 = (\mathbf{H}_1, \mathbf{C}_1)$ contains all the training examples in $L_2 = (\mathbf{H}_2, \mathbf{C}_2)$ plus segments $\mathbf{A}_t$ and features $\mathbf{X}_t$. Then L_2 is tested on segments $\mathbf{A}_t$, in order to locate the matching segment to each target. After computing overlaps $\mathbf{V}_t$ between all segments $\mathbf{A}_t$ and the segments matching each target, $L_1^* = (\mathbf{H}_1, \begin{bmatrix}\mathbf{C}_1 & \mathbf{C}_2 + \mathbf{X}_t \mathbf{V}_t\end{bmatrix})$ becomes the merged result. Note this is only a $O(ndt_2)$ operation where t_2 is the number of targets in L_2. No time-consuming operation on $\mathbf{H}$ is required.

The other new move is called a **free addition**. Interestingly, when updating from L_{t-1} to L_t, if $\mathbf{V}_t$ is all zeros, then $\mathbf{C}_t = \mathbf{C}_{t-1}$ and only $\mathbf{H}_{t-1}$ needs to be updated. Suppose $\mathbf{C}_{t-1} = \begin{bmatrix}\mathbf{C}_{t-1}^v & \mathbf{C}_{t-1}^0\end{bmatrix}$ where $\mathbf{C}_{t-1}^0$ are targets that have all zero outputs in frame t, then $L_t = (\mathbf{H}_t, \begin{bmatrix}\mathbf{C}_t^v & \mathbf{C}_{t-1}^0\end{bmatrix})$. Now if L_{t-1} is updated to L_t in-place, then targets with zero outputs do not need to be updated. Since $\mathbf{H}_t$ is already updated for other targets, new training examples can be added to zero-output targets without any computation. Free additions can last for many frames without a single operation. This is important for occlusion handling, since completely occluded tracks have an overlap of 0 with any segment in the new frame, and can always be updated with free additions while they remain occluded.

The utilization of these efficient updates in the subsequent sections is the main technical contribution of this paper. Merging is important in backtracking and free additions help significantly for occlusion handling.

4. Tracking Segments from Every Frame

The original framework for least squares tracking requires the models for all targets to be trained on the same set of training examples, and could thus start tracks only in the first few frames. A straightforward approach to addressing this deficiency would be to maintain a least squares object starting from frame 1, and for targets originating in frame $k > 1$, simply designate all segments in frames 1 to $k-1$ as having an overlap of 0. However, this simple idea could fail because it incorrectly assumes that the start of each track corresponds to the first observation of the given target. Consequently, for a track starting in frame k, the system would incorrectly add all segments from prior frames as negative examples for the target, even though these could include earlier views of the target. Corrupting the training set in this manner will negatively impact the predictive power of the model.

In this paper, we propose a scheme combining forward tracking and backtracking to solve this problem. To prevent each LSO from having too many tracks, we run each LSO for at least k frames in order to prune out spurious tracks, and then backtrack to merge k LSOs into one, in order to reduce the number of LSOs. Namely, an LSO is initialized in every frame and tracked until $2k$ frames, then LSOs 1 to k will be merged to the first LSO (Fig. 2) while LSOs $k+1$ to $2k$ remain until frame $3k$, and are then merged to the LSO from frame $k+1$. In the end, the system will have an LSO every k frames. And after tracking for some length, some LSOs will end up having no targets and be removed from the system. Thus, the system still maintains good efficiency while being able to capture segments that start in any frame.

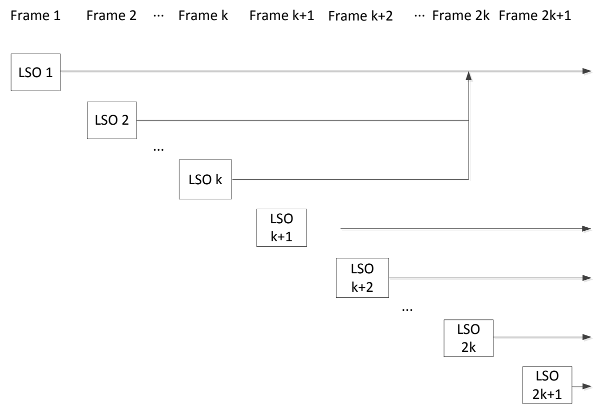

Figure 2: Merging LSOs. One LSO is initiated in each frame. At frame $2k$, the LSOs from frames 1 to k are all merged into the LSO from frame 1. This ensures that every LSO can last at least k frames before merging.

The full algorithm is shown in Algorithm 1. Such a scheme avoids the drawbacks of the original least squares tracking approach. Least squares tracking is often fragile in the beginning and tends to either pick spurious tracks or miss some tracks in the first few frames. However, after running for several frames, there are fewer tracks due to the greedy assignment step 5, and the tracker tracks more meaningful targets. At that time, backtracking those meaningful targets can both extend the length of the tracks and reduce the number of LSOs, which simultaneously improves quality while maintaining efficiency.

Algorithm 1 The tracking algorithm that combines forward tracking and backtracking.

```
input Segments $\mathbf{A}_1, \ldots, \mathbf{A}_T$ for frames 1 to $T$; appearance
  features $\mathbf{X}_1, \ldots, \mathbf{X}_T$ on all segments; overlap matrices
  $\mathbf{V}_1, \ldots, \mathbf{V}_T$.
  for $n = 0$ to $T$ step $k$ do
    for $i = n+1$ to $n+k$ do
      Test all the nonempty LSOs from $L_1$ to $L_{i-1}$ on $X_i$,
      $\hat{\mathbf{V}}_i = \begin{bmatrix}\mathbf{X}_i\mathbf{W}_1 & \ldots & \mathbf{X}_i\mathbf{W}_{i-1}\end{bmatrix}$.
      Find the matching segment of each target by
      $\arg\max_{A_i} \hat{\mathbf{V}}_i$.
      Remove the LSOs $L_t$ that do not have any remaining
      targets.
      Update $L_1, \ldots, L_{i-1}$ by $\mathbf{X}_i$ and the columns from
      $\mathbf{V}_i$ that corresponds to the matching segments.
      Initialize LSO $L_i = (\mathbf{X}_i^\top \mathbf{X}_i; \mathbf{X}_i^\top \mathbf{V}_i^u)$ from frame $i$,
      where $\mathbf{V}_i^u$ are columns from $\mathbf{V}_i$ that corresponding
      to segments not matched with any track.
    end for
    if $n > 0$ then
      for $j = n$ to $n-k$ step $-1$ do
        Merge $L_j$ to $L_{j-1}$, remove $L_j$.
      end for
    end if
  end for
```

5. Occlusion Handling

The least squares tracking framework does not have any dependency on temporal measurements such as optical flow. Therefore, the trained model for each track can naturally be tested at any frame, or even in another video. The main problem is that maintaining every disappeared track proposal from every frame would be expensive in terms of memory. Therefore, we propose an occlusion detection routine for identifying potential occlusions, and then utilize free additions for efficiently tracking occluded targets.

Our occlusion handling algorithm involves three steps:

1. Detect potential occlusions.
2. Track occluded and normal segments simultaneously. Occluded targets are tracked using free additions.
3. If some occluded targets re-emerge, connect the re-emerged segments to the corresponding track before the occlusion and continue to track those segments as normal tracks.

In step 2, occluded tracks are simply updated with free additions, which means that their corresponding columns in $\mathbf{V}$ are kept intact while the LSO is updated, until their track re-appears. All segments from the occluded frames are then automatically added as negative examples. There are two major aspects requiring discussion: the detection of occlusion and the detection of re-entry of occluded tracks.

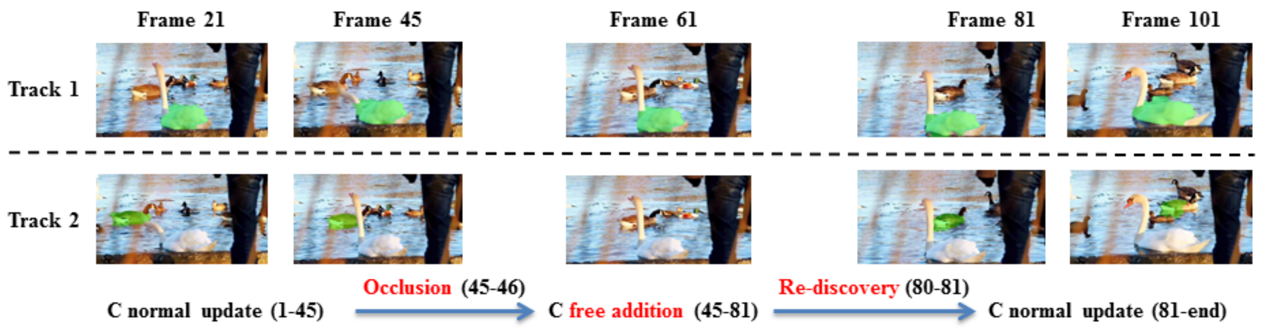

Figure 3: Illustration of our occlusion handling framework. (Top) Track 1 remains unoccluded throughout the video and is consistently tracked. (Bottom) Track 2 is occluded in frames 46–80. Our system tracks it consistently in frames 1–45, then predicts that it is occluded in frame 46 and marks it as occluded. For every frame after 46, we test whether it reappeared. In frame 81, a segment scores above the detection threshold, so our framework deems that the target has reappeared and continues to track it thereafter. The **C** matrix is updated normally in frames 1–45. In frames 46–81, during occlusion, it is updated efficiently with free additions.

5.1. Occlusion Detection

A basic idea in occlusion detection is to model the size of each segment over time, and to predict when the size will reach 0. We assume occlusion happens gradually over the course of several frames and thus the segment size should reduce smoothly. We also require the length of the track to be larger than $2k$ frames since, as discussed above, short tracks are more likely to be spurious. For each sufficiently long track, we construct a linear model of segment size over the last 10 frames to predict the segment's size in the next frame. If the ratio of a segment's predicted size to its average size (computed over the last 10 frames) falls below a threshold, then that segment is considered to be occluded. Note that this model correctly treats targets that leave the camera field-of-view as occlusions to account for the possibility of their reappearance at a later time. The threshold is set to a fairly strict one, since the original framework can already successfully handle partial occlusion.

5.2. Rediscovery of Occluded Tracks

With the free additions, the newly learned regression weight matrix $\mathbf{W}_t$ contains the learned model for both normal tracks and occluded tracks. For each segment proposal $S_{t+1,i}$ in the next frame, $\mathbf{x}_{t+1,i}\mathbf{W}_t$ gives the predicted overlap between the segment proposal and the tracks. For occluded tracks, the overlap is between the segment proposal and the track before the occlusion happens. Then if the overlap prediction is higher than a threshold, we assume that the occluded target has reappeared and mark the track as normal to resume the tracking process. Because our framework maintains the appearance of the occluded tracks before the occlusion happens, no matter how long the occlusion lasts, we can rediscover the occluded track correctly as long as the appearance of the object does not change significantly when it reappears.

6. Full Video and Cross-Video Testing

As discussed in Section 4, we only perform backtracking after k frames to merge LSOs. However, when a track has been tracked for a sufficiently long time, the evidence that it corresponds to a meaningful target is stronger. The appearance models for such tracks are reliable because they include many training examples and can thus be applied throughout the video to recover from tracking failures. Applying such a post-processing step means that the system is no longer online and requires sufficient memory to store all of the features in all the frames. However, when this is computationally feasible, the post-processing can further improve the performance of the proposed proposed algorithm, as seen in the experiments below.

For each frame in which a long track is missing, we test its corresponding appearance model against all of the segment proposals in that frame and select the segment with the highest prediction score as the match. If the score of the match is too low, we determine that the track is occluded in this frame and designate no match. In this way, long tracks are propagated throughout the entire video. Post-processing using a conditional random field or composite statistical inference [31] could be used to further improve performance.

We conduct a preliminary experiment to examine whether the learned appearance model is sufficiently good for recognizing the same target in other videos. Segments are extracted from a testing video and the appearance models of well-defined objects are tested on these segments. As in the post-processing, the best match with score over a threshold is designated the target object segment in the testing video. Experiment result in the next section shows that this simple approach performs well with objects that have similar appearances with the original video.

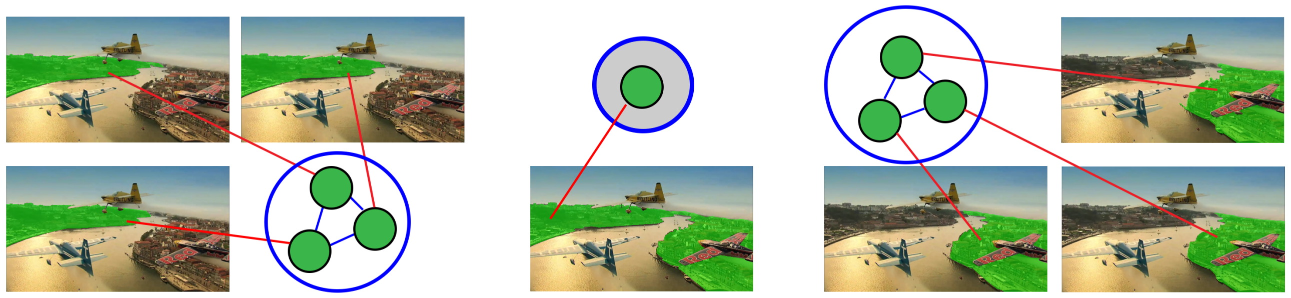

Figure 4: A basic example of our evaluation metric. Here, three of the four annotators treat the two river banks as two separate ground truth objects while one annotator believes that both banks should be treated as a whole object. The pairwise overlap between the three annotators are above the threshold, so they form two 3-cliques, producing two ground truth objects. Because nobody agrees with the fourth annotator, his annotation forms a 1-clique, which is discarded by our evaluation metric.

7. Experiments

We use the challenging VSB-100 dataset from [16] which contains 60 testing video sequences with pixel-level ground truth annotations every 20 frames. Each such frame contains annotations provided independently by four annotators. Compared to other datasets, this dataset has some videos that are extremely difficult due to severe occlusion and similar colors shared by many ground truth objects.

7.1. Evaluation Metric

We note that to make the original evaluation metrics on VSB-100 [16] well-defined, segmentations that are evaluated must have zero overlap with each other. This metric is hence unsuitable to evaluate segment proposals that can overlap each other. In order to explain this, we reproduce a simplified version of the formula for volume precision-recall from [16]. We assume only one human annotator in one frame to avoid further complications:

$$\begin{aligned} P &= \frac{\sum_{s\in\mathbb{S}} \max_{g\in\mathbb{G}} |s\cap g| - \max_{g\in\mathbb{G}} |g|}{|\mathbb{S}| - \max_{g\in\mathbb{G}} |g|} \\ R &= \frac{\sum_{g\in\mathbb{G}} \max_{s\in\mathbb{S}} |s\cap g| - 1}{|\mathbb{G}| - \Gamma_{\mathbb{G}}} \end{aligned} \quad (4)$$

where $|\mathbb{S}|$ represents the total area covered by all the segments in $|\mathbb{S}|$ and $\Gamma_{\mathbb{G}}$ is the number of segments in the ground truth. Assume that every pixel in the image is labeled as some segment in the ground truth, and that the maximal segment size in the ground truth is 20% of the full image size N. Now suppose we have a bag of single pixel segments, plus one segment that covers the whole frame; then, for each single-pixel segment, $|s\cap g| = 1$ because each single pixel is within some ground truth. This generates a precision $P = \frac{N+0.2N-0.2N}{2N-0.2N} = 55\%$ and recall $R = \frac{N-\Gamma_{\mathbb{G}}}{N-\Gamma_{\mathbb{G}}} = 100\%$. Thus, such a trivial segmentation would attain an unreasonably high F-score of 0.71! This counter-intuitive result occurs mainly because we supply conflicting segments to the metric, which is not allowed in the original paper. Since we cannot avoid conflicting segments in figure-ground proposals, we need to design a new metric.

Our metric is primarily based on average IoU overlaps, by treating each human-annotated segment as an object, but taking into consideration the agreement of multiple human annotators as well as temporal consistency. By temporal consistency, we mean that we do not want to evaluate on segments that only have ground truth in a single frame – such segments should be evaluated in image segmentation rather than video. Note that the ground truth is available in VSB-100 only once every 20 frames, thus we evaluate on volumes that are present in at least two ground truth frames. This ensures that most of the time the objects are present in the video for at least 0.6 seconds and not artifacts due to flickering and human error. These frames do not need to be adjacent, since we allow objects to disappear and re-emerge.

For the multiple annotator issue, the basic principle is to assume that an object is confirmed if there is agreement among at least two annotators. However, since human annotators rarely mark perfectly aligned segments, our protocol must correctly capture annotator agreement. We propose a novel definition of agreement as the maximal clique in a similarity graph $G = (S, E)$, built for each sequence. For each segment volume by each human annotator, a node $S_i \in S$ is created. We denote as S_{it} the ground truth annotation in frame t for S_i, and as M_i the annotator that creates S_i. Suppose there are N_s segment volumes from all annotators, we form an $N_s \times N_s$ overlap matrix V, where the overlap is computed across all annotated frames:

$$V_{ij} = \begin{cases} \frac{1}{T}\sum_{t=1}^{T} \mathrm{V}(S_{it}, S_{jt}), & \text{if } M_i \neq M_j \\ 0, & \text{if } M_i = M_j \end{cases} \quad (5)$$

where $\mathrm{V}(S,T) = \frac{|S\cap T|}{|S\cup T|}$ is the conventional IoU overlap. This overlap matrix measures the similarity between two segment volumes. If the overlap between two nodes is larger than a threshold (70% in our setting), then we create an edge between these two nodes. Finally, we find the max-

	A = 2	A = 3	A = 4	L = 2	L = 3	L = 4	L = 5	L = 6	L = 7
Backtrack & Occlusion	40.03	44.09	52.53	26.32	54.58	47.86	48.10	40.11	46.63
+Post-processing	**46.23**	**50.79**	**58.26**	26.32	**55.14**	50.93	**54.95**	**45.67**	**53.79**
Li et al. [31]	36.36	40.32	46.67	24.37	49.85	39.45	45.68	31.10	43.23
Grundmann et al. [19]	41.03	47.14	42.82	25.42	50.91	51.22	45.05	39.86	42.26
Galasso et al. [15]	38.5	40.86	47.48	**31.15**	52.72	**52.56**	42.40	37.97	42.21
Number of GT Objects	193	94	190	10	22	34	66	77	268

Table 1: Detailed score breakdowns. A = X represents the ground truth objects that X annotators agree with each other, L = X indicates ground truth objects that last X ground truth frames. The last row is the number of ground truth objects that belong to each category. Our algorithm is significantly better in the consistent objects, and in objects that appear in more frames.

imal clique of the graph and eliminate size-1 cliques. Now, our ground truth becomes the cliques. Let C_j^i represent the jth segment sequence in ith clique and C_{jt}^i represent frame t in this segment sequence. Given a segment proposal S_{kt}, our evaluation metric is

$$Score_i = \max_k \frac{1}{T}\sum_{t=1}^{T} \max_j \mathrm{V}(S_{kt}, C_{jt}^i). \quad (6)$$

Since we treat each clique as one object, we define the overlap between the clique and a segment proposal as the maximum overlap between the segment proposal and each ground truth segment inside the clique. Therefore, any human annotator would score 100% if their annotation belongs to the clique. For each clique (ground truth), we find the candidate segment with the largest overlap with this clique that is common in the evaluation of segment proposals. Our overall metric for one algorithm and one video sequence is thus:

$$S_o = \frac{1}{N}\sum_{i=1}^{N} Score_i, \text{where } N = \text{number of cliques}, \quad (7)$$

where the average is computed over the entire dataset. We also report the sequence score S_v, where we first average among all the ground truth objects within a sequence, then average this sequence score over the entire dataset.

7.2. Experiment Setup and Results

The framework in [31] used the segmentation proposal algorithm in [9]. In our experiments, we use our RIGOR segmentation proposal algorithm [23], because it produces proposals quickly while maintaining good quality. For features, we use Random Fourier transformation of bag-of-words on ColorSIFT, producing a 3000-dimensional feature vector for each segment proposal. The ratio threshold for detecting occlusion is set to 0.2. The threshold for re-discovering occluded track is set to 0.3. Regularization term λ for Eq. (1) is set to 100. All parameters are fixed across all testing videos. We conducted the experiments on a Intel Xeon W3550 (4 cores, 3.06GHz) machine. We compare our result to two state-of-art coarse-to-fine hierarchical video segmentation algorithms [19] and [15]. Both algorithms produce a hierarchy of segments ranging from a few pixels to the entire image. For fairness, we compute the maximum score over all the segments from all hierarchies as the score of the algorithm for each video. We report both the average score per video S_v and the average score per ground truth object S_o (introduced in Sec. 7.1) for each algorithm, and the average number of segments generated by each algorithm. We also compare our method against our previous work [31], but note that it suffers from not tracking segments that start later in the video. Finally, we include the score obtained by choosing the best RIGOR proposals as a theoretical upper bound for our algorithm.

	S_v	S_o	# Segs
Backtrack & Occlusion	50.12	45.81	324
+Post-processing	**56.13**	**51.92**	324
Original Li et al. [31]	44.81	41.25	46
Grundmann et al. [19]	45.28	42.94	737
Galasso et al. [15]	45.32	42.54	2850
RIGOR upp. bnd [23]	69.63	65.89	1420/frame

Table 2: VSB-100 results for different algorithms. S_v denotes the score averaged per video, S_o denotes the score averaged per object, see Sec. 7.1 for details of the metrics.

Table 2 summarizes the results. Our proposed algorithm with backtracking and occlusion handling improves significantly over previous approaches. Post-processing improves the results even further. In addition, the number of segments is much smaller than with hierarchical segmentation methods.

We also report the breakdown of the score in the amount of annotator agreement and the length of the ground truth track. Table 1 shows that targets on which more annotators agree also tend to be easier for algorithms to capture, because the object is more likely to have a distinct appearance. Also, for all algorithms, segments that appear in more than 3 frames (spanning at least 40 frames) are significantly

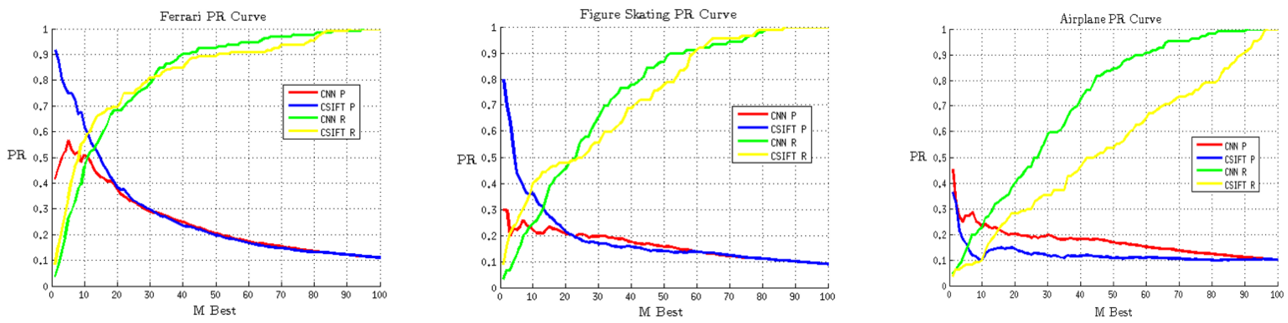

Figure 5: PR Curves of example categories of the cross-video object retrieval. The X-axis represents the number of retrieved videos, M. For objects that do not vary significantly across the video, our simple algorithm is able to predict labels with relatively high prediction and recall.

Table 3: Average per frame running time

Method	Time (s)
RIGOR Object Proposal	5.08
Feature Generation	27.52
Occlusion Handling	0.7
LSO Training	18.13
LSO Testing	0.6
Merge LSO (every k frames)	5.67

easier to track than ones that appear briefly. But our algorithm excels on long tracks that appear more than 5 frames (spanning at least 80 frames), showing the power of the proposed model in tracking through appearance changes and occlusion.

The time complexity required to process each LSO is $\mathcal{O}(nd^2 + d^3)$ (same as [31]). Multiplying by the number of LSOs, which is upper bounded by $T/k + k$, the overall complexity per frame is $\mathcal{O}(Tnd^2/k + Td^3/k)$. The complexity of the occlusion handling framework is dominated by other terms. In reality, most LSOs are empty after, e.g., 50 frames, and no longer tracked. We report our per frame running time averaged by all sequences below in Table 3. Note that faster features with fewer dimensions would speed the framework significantly.

7.3. Cross-video Segmentation

We present some preliminary results on cross-video object retrieval by testing a tracked video segment in other videos. We use the dataset from [41] which contains 10 types of objects such as airplane and bear. Each object is present in several different videos and the objects have different shapes and colors in each of these videos. There are 101 videos in total. For each video, a pool of candidate tracks are provided by the segmentation algorithm, and we pick the track that best matches the ground truth. Then the appearance model of this track is tested on all other videos of all object types. The testing score is computed as:

$$\text{Cross}_{i,j} = \frac{1}{N_j}\sum_{t=1}^{N_j} \max(\mathbf{X}_j^t \mathbf{W}_i), \tag{8}$$

where $\text{Cross}_{i,j}$ is the score of testing on video j with video i as training, N_j the number of frame in video j, $\mathbf{W}_i$ the appearance model (weight vector) of the best track in training video i, and $\mathbf{X}_j^t$ the feature matrix of segments in frame t of testing video j. For each training video, we then sort the scores descendingly and generate a list of M retrieved videos. With varying M, we report the average precision and recall for each object. We report our results with ColorSIFT feature as well as convolutional neural network features extracted from the two fully connected layers in [11].

Results are shown on 3 training categories: Ferrari, figure skating and airplane. Fig. 5 shows that the performance with ColorSIFT is relatively good on Ferrari, because the cars are consistent in appearance. The CNN features perform better on Airplane because videos in the airplane category include airplanes with different colors and shapes; CNN features generalize better across different shapes. However, neither of the two features dominate in all object categories and ColorSIFT dominates the precision when color is generalizable across videos. From our preliminary experiments, we believe that we need to design better features or a more sophisticated system to track a specific object in one video and generalize it across videos.

8. Conclusion

This paper proposes a video segment proposal approach with complete occlusion handling. This approach generalizes our previous least squares tracking framework by introducing efficient moves. Forward tracking and backtracking schemes are used to track segments starting from every frame and through complete occlusions. Results on the challenging VSB-100 dataset demonstrate the state-of-the-art performance of our approach. Preliminary results on cross-video generalization are also reported.

Acknowledgements

This research is partially supported by NSF IIS-1320348 and ARO MURI W911NF-11-1-0046.

References

[1] P. Arbeláez, J. Pont-Tuset, J. T. Barron, F. Marques, and J. Malik. Multiscale combinatorial grouping. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[2] V. Badrinarayanan, I. Budvytis, and R. Cipolla. Mixture of trees probabilistic graphical model for video segmentation. *IJCV*, 2013.

[3] D. Banica, A. Agape, A. Ion, and C. Sminchisescu. Video object segmentation by salient segment chain composition. In *International Conference on Computer Vision, IPGM Workshop*, 2013.

[4] M. V. D. Bergh, G. Roig, X. Boix, S. Manen, and L. V. Gool. Online video seeds for temporal window objectness. In *ICCV*, 2013.

[5] W. Brendel and S. Todorovic. Video object segmentation by tracking regions. In *ICCV*, pages 833 –840, 2009.

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

[7] J. Carreira, F. Li, and C. Sminchisescu. Object Recognition by Sequential Figure-Ground Ranking. *International Journal of Computer Vision*, 2012. First two authors contributed equally.

[8] J. Carreira and C. Sminchisescu. Constrained parametric min cuts for automatic object segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[9] J. Carreira and C. Sminchisescu. CPMC: Automatic Object Segmentation Using Constrained Parametric Min-Cuts. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 2012.

[10] J. Chang, D. Wei, and J. W. Fisher. A video representation using temporal superpixels. In *CVPR*, 2013.

[11] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets. *arXiv preprint arXiv:1405.3531*, 2014.

[12] R. Dragon, B. Rosenhahn, and J. Ostermann. Multi-scale clustering of frame-to-frame correspondences for motion segmentation. In *ECCV*, 2012.

[13] A. M. Elgammal and L. S. Davis. Probabilistic framework for segmenting people under occlusion. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 2, pages 145–152. IEEE, 2001.

[14] K. Fragkiadaki and J. Shi. Video segmentation by tracing discontinuities in a trajectory embedding. In *CVPR*, 2012.

[15] F. Galasso, M. Keuper, T. Brox, and B. Schiele. Spectral graph reduction for efficient image and streaming video segmentation. In *IEEE Conf. on Computer Vision and Pattern Recognition*, 2014.

[16] F. Galasso, N. S. Nagaraja, T. J. Cardenas, T. Brox, and B. Schiele. A unified video segmentation benchmark: Annotation, metrics and analysis. In *International Conference on Computer Vision*, 2013.

[17] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[18] M. Godec, P. M. Roth, and H. Bischof. Hough-based tracking of non-rigid objects. In *ICCV*, 2011.

[19] 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.

[20] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Simultaneous detection and segmentation. In *European Conference on Computer Vision*, pages 297–312, 2014.

[21] Y. Huang and I. Essa. Tracking multiple objects through occlusions. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, pages 1051–1058. IEEE, 2005.

[22] Y. Huang, Q. Liu, and D. Metaxas. Video object segmentation by hypergraph cut. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[23] A. Humayun, F. Li, and J. M. Rehg. RIGOR: Reusing Inference in Graph Cuts for generating Object Regions. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[24] A. Jain, S. Chatterjee, and R. Vidal. Coarse-to-fine semantic video segmentation using supervoxel trees. In *ICCV*, 2013.

[25] K. Kim and L. S. Davis. Multi-camera tracking and segmentation of occluded people on ground plane using search-guided particle filtering. In *Computer Vision–ECCV 2006*, pages 98–109. Springer, 2006.

[26] P. Krahenbuhl and V. Koltun. Geodesic object proposals. In *European Conference on Computer Vision*, 2014.

[27] J. Lee, S. Kwak, B. Han, and S. Choi. Online video segmentation by bayesian split-merge clustering. In *ECCV*, pages 856–869, 2012.

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

[29] J. Lezama, K. Alahari, J. Sivic, and I. Laptev. Track to the future: Spatio-temporal video segmentation with long-range motion cues. In *CVPR*, 2011.

[30] F. Li, J. Carreira, and C. Sminchisescu. Object recognition as ranking holistic figure-ground hypotheses. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2010. First two authors contributed equally.

[31] F. Li, T. Kim, A. Humayun, D. Tsai, and J. M. Rehg. Video segmentation by tracking many figure-ground segments. In *International Conference on Computer Vision*, 2013.

[32] T. Ma and L. J. Latecki. Maximum weight cliques with mutex constraints for video object segmentation. In *CVPR*, 2012.

[33] A. Mittal and L. S. Davis. M2tracker: A multi-view approach to segmenting and tracking people in a cluttered scene. *International Journal of Computer Vision*, 51(3):189–203, 2003.

[34] P. Ochs and T. Brox. Object segmentation in video: a hierarchical variational approach for turning point trajectories into dense regions. In *ICCV*, 2011.

[35] P. Ochs, J. Malik, and T. Brox. Segmentation of moving objects by long term video analysis. *PAMI*, 2014.

[36] G. Palou and P. Salembier. Hierarchical video representation with trajectory binary partition tree. In *CVPR*, 2013.

[37] A. Papazoglou and V. Ferrari. Fast object segmentation in unconstrained video. In *ICCV*, 2013.

[38] D. Tsai, M. Flagg, A. Nakazawa, and J. M. Rehg. Motion coherent tracking using multi-label mrf optimization. *International Journal of Computer Vision*, pages 1–13, 2012.

[39] J. R. Uijlings, K. E. van de Sande, T. Gevers, and A. W. Smeulders. Selective search for object recognition. *International Journal of Computer Vision*, 104(2):154–171, 2013.

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

[41] L. Wang, G. Hua, R. Sukthankar, J. Xue, and N. Zheng. Video object discovery and co-segmentation with extremely weak supervision. In *Computer Vision–ECCV 2014*, pages 640–655. Springer, 2014.

[42] C. Xu, C. Xiong, and J. Corso. Streaming hierarchical video segmentation. In *ECCV*, 2012.

[43] C. Xu, C. Xiong, and J. J. Corso. Streaming hierarchical video segmentation. In *European Conference on Computer Vision*, 2012.

[44] A. Yilmaz, X. Li, and M. Shah. Contour-based object tracking with occlusion handling in video acquired using mobile cameras. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(11):1531–1536, 2004.

[45] D. Zhang, O. Javed, and M. Shah. Video object segmentation through spatially accurate and temporally dense extraction of primary object regions. In *CVPR*, 2013.

[46] Y. Zhou and H. Tao. A background layer model for object tracking through occlusion. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 1079–1085. IEEE, 2003.