Spatio-temporal Object Detection Proposals

Dan Oneata, Jerome Revaud, Jakob Verbeek, and Cordelia Schmid

Inria*

Abstract. Spatio-temporal detection of actions and events in video is a challenging problem. Besides the difficulties related to recognition, a major challenge for detection in video is the size of the search space defined by spatio-temporal tubes formed by sequences of bounding boxes along the frames. Recently methods that generate unsupervised detection proposals have proven to be very effective for object detection in still images. These methods open the possibility to use strong but computationally expensive features since only a relatively small number of detection hypotheses need to be assessed. In this paper we make two contributions towards exploiting detection proposals for spatio-temporal detection problems. First, we extend a recent 2D object proposal method, to produce spatio-temporal proposals by a randomized supervoxel merging process. We introduce spatial, temporal, and spatio-temporal pairwise supervoxel features that are used to guide the merging process. Second, we propose a new efficient supervoxel method. We experimentally evaluate our detection proposals, in combination with our new supervoxel method as well as existing ones. This evaluation shows that our supervoxels lead to more accurate proposals when compared to using existing state-of-the-art supervoxel methods.

1 Introduction

Detection of human actions and activities is one of the most important and challenging problems in automatic video analysis. Recently there has been considerable progress in the recognition of human actions and events in challenging uncontrolled datasets such as HMDB [21], Hollywood2 [27], and TrecVid MED [30]. There are two main driving factors behind this progress. The first is the use of well engineered optical-flow based features computed along dense feature trajectories, see *e.g.* [19,42]. The second important factor is the use of state-of-the-art local feature pooling methods, such as the Fisher vector [28,36]. This progress, however, concerns the problem of classification of complete videos (as for TrecVid MED), or classification of clips that are well cropped around the actions of interest (as for HMDB and Hollywood2). The spatio-temporal detection problem is a more challenging one, since in each frame of the video we need to estimate a bounding box of the action of interest, which together form a spatio-temporal tube that locates the action in space and time.

Whereas sliding window search is still viable for temporal action localization, see *e.g.* [11,15,29], it becomes computationally prohibitive when searching in the much

* LEAR team, Inria Grenoble Rhône-Alpes, Laboratoire Jean Kuntzmann, CNRS, Univ. Grenoble Alpes, France.

larger space of spatio-temporal tubes. Efficient search methods for spatio-temporal action localization have been proposed in the past, exploiting additivity structures of bag-of-word representations and linear classifiers, *e.g.* using branch-and-bound search or dynamic programming [38,48]. These methods, however, do not apply when the representation is non-additive, as is the case for the Fisher vector representation used in recent state-of-the-art action classification methods, due to its non-linear power and ℓ_2 normalizations. Some recent work partially addresses this issue using efficient implementations [23,40] or approximate normalizations [29]. A more general technique to tackle this issue, which has recently surfaced in the 2D object recognition literature, is the use of generic class independent detection proposals [2,12,26,39]. These methods produce image-dependent but unsupervised and class-independent tentative object bounding boxes, which are then assessed by the detector. This enables the use of more computationally expensive features, that would be too expensive to use in sliding window approaches. Recent state-of-the-art object detectors based on this approach use various representations, *e.g.* Fisher vectors [8], max-pooling regionlets [42], and convolutional networks [17].

In this paper we explore how we can generate video tube proposals for spatio-temporal action detection. We build on the recent approach of Manen *et al.* [26] that uses a randomized superpixel merging procedure to obtain object proposals. Our first contribution is to extend their approach to the spatio-temporal domain, using supervoxels as the units that will be merged into video tube proposals. We introduce spatial, temporal and spatio-temporal pairwise supervoxel features that are used to learn a classifier that guides the random merging process. Our second contribution is a new hierarchical supervoxel method that starts with hierarchical clustering of per-frame extracted superpixels. We experimentally evaluate our detection proposals, in combination with our new supervoxel method as well as existing ones. This evaluation shows that our supervoxels lead to more accurate proposals when compared to using existing state-of-the-art supervoxel methods.

Below, we first review related work in more detail in Section 2. Then, in Section 3 we present our supervoxel method, and in Section 4 our tube proposal method. In Section 5 we present our experimental evaluation results, and we conclude in Section 6.

2 Related Work

In this section we discuss the most relevant related work on supervoxels, and efficient detection methods based on object proposals and other techniques.

2.1 Supervoxel Methods

Instead of a complete survey of supervoxel methods, we concentrate here on the approaches most related to our work. The recent evaluation by Xu and Corso [45] compares five different methods [9,13,14,18,32] to segment videos into supervoxels. They identify GBH [18] and SWA [9] as the most effective supervoxel methods according to several generic and application independent criteria. SWA is a hierarchical segmentation method that solves normalized cuts at each level. At the finest levels, it defines similarities from voxel intensity differences, while at higher levels it uses aggregate features which are computed over regions merged at earlier levels. GBH is a hierarchical extension of the graph-based method of Felzenszwalb and Huttenlocher [13]. A streaming version of GBH was introduced in [47], which performs similar to GBH, but at a fraction of the cost by using overlapping temporal windows of the video to optimize the segmentation. GBH, similar to SWA, also uses aggregate features (such as color histograms) to define similarities once an initial segmentation is performed based on intensity differences.

While SWA and GBH directly work on the 3D space-time voxel graph, the recent VideoSEEDS approach of Van den Bergh *et al.* [41] shows that supervoxels of similar quality can be obtained by propagating 2D superpixels computed over individual frames in a streaming manner. In our own work we take a similar approach, in which we take per-frame SLIC superpixels [1] as the starting point, and merge them spatially and temporally to form supervoxels. The advantage of starting from per-frame superpixels is that the graphs that are used to form larger supervoxels are much smaller than those based on individual voxels. The superpixels themselves are also efficient to obtain since they are extracted independently across frames.

Since objects can appear at different scales, a single segmentation of a video into supervoxels does typically not succeed in accurately capturing all objects and either leads to under or over segmentation. Xu *et al.* [46] recently proposed a supervoxel hierarchy flattening approach that selects a slice through such a hierarchy that can maximize a variety of unsupervised or supervised criteria. In this manner the segmentation scale can be locally adapted to the content. Our work is related, in the sense that we also aim to use (hierarchical) supervoxel segmentation to find regions that correspond to objects. Unlike Xu *et al.* [46], however, we do not restrict ourselves to finding a single segmentation of the video, and instead allow for overlap between different detection hypotheses.

2.2 Object Proposals for Detection in Video

Several spatio-temporal action detection approaches have been developed based on ideas originally developed for efficient object detection in still images. Yuan *et al.* [48] proposed an efficient branch-and-bound search method to locate actions in space-time cuboids based on efficient subwindow search [22]. Search over space-time cuboids is, however, not desirable since the accuracy of the spatial localization will be compromised as the object of interest undergoes large motion. Tran and Yuan [38] proposed an efficient method for spatio-temporal action detection, which is based on dynamic programming to search over the space of tubes that connect still-image bounding boxes that are scored prior to the spatio-temporal search. Building the trellis, however, is computationally expensive since it requires per frame a sliding-window based scoring of all considered bounding boxes across different scales and aspect ratios if the tube size is allowed to vary over time. Moreover, the efficiency of such approaches relies on the additive structure of the score for a video cuboid or tube. This prevents the use of state-of-the-art feature pooling techniques that involve non-additive elements, including max-pooling [42], power and ℓ_2 normalization for Fisher vector representations [28] or second-order pooling [6].

Recently, several authors have proposed efficient implementations [23,40] and approximate normalizations [29] to efficiently use Fisher vector representations with non-linear normaliztions. A technique that is more general and applies to arbitrary representations, is the use of generic class-independent proposals [2,12,26,39], which have recently surfaced in the context of 2D object localization. These methods rely on low-level segmentation cues to generate in the order of several hundreds to thousands of object proposals per image, which cover most of the objects. Once the detection problem is reduced to assessing a relatively modest number of object hypotheses, we can use stronger representations that would otherwise have been prohibitively costly if employed in a sliding window detector, see the recent state-of-the-art results in *e.g.* [8,17,43]. Here we just discuss two of the most effective object proposal methods. Uijlings *et al.* [39] generate proposals by performing a hierarchical clustering of superpixels. Each node in the segmentation hierarchy produces a proposal given by the bounding box of the merged superpixels. The approach of Manen *et al.* [26] similarly agglomerates superpixels, but does so in a randomized manner. In Section 4 we show how this technique can be adapted for space-time detection proposals based on supervoxel segmentation.

Van den Bergh *et al.* [41] proposed a video objectness method based on tracking windows that align well with supervoxel boundaries. The tracking is based on the evolution of the supervoxels inside the tracked window. Unlike [26,39] and our work, however, their method is inherently dependent on the scale of the supervoxels that produce the boundaries which define the objectness measure.

In parallel to our work, Jain *et al.* [20] developed an extension of the hierarchical clustering method of Uijlings *et al.* [39] to the video domain to obtain object proposals. The most notable difference with our work is that they compute their initial supervoxels from an "independent motion evidence" map. This map estimates for each pixel in each frame the likelihood that its motion is different from the dominant motion. While this approach is effective to segment out objects that are in motion w.r.t. the background, it does not provide a mechanism to recover objects that are static in the scene. Furthermore, estimating the dominant motion is often error prone in real world videos.

Finally, several recent methods address the related but different problem of motion segmentation in video [25,31,49]. Their goal is to produce a pixel-wise segmentation of the dominant moving object in videos. Unlike the methods discussed above, they produce a single estimate of the dominant object, which is assumed to be at least partially in motion. Both [25] and [49] are based on linking still-image object proposals from [12]. They refine the window-based solution using a pixel-wise MRF. Papazoglou and Ferrari [31] proposed a method using motion boundaries to estimate the outline of the object of interest, and refine these estimates using an object appearance model. Instead of relying on object proposal bounding boxes per frame, they rely on per-frame superpixel segmentation as the base units over which the energy function is defined. In our experiments we compare to the results of [31] on the YouTube Objects dataset.

3 Hierarchical Supervoxels by Spatio-temporal Merging

Our supervoxel approach starts from superpixels as basic building blocks, and aggregates spatially and temporally connected superpixels using hierarchical clustering.

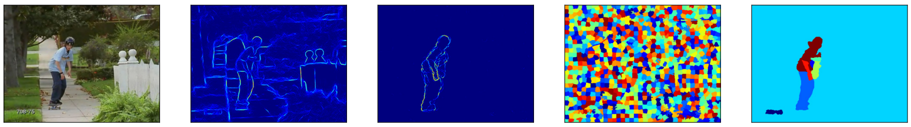

Fig. 1. Illustration of our supervoxel construction. From left to right: video frame, detected edges, flow boundaries, superpixels, and hierarchical clustering result at the level with eight supervoxels.

In Section 3.1 we detail the superpixel graph construction and the definition of the edge costs. Then, in Section 3.2 we present the hierarchical clustering approach which includes a novel penalty term that prevents merging physically disconnected objects.

3.1 Construction of the Superpixel Graph

We use SLIC [1] to independently segment each video frame into N superpixels. SLIC superpixels have been shown to accurately follow occlusion boundaries [24], and are efficient to extract. For each superpixel n, we compute its mean color $\mu(n)$ in Lab space, a color histogram $h_{\text{col}}(n)$ using ten bins per channel, and a flow histogram $h_{\text{flow}}(n)$ that uses nine orientation bins. We construct a graph $\mathcal{G} = (\mathcal{S}, \mathcal{E})$, where $\mathcal{S}$ is the set of superpixels, and $\mathcal{E}$ is the set of edges between them. In Section 5.1 we detail how we set the parameters of the graph weights using a small set of training images.

Spatial Neighbor Connections. Spatial connection edges are created per frame for each pair of neighboring superpixels n and m, and their weight $w^{\text{sp}}(n,m)$ is given by the weighted sum of distances based on several cues that we detail below:

$$\begin{aligned} w^{\text{sp}}(n,m) = {} & \alpha_\mu d_\mu(n,m) + \alpha_{\text{col}} d_{\text{col}}(n,m) + \alpha_{\text{flow}} d_{\text{flow}}(n,m) \\ & + \alpha_{\text{mb}} d_{\text{mb}}(n,m) + \alpha_{\text{edge}} d_{\text{edge}}(n,m), \end{aligned} \tag{1}$$

where $d_\mu(n,m) = \min(\|\mu(n) - \mu(m)\|, 30)$ is the robust thresholded-distance between the color means [33], $d_{\text{col}}(n,m)$ and $d_{\text{flow}}(n,m)$ are chi-squared distances between the color and flow histograms. In our implementation we use the LDOF optical flow method of Brox and Malik [5].

The last two terms, $d_{\text{mb}}(n,m)$ and $d_{\text{edge}}(n,m)$, are geodesic distances between the superpixels centroids, efficiently computed using the distance transform [44]. We use the norm of the gradient of the flow and the output of the recent structured edge detector [10] respectively, to define their geodesic pixel-wise cost. In practice, it means that if two superpixels are separated by an edge —either image-based or motion-based— the distance between them will increase proportionally to the edge strength. See Figure 1 for an illustration of these two distance terms.

Second-Order Spatial Connections. We also add second-order spatial edges to connect neighbors of neighbors. The rationale behind this setting is to be robust to small occlusions. Imagine, *e.g.*, the case of a lamp post in front of a tree: we would like the two parts of the tree to be connected together before they are merged with the lamp post. In this case we drop the geodesic distances, since they are affected by occlusions, and add a constant penalty α_{2hop} instead:

$$w^{\text{2hop}}(n,m) = \alpha_\mu d_\mu(n,m) + \alpha_{\text{col}} d_{\text{col}}(n,m) + \alpha_{\text{flow}} d_{\text{flow}}(n,m) + \alpha_{\text{2hop}}. \quad (2)$$

Whenever a second-order neighbor can be reached through an intermediate neighbor k with a smaller distance, *i.e.* when $w^{\text{sp}}(n,k) + w^{\text{sp}}(k,m) \leq w^{\text{2hop}}(n,m)$, we remove the second-order edge between n and m. This avoids spurious connections between physically disconnected regions, and also significantly reduces the number of edges in the graph.

Temporal Neighbor Connections. Temporal connections are naturally introduced using optical flow. We connect each superpixel with its neighbor in the next frame indicated by the flow. Because the flow is sometimes noisy, we enforce a one-to-one correspondence in the temporal connectivity. More precisely, for each superpixel at frame t, we compute its best match match in frame $t+1$ according to flow and pixel-wise color difference. We run this procedure in the opposite temporal direction as well, and keep only reciprocal connections. For two temporally connected superpixels n and m, we set the edge weight to:

$$w^t(n,m) = \alpha^t_\mu d_\mu(n,m) + \alpha^t_{\text{col}} d_{\text{col}}(n,m) + \alpha^t_{\text{flow}} d_{\text{flow}}(n,m). \quad (3)$$

This is similar to the spatial edge weight, but excludes the motion and contour boundaries as we do not have a temporal counterpart for them.

3.2 Hierarchical Clustering

Once the superpixel graph $\mathcal{G} = (\mathcal{S}, \mathcal{E})$ is constructed, we run a hierarchical clustering with average linkage [3]. For two clusters $A \subset \mathcal{S}$ and $B \subset \mathcal{S}$, we denote the set of edges connecting them as $\mathcal{B}(A,B) = \{(n,m) \mid n \in A, m \in B, (n,m) \in \mathcal{E}\}$. By construction, $\mathcal{B}(A,B)$ only contains edges at the boundary between A and B, or slightly deeper for the second-order connections. We define the distance between A and B as

$$w(A,B) = \frac{1}{|\mathcal{B}(A,B)|} \sum_{(n,m)\in\mathcal{B}(A,B)} w(n,m). \quad (4)$$

This corresponds to measuring the distance between two clusters as the average edge weight along their common boundary. Because the graph is sparse, the clustering can be computed efficiently: if the number of connections per superpixel is independent of the number of superpixels —as is the case in practice— the complexity is linear in the number of superpixels.

While such a clustering approach gives good results for image segmentation [3], its temporal extension tends to group clusters corresponding to different physical objects that are only accidentally connected in a small number of frames, see Figure 2.

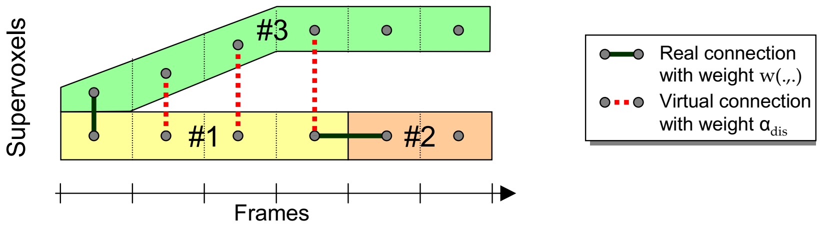
Supervoxels
#3
#1
#2
Frames
Real connection with weight w(.,.)
Virtual connection with weight α_dis

Fig. 2. Illustration of several supervoxels (SV) during the hierarchical clustering. While SV#1 and SV#2 can be merged without penalty, merging SV#1 and SV#3 will trigger a penalty α_{dis} in the form of a virtual edge added in all frames where both are present but not in contact.

We propose a simple solution to solve this issue. We add a penalty α_{dis} acting as a virtual edge for each frame where the two clusters are present but not in contact:

$$\tilde{w}(A,B) = \frac{c(A,B)w(A,B) + s(A,B)\alpha_{\text{dis}}}{c(A,B) + s(A,B)}, \tag{5}$$

where $c(A,B)$ is the number of frames where A and B are connected by direct or second-order spatial connections, and $s(A,B)$ is the number of frames where both are present but not connected. In practice, we set α_{dis} to the cost of the last merge of a hierarchical clustering performed preliminarily without temporal penalty, which corresponds to the weight of the hardest merge.

For an illustration of the supervoxel clustering process see the two right-most panels of Figure 1 which show the SLIC superpixels, and the hierarchical clustering result obtained from them.

The overall complexity of our method is linear in the number of frames. In practice, about 99% of the computational cost is devoted to computing LDOF optical flow [5]. Concretely, for a video of 55 frames with resolution 400×720 pixels, computing the flow with LDOF takes 13.8 minutes (about 15s/fr), computing the SLIC superpixels 9.7s, computing superpixels connection weights 7.8s, and performing hierarchical clustering 1.7s (all times are given for a single core @3.6GHz). Our method can benefit from the existing GPU implementation of LDOF [37], as well as a trivial parallelization over frames for the per-frame feature pipeline to compute SLIC, edges, and flow.

4 Spatio-temporal Object Detection Proposals

In this section we describe how to produce spatio-temporal object detection proposals based on a given supervoxel segmentation.

4.1 Randomized Supervoxel Agglomeration

We extend the region growing method of Manen *et al.* [26], which is a randomized form of Prim's maximum spanning tree algorithm. It starts from a random seed superpixel, and iteratively adds nodes that are connected to the ones that are already selected.

Instead of adding the node with the maximum edge, as in Prim's algorithm, edges are sampled with probability proportional to the edge weight. The edge weight w_{nm} that connects two superpixels n and m is given by a logistic discriminant classifier that linearly combines several pairwise features over superpixels, and predicts whether two superpixels belong to the same object or not.

At each merging step t a random stopping criterion is evaluated. The stopping probability is given as $(1 - w_{nm} + s(a_t))/2$, which is the average of two terms. The first, $(1 - w_{mn})$, is the probability (as given by the classifier) that the sampled edge connects superpixels that belong to different objects. This term avoids growing the proposal across object boundaries. The second term, $s(a_t)$ gives the fraction of objects in the training dataset that is smaller than the size a_t of the current proposal. This term ensures that the size distribution of the proposals roughly reflects the size distribution of objects on the training set. The sampling process can be repeated to produce a desired number of detection proposals.

To apply this method for spatio-temporal proposals, we consider a graph over the supervoxels, with connections between all supervoxels that are connected by a regular 3D 6-connected graph over the individual voxels. To ensure that we sample proposals that last for the full video duration, we continue the merging process as long as the full duration of the video is not covered, and use the stopping criterion only after that point. Once a merged set of supervoxels is obtained, we produce a space-time tube by taking in each frame the bounding box of the selected supervoxels in that frame.

4.2 Learning Supervoxel Similarities

We train a logistic discriminant model to map a collection of pairwise features to a confidence value that two supervoxels should be merged. To this end we generate a training set of supervoxel pairs from a collection of training videos. Each supervoxel is labeled as positive if it is contained for at least 60% inside a ground truth object, and negative otherwise. We then collect pairs of neighboring supervoxels that are either both positive, which leads to a positive pair, or for which one is positive and the other is negative, which leads to a negative pair. Neighboring supervoxel pairs that are both negative are not used for training.

We use eight different pairwise features between supervoxels, and list them below.

1. **Color feature.** We use the chi-squared distance between supervoxel color histograms $f_{\text{color}}(n, m) = d_{\chi^2}(h_{\text{col}}(n), h_{\text{col}}(m))$. We use the same color histogram as used for our supervoxels, *i.e.* using ten bins per channel in the Lab space.
2. **Flow feature.** We measure chi-squared distances between histograms of optical flow $f_{\text{flow}}(n, m) = d_{\chi^2}(h_{\text{flow}}(n), h_{\text{flow}}(m))$. Here we also use the same histograms as before, *i.e.* using LDOF [5] and nine orientation bins.
3. **Size feature.** The size feature favors merging small supervoxels first, and is defined as the sum of the volumes of the two supervoxels: $f_{\text{size}}(n, m) = a_n + a_m$. The volumes a_n and a_m are normalized by dividing over the volume of the full video.
4. **Fill feature.** The fill feature that favors merging supervoxels that form a compact region, and measures to which degree the two supervoxels fill their bounding box: $f_{\text{fill}}(n, m) = (a_n + a_m)/b_{nm}$, where b_{nm} is the volume of the 3D bounding box of the two supervoxels, again normalized by the video volume.

5. **Spatial size feature.** This feature only considers the spatial extent of the supervoxels, and not their duration: $f_{\text{size2D}}(n,m) = \frac{1}{|t_m \cup t_n|} \sum_{t \in (t_m \cup t_n)} a_n^t + a_m^t$, where t_n is a set of frames in which supervoxel n exists, while a_n^t denotes the area of the supervoxel in frame t, normalized by the frame area.
6. **Spatial fill feature.** Similarly, the fill feature can be made duration invariant by averaging over time: $f_{\text{fill2D}}(n,m) = \frac{1}{|t_m \cup t_n|} \sum_{t \in (t_m \cup t_n)} (a_n^t + a_m^t)/b_{nm}^t$, where b_{nm}^t gives the area of the bounding box in frame t.
7. **Temporal size feature.** In analogy to the spatial size feature, we also consider the joint duration $f_{\text{duration}}(n,m) = |t_m \cup t_n|/T$, where T is the duration of the video.
8. **Temporal overlap feature.** We measure to what extent the supervoxels last over the same period of time by intersection over union: $f_{\text{overlap}} = |t_m \cap t_n|/|t_m \cup t_n|$.

The color (1), size (3) and fill (4) features are similar to those used by [26,39] for still-image object detection proposals. Besides these features, we also include features based on optical flow, as well as size and fill features that consider separately the spatial and temporal extent of the supervoxels (2, 5, 6, 7, 8). We do not include the motion boundary and edge features of Section 3, since these are not defined for neighboring supervoxels that have no temporal overlap.

In our experiments we use the same set of eight features regardless of the underlying supervoxel segmentation, but we do train specific weights for each segmentation to account for their different characteristics.

5 Experimental Evaluation Results

Before presenting our experimental results, we first briefly describe the experimental setup, datasets, and evaluation protocols in Section 5.1. We then evaluate our supervoxel algorithm in Section 5.2, and our spatio-temporal detection proposals in Section 5.3.

5.1 Experimental Setup

Supervoxel Segmentation. We evaluate our supervoxel method on the Xiph.org benchmark of Chen *et al.* [7], using the following two of the evaluation measures proposed in [45]. The *3D segmentation accuracy* averages over all ground truth segments g the following accuracy: the volume of g covered by supervoxels that have more than 50% of their volume inside g, divided by the volume of g. The *3D undersegmentation error* averages the following error over all ground truth segments: the total volume of supervoxels that intersect g minus the volume of g, divided by the volume of g.

Spatio-temporal Detection Proposals. The first dataset we use is UCF Sports [35], which consists of 150 videos of 10 sports: diving, golf, kicking, lifting, horse riding, running, skating, swinging, high bar, and walking. We use five videos of each class for training our supervoxel similarities, and to select other hyperparameters such as the granularity of the base segmentation level. The remaining 100 videos are used for testing. Since the original ground-truth data is not very precise, we re-annotated the objects more accurately every 20 frames. These annotations, as well as the train-test division, are all publicly available on our project webpage.[1]

The second dataset we consider is the YouTube Objects dataset [34]. It contains a total of 1407 video shots divided over ten object categories. Each video contains one dominant object, for which a bounding box annotation is available in only one frame.

Similar to the 2D object proposal evaluation in [39], we measure performance using the best average overlap (BAO) of proposals with ground truth actions and objects. The BAO of a ground truth object v is given by the proposal p in the set of proposals P_v for v that maximizes the average overlap with the ground truth bounding boxes across all annotated frames. More formally:

$$\text{BAO}(v) = \max_{p \in P_v} \frac{1}{|T_v|} \sum_{t \in T_v} \text{Overlap}(p_t, b_v^t), \tag{6}$$

where T_v is the set of frames for object v with ground-truth annotation, and b_v^t denotes the bounding box of v in frame t, and p^t is the bounding box of the proposal in that frame. We measure the per-frame overlap in the usual intersection-over-union sense.

Based on the BAO we compute the mean BAO (mBAO) across all ground truth actions/objects. We also consider the correct localization (CorLoc) rate, as in [31], which measures the fraction of objects for which the BAO is above 50%.

5.2 Experimental Evaluation of Supervoxel Segmentation

Setting of Supervoxel Parameters. For our supervoxel method we set the number of SLIC superpixels to $N = 1,000$ per frame. We set the parameters of the hierarchical superpixel merging weights using a 9D grid search over their values, and evaluate the performance using a subset of 10 videos from the UCF Sports training set. For a given set of parameters, we generate the segmentation hierarchy, and for each node n in the hierarchy we evaluate the precision and recall w.r.t. the ground-truth object bounding box in all annotated frames. Precision $p(n)$ is defined as the fraction of the supervoxel that is inside the ground-truth box, and recall $r(n)$ the fraction of the ground-truth box that is inside the supervoxel. We then compute the maximum score of the product of recall and precision $F(n) = p(n)r(n)$ across all supervoxels in the hierarchy, and take the average of $\max_n F(n)$ across all annotated frames in all videos. In experiments on the Xiph.org benchmark we do not use boundary features, since the resolution of 240×160 of the videos in this data set is too small to obtain accurate boundary estimates.

Supervoxel Segmentation Evaluation Results. We compare our approach to GBH [18] and SWA [9], as well as the GB [13], Nyström [14], and meanshift [32] methods as evaluated in [45]. We used the publicly available implementation in LIBSVX.[2] For GBH we also used the online processing service which also uses optical flow features that are not included in the SVX implementation.[3] For our own method we also evaluate the effect of the penalty term α_{dis} that penalizes spatially distant supervoxel clusters.

[1] See `http://lear.inrialpes.fr/~oneata/3Dproposals`.

[2] See `http://www.cse.buffalo.edu/~jcorso/r/supervoxels`.

[3] See `http://www.cc.gatech.edu/cpl/projects/videosegmentation`.

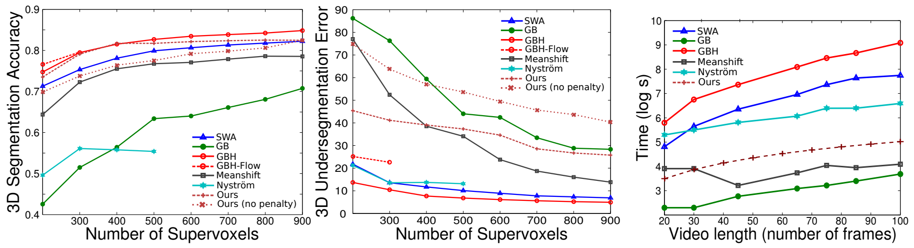

Fig. 3. Comparison of our and state-of-the-art supervoxel methods on the Xiph.org benchmark

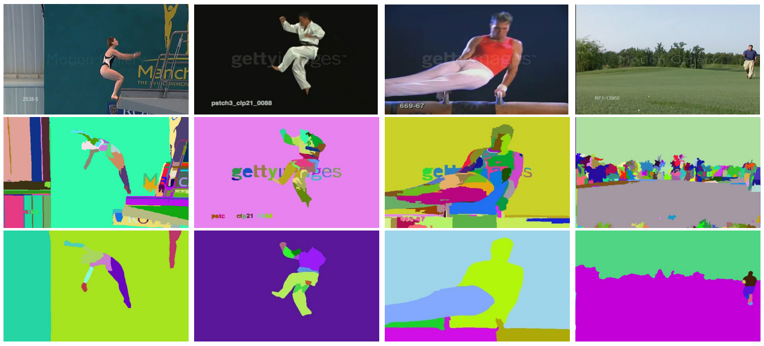

Fig. 4. Supervoxel comparison on videos of UCF Sports: video frames (top), GBH-Flow (middle), ours (bottom). For each video both methods are set to produce the same number of supervoxels.

The evaluation results are presented in Figure 3. In terms of segmentation accuracy (left) our method is comparable to the best methods: GBH and GBH-Flow. For the undersegmentation error (middle) our method gives significantly worse results than the best results obtained with GBH. The discrepancy of the evaluation results across these measures is due to the fact that our method tends to produce larger supervoxels, as well as many tiny supervoxels that consist of single isolated superpixels. Figure 4 illustrates this in comparison to GBH-Flow, where both methods are set to produce the same number of supervoxels for each video. Our method seems to produce less supervoxels due to isolated superpixels, which constitute more than 50% of the supervoxels. The undersegmentation error suffers from this, since a large supervoxel that overlaps a ground truth segment by a small fraction can deteriorate the error significantly.

Our method improves in both evaluation measures with the addition of the penalty term α_{dis} for spatially disconnected components, demonstrating its effectiveness. We therefore include it in all further experiments.

We compare the run times of the different supervoxel methods in the right panel of Figure 3. We do not include the GBH-Flow method here, since we ran it over the online service which does not allow us to evaluate its run time. As compared to GBH, the top performing method, our method runs one to two orders of magnitudes faster; and compared to the fastest method, GB, it is only about 4 times slower.

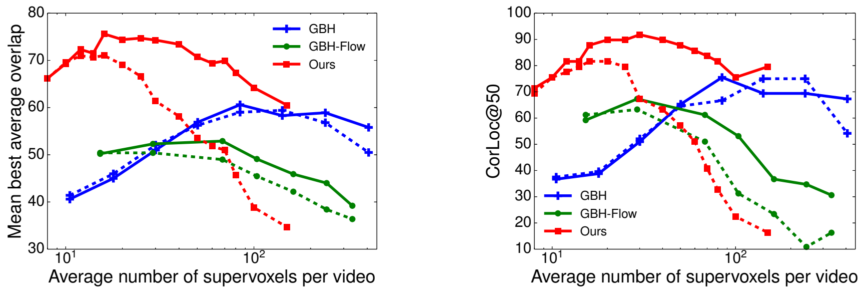

Fig. 5. Evaluation in terms of mBAO (left) and 50% CorLoc (right) of our, GBH, and GBH-Flow supervoxels on the UCF Sport train set, using 1,000 proposals as a function of the supervoxel granularity. Results with (solid) and without (dashed) full-duration constraint are shown.

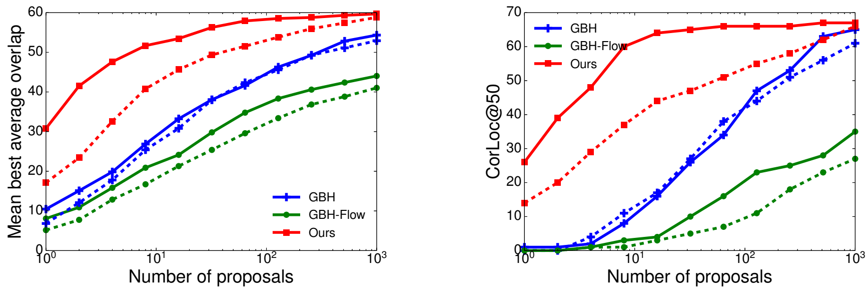

Fig. 6. UCF Sport testset performance against the number of proposals for our, GBH, and GBH-Flow supervoxels. Results with (solid) and without (dashed) full-duration constraint are shown.

5.3 Evaluation of Spatio-tempoal Dection Proposals

Video Tube Proposals for UCF Sports. In our first experiment we consider the performance using supervoxels from GBH, GBH-Flow, and our method with different granularities, i.e. different numbers of extracted supervoxels. In Figure 5 we compare the performance on the training set of the UCF Sports dataset. We consider the performance using 1,000 proposals for different granularities. The results show that our supervoxels lead to substantially better proposals. Moreover, using our supervoxels, a smaller number of supervoxels leads to optimal results. This means that the random proposal sampling is performed on a smaller graph which improves its efficiency. The full-duration temporal constraint, which accepts proposals only if they span the full duration of the video, improves results for all methods. It is particularly important for our supervoxels when using finer segmentations, probably because it helps to deal with very short supervoxels that are frequent in our approach. Based on these results we choose for each supervoxel method the optimal granularity, which will be used on the test set.

The results for the UCF Sports test set are given in Figure 6. For both evaluation measures, we need far fewer proposals for a given level of performance using our supervoxels, as compared to using GBH or GBH-Flow supervoxels. Also in this case the

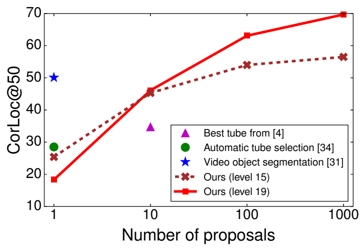

Fig. 7. Performance on the YouTube Objects dataset (50% CorLoc), as a function of the number of proposals. We show results for two levels of our supervoxel hierarchy (depicted as lines) and compare to other related methods (depicted as points). Level 15 (brown, dashed line) was the best performing one on the UCF Sports train set, while level 19 (red, solid line) was the best one tested on the YouTube Objects dataset.

full-duration temporal constraint benefits the performance, particularly so when generating few proposals using our supervoxels.

Video Tube Proposals for YouTube Objects. We now evaluate our approach on the YouTube Objects dataset, and compare our results to the video object segmentation method of [31], and weakly supervised learning method of Prest *et al*. [34]. The video object segmentation approach of [31] is unsupervised, and only outputs a single segmentation per shot. Prest *et al*. [34] used the method of Brox and Malik [4] to output coherent motion segments, to which they fit spatio-temporal tubes. This leads to between 3 and 15 tubes per shot. They then use a weakly supervised training to automatically select the tube corresponding to the object of interest, exploiting class labels given at the video level. We report their results, and the result for the best tube among the proposals from [4].

Figure 7 compares the aforementioned methods to ours when using between one and 1,000 proposals. When using 10 proposals, comparable to the number produced by [4], we obtain with 46.1% CorLoc, a result that is more than 10% points above the 34.8% of [4]. As compared to the video object segmentation method of [31], our method is about 5% worse when using 10 proposals, but eventually we recover more objects when more proposals are used.

5.4 Discussion

In our experiments we have found that the supervoxel evaluation measures of [45] are not directly indicative of the performance of these methods when used to generate spatio-temporal detection proposals (which is the goal of this paper). In its hierarchical clustering process our method quickly agglomerates large segments for objects, while also retaining a set of small fragments. In contrast, other methods such as GBH steadily produce larger segments from smaller ones, which leads to more homogeneously sized supervoxels. See Figure 4 for an example of our and GBH supervoxels at a given frame, and Figure 8 for the distribution of supervoxel durations. It seems that our more heterogeneous size distribution is advantageous for proposal generation, since good proposals can be made by merging a few large supervoxels. For spatio-temporal over-segmentation as measured by the supervoxel benchmark metrics, however, this unbalanced size distribution is sub-optimal since it is more likely to be imprecise at the object boundaries.

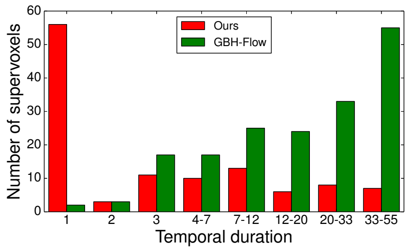

Fig. 8. Distribution of the supervoxel durations obtained on four video sequences from the UCF Sports dataset. Note the logarithmic binning of the duration, which is measured in frames. Both methods are set to produce the same number of supervoxels per video.

Recent supervoxel evaluation metrics proposed by Galasso *et al.* [16], which also assess temporal consistency and hierarchies, might be more related to the properties that are important for proposal generation.

6 Conclusion

In this paper we made two contributions. First, we have presented a new supervoxel method, that performs a hierarchical clustering of superpixels in a graph with spatial and temporal connections. Experimental results demonstrate that our supervoxel method is efficient, and leads to state-of-the-art 3D segmentation accuracy. Its 3D undersegmentation error is worse than that of competing state-of-the-art methods. This is probably due to the fact that our method yields supervoxels with a more heterogeneous size and duration distribution as compared to other methods. This seems to be detrimental for the undersegmentation error, but advantageous for proposal generation.

Second, we have adapted the randomized Prim 2D object proposal method to the spatio-temporal domain. To this end we have introduced a set of new pairwise supervoxel features, which are used to learn a similarity measure that favors grouping supervoxels that belong to the same physical object. Our experimental evaluation demonstrates that using our supervoxels leads to significantly better proposals than using existing state-of-the-art supervoxel methods. In future work we will integrate and evaluate the spatio-temporal proposals in a full detection system.

In our work we used optical flow as a cue to derive similarity in both the supervoxel stage and the proposal generation stage. Recent related approaches [20,31] used optical flow also to focus the proposal generation on areas where flow indicates the presence of objects moving against their background. While such an approach is clearly effective in cases where the object of interest undergoes significant motion as a whole, it is less clear whether it is still effective in cases where only part of the object is in motion. For example, consider a case where we want to detect a person that is drinking: if the person is seated, most of the body will be static and only the arm, hand, and cup in motion might be segmented out. In future work we plan to investigate this issue.

Acknowledgements. This work was supported by the European integrated project AXES and the ERC advanced grant ALLEGRO.

References

1. Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., Süsstrunk, S.: SLIC superpixels compared to state-of-the-art superpixel methods. PAMI 34(11), 2274–2282 (2012)
2. Alexe, B., Deselares, T., Ferrari, V.: Measuring the objectness of image windows. PAMI 34(11), 2189–2202 (2012)
3. Arbeláez, P., Maire, M., Fowlkes, C., Malik, J.: Contour detection and hierarchical image segmentation. PAMI 33(5), 898–916 (2011)
4. Brox, T., Malik, J.: Object segmentation by long term analysis of point trajectories. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part V. LNCS, vol. 6315, pp. 282–295. Springer, Heidelberg (2010)
5. Brox, T., Malik, J.: Large displacement optical flow: Descriptor matching in variational motion estimation. PAMI (2011)
6. Carreira, J., Caseiro, R., Batista, J., Sminchisescu, C.: Semantic segmentation with second-order pooling. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part VII. LNCS, vol. 7578, pp. 430–443. Springer, Heidelberg (2012)
7. Chen, A., Corso, J.: Propagating multi-class pixel labels throughout video frames. In: Proceedings of Western New York Image Processing Workshop (2010)
8. Cinbis, R., Verbeek, J., Schmid, C.: Segmentation driven object detection with Fisher vectors. In: ICCV (2013)
9. Corso, J., Sharon, E., Dube, S., El-Saden, S., Sinha, U., Yuille, A.: Efficient multilevel brain tumor segmentation with integrated Bayesian model classification. IEEE Trans. Med. Imaging 27(5), 629–640 (2008)
10. Dollár, P., Zitnick, C.: Structured forests for fast edge detection. In: ICCV (2013)
11. Duchenne, O., Laptev, I., Sivic, J., Bach, F., Ponce, J.: Automatic annotation of human actions in video. In: ICCV (2009)
12. Endres, I., Hoiem, D.: Category independent object proposals. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part V. LNCS, vol. 6315, pp. 575–588. Springer, Heidelberg (2010)
13. Felzenszwalb, P., Huttenlocher, D.: Efficient graph-based image segmentation. IJCV 59(2), 167–181 (2004)
14. Fowlkes, C., Belongie, S., Chung, F., Malik, J.: Spectral grouping using the nyström method. PAMI 26(2), 214–225 (2004)
15. Gaidon, A., Harchaoui, Z., Schmid, C.: Actom sequence models for efficient action detection. In: CVPR (2011)
16. Galasso, F., Nagaraja, N., Cardenas, T., Brox, T., Schiele, B.: A unified video segmentation benchmark: Annotation, metrics and analysis. In: ICCV (2013)
17. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: CVPR (2014)
18. Grundmann, M., Kwatra, V., Han, M., Essa, I.: Efficient hierarchical graph-based video segmentation. In: CVPR (2010)
19. Jain, M., Jégou, H., Bouthemy, P.: Better exploiting motion for better action recognition. In: CVPR (2013)
20. Jain, M., van Gemert, J., Bouthemy, P., Jégou, H., Snoek, C.: Action localization with tubelets from motion. In: CVPR (2014)
21. Kuehne, H., Jhuang, H., Garrote, E., Poggio, T., Serre, T.: HMDB: A large video database for human motion recognition. In: ICCV (2011)
22. Lampert, C., Blaschko, M., Hofmann, T.: Efficient subwindow search: a branch and bound framework for object localization. PAMI 31(12), 2129–2142 (2009)

23. Li, Z., Gavves, E., van de Sande, K., Snoek, C., Smeulders, A.: Codemaps, segment classify and search objects locally. In: ICCV (2013)
24. Lu, J., Yang, H., Min, D., Do, M.: Patch match filter: Efficient edge-aware filtering meets randomized search for fast correspondence field estimation. In: CVPR (2013)
25. Ma, T., Latecki, L.: Maximum weight cliques with mutex constraints for video object segmentation. In: CVPR (2012)
26. Manén, S., Guillaumin, M., Gool, L.V.: Prime object proposals with randomized Prim's algorithm. In: ICCV (2013)
27. Marszalek, M., Laptev, I., Schmid, C.: Actions in context. In: CVPR (2009)
28. Oneata, D., Verbeek, J., Schmid, C.: Action and event recognition with Fisher vectors on a compact feature set. In: ICCV (2013)
29. Oneata, D., Verbeek, J., Schmid, C.: Efficient action localization with approximately normalized Fisher vectors. In: CVPR (2014)
30. Over, P., Awad, G., Michel, M., Fiscus, J., Sanders, G., Shaw, B., Kraaij, W., Smeaton, A., Quénot, G.: TRECVID 2012 – an overview of the goals, tasks, data, evaluation mechanisms and metrics. In: Proceedings of TRECVID (2012)
31. Papazoglou, A., Ferrari, V.: Fast object segmentation in unconstrained video. In: ICCV (2013)
32. Paris, S., Durand, F.: A topological approach to hierarchical segmentation using mean shift. In: CVPR (2007)
33. Pele, O., Werman, M.: Fast and robust earth mover's distances. In: ICCV (2009)
34. Prest, A., Leistner, C., Civera, J., Schmid, C., Ferrari, V.: Learning object class detectors from weakly annotated video. In: CVPR (2012)
35. Rodriguez, M., Ahmed, J., Shah, M.: Action MACH: a spatio-temporal maximum average correlation height filter for action recognition. In: CVPR (2008)
36. Sánchez, J., Perronnin, F., Mensink, T., Verbeek, J.: Image classification with the Fisher vector: Theory and practice. IJCV 105(3), 222–245 (2013)
37. Sundaram, N., Brox, T., Keutzer, K.: Dense point trajectories by gpu-accelerated large displacement optical flow. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part I. LNCS, vol. 6311, pp. 438–451. Springer, Heidelberg (2010)
38. Tran, D., Yuan, J.: Optimal spatio-temporal path discovery for video event detection. In: CVPR (2011)
39. Uijlings, J., van de Sande, K., Gevers, T., Smeulders, A.: Selective search for object recognition. IJCV 104(2), 154–171 (2013)
40. van de Sande, K., Snoek, C., Smeulders, A.: Fisher and VLAD with FLAIR. In: CVPR (2014)
41. Van den Bergh, M., Roig, G., Boix, X., Manen, S., Gool, L.V.: Online video SEEDS for temporal window objectness. In: ICCV (2013)
42. Wang, H., Schmid, C.: Action recognition with improved trajectories. In: ICCV (2013)
43. Wang, X., Yang, M., Zhu, S., Lin, Y.: Regionlets for generic object detection. In: ICCV (2013)
44. Weber, O., Devir, Y., Bronstein, A., Bronstein, M., Kimmel, R.: Parallel algorithms for approximation of distance maps on parametric surfaces. ACM Trans. Graph. (2008)
45. Xu, C., Corso, J.: Evaluation of super-voxel methods for early video processing. In: CVPR (2012)
46. Xu, C., Whitt, S., Corso, J.: Flattening supervoxel hierarchies by the uniform entropy slice. In: ICCV (2013)
47. Xu, C., Xiong, C., Corso, J.J.: Streaming hierarchical video segmentation. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part VI. LNCS, vol. 7577, pp. 626–639. Springer, Heidelberg (2012)
48. Yuan, J., Liu, Z., Wu, Y.: Discriminative subvolume search for efficient action detection. In: CVPR (2009)
49. Zhang, D., Javed, O., Shah, M.: Video object segmentation through spatially accurate and temporally dense extraction of primary object regions. In: CVPR (2013)