This CVPR2015 paper is the Open Access version, provided by the Computer Vision Foundation.
The authoritative version of this paper is available in IEEE Xplore.

Learning Similarity Metrics for Dynamic Scene Segmentation

Damien Teney[1] Matthew Brown[2] Dimitry Kit[2] Peter Hall[2]
[1]Carnegie Mellon University [2]University of Bath
dteney@andrew.cmu.edu {m.brown,d.m.kit,maspmh}@bath.ac.uk

Abstract

*This paper addresses the segmentation of videos with arbitrary motion, including dynamic textures, using novel motion features and a supervised learning approach. Dynamic textures are commonplace in natural scenes, and exhibit complex patterns of appearance and motion (*e.g. *water, smoke, swaying foliage). These are difficult for existing segmentation algorithms, often violate the brightness constancy assumption needed for optical flow, and have complex segment characteristics beyond uniform appearance or motion. Our solution uses custom spatiotemporal filters that capture texture and motion cues, along with a novel metric-learning framework that optimizes this representation for specific objects and scenes. This is used within a hierarchical, graph-based segmentation setting, yielding state-of-the-art results for dynamic texture segmentation. We also demonstrate the applicability of our approach to general object and motion segmentation, showing significant improvements over unsupervised segmentation and results comparable to the best task specific approaches.*

1. Introduction

We study the segmentation of videos of arbitrary dynamic scenes, focussing on dynamic textures such as water, fire, or swaying trees [5]. These phenomena are commonplace in videos of natural scenes, but are poorly represented in general-purpose segmentation benchmarks [16, 40, 27], which mainly involve rigid or smooth non-rigid motion. Dynamic textures exhibit complex appearance and motion patterns, that usually have semantics beyond a simple consistency metric. A sequence depicting trees, for example, may contain smaller and larger branches, some static and others swaying in the wind, that would generally be assigned separate segments by unsupervised segmentation methods, while they ideally should all be part of one "tree" dynamic texture. Such mid-level interpretations of a scene are beyond the uniform priors of most "supervoxels" and unsupervised video segmentation methods [43]. To overcome these limitations, we present more appropriate features, together with a supervised algorithm to learn such information from annotated, ground truth segmentations.

This Work is supported by the EPSRC Grant EP/K02339X/1.

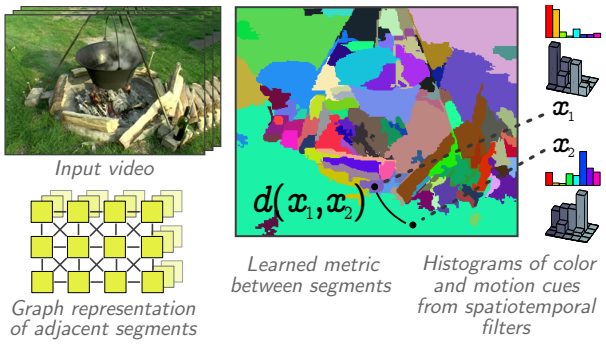


Figure 1. We extend the hierarchical graph-based segmentation technique and apply it to dynamic scene segmentation by learning distance metrics over motion and appearance features. Segments are iteratively merged, spatially and temporally, to form larger and larger segments. Merges occur between the most "similar" segments, as determined by our learned metric.

Our contribution consists of two parts. **First**, we use the responses to a bank of spatiotemporal filters to capture the appearance and motion characteristics of dynamic textures. Similarly to 2D filters that can capture structure in static images (*e.g.* edges), these 3D filters capture structure in the video volume, such as moving patterns [11]. The filters in the bank are tuned to various scales, orientations, and speeds, and provide a rich set of features that characterize image appearance and dynamics. Different dynamic textures often present different motion statistics. Water ripples in a pond, for example, will exhibit more low-frequency motion than ocean waves. These examples can be treated as a single "water" class, or as seperate phenomena depending on the labels in the training set, which justifies a learning approach. Therefore and **secondly**, we show how to learn a metric between descriptors made of histograms of color and filter-based motion cues, that allows supervised video segmentaton in a hierarchical, graph-based framework [18] (Fig. 1). We use ground truth segmentations to generate pairwise constraints between descriptors, and learn a metric that predicts whether segments should be merged or not

during the segmentation. If training segments are provided with semantic labels, we use additional pairwise constraints between segments of ("same-" or "different-class") to learn a metric that explicitly separates multiple semantic classes. This allows a variety of training scenarios. The general task of video segmentation encompasses a number of specific applications that can benefit from learned mid-level models and we additionally evaluate the applicability of our method to the more classical tasks of motion segmentation [28] and object segmentation from motion boundaries [35].

Practically, our work extends the hierarchical graph-based video segmentation technique of Grundmann *et al.* [18], which constructs a hierarchy of supervoxels of decreasing granularity. This algorithm is intrinsically suitable for a variety of tasks where the size of the desired segments is not fixed or known a priori. Although it already proved very successful in a number of benchmarks [43, 16], the algorithm is still limited to the grouping of pixels with a uniform prior on appearance and/or motion. Another practical, but significant drawback is its large memory requirements (handled *e.g.* with block-processing [45]), especially with the rich motion features that we propose. We address both of these issues by learning a metric between segment descriptors in a supervised setting, and *jointly* optimizing dimensionality reduction of the descriptors to dramatically reduce the memory requirements of the original algorithm. Note that this is not equivalent to first projecting the data with *e.g.* PCA and then learning a metric on low-dimensional descriptors, which may lose discriminative information. We rather optimize both tasks with a single objective function, thus fully exploiting the available supervision [34].

In summary, the technical contributions of this paper are threefold. (1) A new method to improve hierarchical video segmentation with supervised learning. We optimize a metric between segment descriptors over labelled training data, using a large-margin formulation suitable for hierarchical segmentation, unlike existing algorithms designed for nearest-neighbour classification. (2) We characterize the appearance and the motion within segments with a novel set of features based on spatiotemporal filters, that allows segmenting videos with arbitrary motions and complex dynamic textures. (3) We provide a method to optimize dimensionality reduction together with the learned metric, which drastically reduces the memory requirements of the original segmentation algorithm. These contributions are evaluated on a number of tasks and datasets, which demonstrate their value with results comparable or superior to the state of the art.

2. Related work

Supervised video segmentation Video segmentation has been studied extensively, in particular in the context of over-segmentation into supervoxels (see [43, 16] for benchmarks and reviews). These supervoxels are typically used as an efficient representation of videos for further higher-level processing. Supervised learning is a way to improve the performance of segmentation for specific tasks. Jiang *et al.* [26] showed recently that the inclusion of supervised learning in spectral clustering could lead to significant improvements for video segmentation. Closely related to our approach is the earlier work of Alpert *et al.* [1], who learned predictors on pairs of segments to guide hierarchical merging in the context of image segmentation. Learning pairwise predictors for agglomerative clustering, in the form of a similarity measure, was also studied recently by Jain *et al.* [24], using reinforcement learning, and applied to the segmentation of 3D medical images.

A number of closely-related mid-level tasks can be formulated as segmentation problems. Xu *et al.* [44] for example focused on tracking cars and humans in a segmentation framework, using learned models to improve the segmentation of instances of these specific classes. Object [35, 23] and motion [2, 37, 25] segmentation are other tasks that have been addressed both with specific methods [2, 37, 25], and within a general segmentation framework [39]. We have evaluated our method on mid-level tasks such as these, showing that the inclusion of supervised learning significantly improves results. Unlike previous works, however, our approach is able to handle a wider variety of cases, *e.g.* mixed scenes containing both dynamic textures and rigid motions.

Dynamic texture segmentation Extracting relevant motion features to characterize dynamic textures is challenging as they often violate the common optic flow assumptions of brightness constancy and rigid motion (for example in the case of water or smoke). Generative models have been proposed to directly model image intensities with linear dynamical systems [12, 7], then used for segmentation by iterative fitting [4, 5]. Most recent works proposed *ad hoc*, handcrafted descriptors [6, 20]. Spatiotemporal filters [15] were proposed early as a way to extract optical flow [22], though the responses to a bank of such oriented filters actually provide much richer information than optical flow. They allow the capture of multiple oriented structures at any space-time location, and handle both motion (*e.g.* translating objects) and non-motion (*e.g.* flickering effects) alike with the same formulation. Derpanis *et al.* looked extensively into their use for recognition of dynamic textures and scenes [11, 14], and Teney and Brown [39] recently used them as features for video segmentation. This paper leverages the richness of these features via supervised learning, in particular when using semantic ("class") annotations of training examples.

Metric learning We integrate supervised learning into the segmentation framework by learning a dissimilarity, or dis-

tance metric between pairs of segments. Metric learning has been studied extensively [41] mostly with the goal of improving nearest-neighbour classification. For example, Weinberger *et al.* [42] optimize a generalized Mahalanobis metric, which is equivalent to a linear transformation of the input features, to bring closer the close neighbours of the same class, and further apart those of different classes. Rather than optimising for nearest neighbour performance, Simonyan *et al.* [34] perform discriminative dimensionality reduction by optimizing a non-square linear projection matrix. We tried both of these approaches, finding the large margin formulation of Simonyan *et al.* [34] to perform better in our case (Section 5), ultimately improving the ability of our segment descriptors to predict whether they should be merged during segmentation.

3. Hierarchical video segmentation

We will now provide a brief review of the graph-based hierarchical video segmentation algorithm [18] then show how to integrate supervised learning to guide the process.

An input video is represented by a graph of connected regions, initialized as the lattice of adjacent voxels of the video (Fig. 1). Formally, the graph is represented by its nodes $\mathcal{N}$, and edges $\mathcal{E}$. Each node is assigned a descriptor of appearance x_i as described below. The edge topology initially represents the 26-connectivity of adjacent voxels in the video volume. An edge between nodes i and j is assigned a weight that represent a distance, or dissimilarity between their descriptors, $e_{ij} = d(x_i, x_j)$. This distance function can be unsupervised as in [18], or, as proposed, *learned*. The segmentation algorithm proceeds iteratively and, at each iteration, merges the pair of nodes i, j of lowest edge weight e_{ij}. The merged node, k, is then assigned a new descriptor x_k, the graph structure $\mathcal{E}$ is updated to represent spatial and temporal adjacency in the video volume, and the edges weights $e_{kl}\ \forall\, l$ are updated accordingly. The nodes of the graph thus represent growing segments. They are strictly decreasing in number, and they constitute segmentations of decreasing granularity. Most implementations of the algorithm return segmentations at a few discrete "levels" of segmentation, e.g. when a fixed percentage of edges have been removed, though the algorithm effectively produces a continuous tree of regions from pairwise merges[1].

The appearance descriptors of the segments consist of histograms of color [18] and histograms of filter-based motion cues (Section 4). There are two major advantages of using histograms as descriptors. First, they can be efficiently updated, the histogram of a merged node being the weighted average (by relative size) of those of the two original nodes. Second, and more importantly, histograms can effectively

[1]We consider an implementation of the algorithm where edge weights are continuously updated after each merge [17] instead of periodic updates at fixed levels [18].

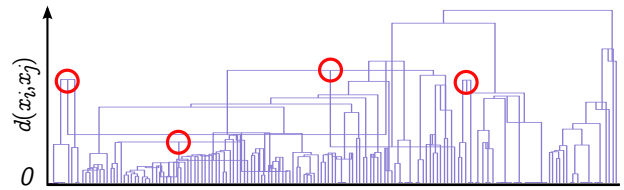


Figure 2. Dendrogram of a typical segmentation (car sequence of Fig. 7); the lowest, pixel-scale levels are not shown for clarity. The vertical axis shows the distance between merged segments, which is *not* monotonically increasing, as indicated by inversions in the dendrogram (circled). This stems from the segment descriptors being recomputed at the varying scales, and motivates the use of different metrics for different segment sizes.

represent appearance within segments at varying scales. A key factor in the effectiveness of the segmentation algorithm was shown to be in recomputing the similarity between segments as the algorithm proceeds, thus incorporating fine and coarse information [8]. However, a consequence of this is that weights of merged edges are not monotonically increasing, since these weights are recomputed using updated descriptors. Intuitively, two small segments with very different contents at a small scale, thus linked by an edge of large weight, will have, once merged, a more "uniform" histogram descriptor, and thus smaller edge weights with other segments. This can be visualized as inversions on a dendrogram of the merging process (Fig. 2), which are likely in hierarchical clustering with recomputed distances (as *e.g.* in centroid clustering [29]). Practically, this observation indicates that the distance function between segments may benefit from being adapted to the actual size of the segments being compared (Section 5.2).

4. Motion cues from spatiotemporal filters

We characterize texture and motion in the video using a bank of 3D, spatiotemporal filters [15, 10], that help reveal structure in the video volume. Considering a Gaussian-like function of three variables $G(x, y, t) = e^{-(x^2+y^2+t^2)}$, we use its third order derivatives $G3_{\hat{\theta}}(x, y, t) = \frac{\partial^3 G}{\partial \hat{\theta}^3}$ and their Hilbert transforms $H3_{\hat{\theta}}(x, y, t)$, steered to a spatiotemporal orientation of unit vector $\hat{\theta}$ (the symmetry axis of the $G3$ filter). We denote the video volume of stacked frames at a spatial scale $\frac{1}{\sigma}$ with $\mathcal{V}_\sigma$. The energy response for a given $\hat{\theta}$ at a given scale σ is then measured by

$$E_{\hat{\theta},\sigma} = (G3_{\hat{\theta}} * \mathcal{V}_\sigma)^2 + (H3_{\hat{\theta}} * \mathcal{V}_\sigma)^2 \ , \qquad (1)$$

where $*$ denotes a convolution. Note that the Hilbert transform corresponds to a phase shift of $\pi/2$, and the quadrature pair of filters $G3/H3$ allows for extracting spectral strength independent of the phase [15]. The bank of filters is built by choosing a number of scales σ_j and a number of 3D orientations θ_i that span a range of speeds and (2D) orientations

in the image. For example, a filter steered to $\theta = (1, 1, 1)$ would respond to oblique patterns moving at 1 px/frame, while a filter steered to $\theta = (1, 0, 0)$ would capture vertical, static stuctures.

The filter responses (Eq. 1) are sensitive to contrast in the image. We mitigate this with per-pixel normalization with respect to the sum of responses of all filters and defining an additional channel that captures the lack of structure in untextured regions of the image [11]:

$$E'_{\hat{\theta}_i,\sigma_j} = \frac{E_{\hat{\theta}_i,\sigma_j}}{\sum_k E_{\hat{\theta}_k,\sigma_j} + \epsilon} , \quad (2)$$

$$E'_{\epsilon,\sigma_j} = \frac{\epsilon}{\sum_k E_{\hat{\theta}_k,\sigma_j} + \epsilon} , \quad (3)$$

where ϵ is a noise threshold that prevents $E'_{\hat{\theta}_i,\sigma_j}$ from getting large wherever no filter of the bank gives a significant response. All measurements $E'_{\hat{\theta}_i,\sigma_j}$ and E'_{ϵ,σ_j} are concatenated to form L-1 normalized motion histograms. Note that we do not sum responses of multiple filters as in existing work [11, 39], where the authors sought measurements of motion independent of image appearance. We rather assume that such appearance information is valuable and that filter groupings can subsequently be learned within our metric. The motion histograms are conacatenated with classical color histograms [18] to form the feature vector $x_i \in \mathbb{R}^d$ of a pixel or segment.

5. Learning a metric between segments

During the agglomerative segmentation process (Section 3), segments are described by color and motion histograms, and we now show how to learn a metric to compare these segment descriptors. At each iteration of the segmentation, a merge occurs between the two connected segments of smallest mutual distance, and the metric alone thus determines the outcome of the whole process. We learn a generalized Mahalanobis metric that compares descriptors x_1, $x_2 \in \mathbb{R}^d$ as

$$d_L^2(x_1, x_2) = (x_1 - x_2)^\top L^\top L\,(x_1 - x_2) , \quad (4)$$

where $L^T L \in \mathbb{R}^{d\times d}$ is the symmetric positive semi-definite Mahalanobis matrix that defines the metric.

Training data is provided in the form of videos with manual segmentations, from which we generate a set of constraints between pairs of segments of various sizes. To obtain realistic constraints, we simulate the hierarchical segmentation process with an unsupervised distance metric, where merges are only allowed when consistent with the ground truth. After each merge, the updated edges are added to the training set as additional pairwise constraints. Formally, one such constraint between segments i and j is defined by the descriptors x_i and x_j, and an annotation

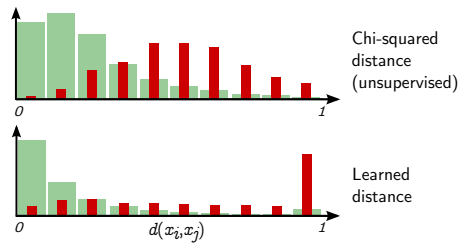


Figure 3. We seek a distance function between segment descriptors that predicts whether they should be merged (green) or not (red). Compared to an unsupervised metric such as a chi-squared distance (upper), the metric learned with our large-margin optimization much better predicts and separates the two outcomes (lower).

$y_{ij} = 1$ if they correspond to a same ground truth label, $y_{ij} = -1$ otherwise. Our goal is to optimize L in Eq. 4 so that the distance between nodes to be merged is small relative to *all other* pairs of segments that should not. Formally, we want

$$d_L(x_i, x_j) \ll d_L(x_k, x_l) \quad (5)$$

$$\forall\, i, j, k, l \;\text{ s.t. }\; y_{ij} = +1,\; y_{kl} = -1 . \quad (6)$$

Note that this contrasts with metric learning designed for nearest-neighbour classification (*e.g.* LMNN [42]) where the distance should only be small to a few points of the same class. This motivates the use of large-margin contraints [34] for each training pair i, j:

$$y_{ij}(t - d_L^2(x_i, x_j)) \;>\; 1 , \quad (7)$$

where t is the separating threshold. This constrains the distance to predict y_{ij} with margin of 1 on either side of t. The constraints are grouped into the following objective function using a hinge-loss formulation:

$$\arg\min_{L,t} \sum_{i,j} \max\Big[\, 1 - y_{ij}\big(t - d_L^2(x_i, x_j)\big),\; 0 \,\Big] \quad (8)$$

We use stochastic subgradient descent to optimize this objective as in [34].

5.1. Joint metric learning and dimensionality reduction

Examining Eq. 4, one can observe that the generalized Mahalanobis distance is equivalent to an Euclidean distance with the data in the projected space transformed by L, *i.e.*

$$d_L^2(x_1, x_2) = ||\,(Lx_1 - Lx_2)\,||_2^2 . \quad (9)$$

One can learn a non-square matrix $L \in \mathbb{R}^{p\times d}$, $p < d$, that projects $x_i \in \mathbb{R}^d$ to a space of lower dimension $\mathbb{R}^p$. This allows us to transform the segment descriptors with L in a preprocessing step, and let the segmentation algorithm work with lower-dimensional data. The dimensionality reduction is readily integrated into the objective function (8)

since we optimize over the matrix L, as opposed to the now rank-deficient Mahalanobis matrix $L'L$. Note that reducing the dimension of our segment descriptors with a linear transform is consistent with the merging of histograms by weighted averging, which keeps the learned metric consistent as segments are merged.

5.2. Multiscale metric learning

The distance metric is used to compare segments of increasing size as the segmentation algorithm proceeds. As mentioned in Section 3, typical histograms of small segments are likely to be different than those of larger ones. Optimal rules for merging them are thus likely to differ as well. In other words, features that matter at a small scale in the image (*e.g.* color consistency) may not matter so much with segments of larger spatial extent (where certain motions may then be more relevant, for example). We therefore experimented with adapting the learned distance to different scales.

For each pair of connected segments i, j, we denote with a_{ij} the image area covered by the smallest of the two. A number S of scales are defined as ranges of values on a_{ij}, and we learn a different L_s for each range ($s = 1..S$). The L_s to apply to compare a given pair of segments i, j is simply determined from the range their a_{ij} falls in. It is crucial for the learned metrics to stay consistent and comparable across scales. Since the optimized objective (8) is not convex in L, we do so by reoptimizing locally L_{s-1} to obtain the L_s of the following scale (see the algorithm listing). This encourages the learned metric to vary smoothly across scales, even though these are defined with fixed, hard thresholds. This also helps to avoid overfitting the scarcer training data of larger scales. Note that using a scale-dependent metric ($S > 1$) does not allow pre-projecting the data into a lower-dimensional space.

Algorithm Supervised learning of a multiscale, large-margin metric between segments from pairwise constraints.

Input:
$\mathcal{C} = \{(x_i, x'_i, y_i, a_i)\}_i$ constraints between pairs of segments
$x_i \in \mathbb{R}^d$: segment descriptors
$y_i = +1/-1$: annotations, same/different ground truth label
a_i: mean per-frame area of the smallest of the two segments
$p \leq d$: effective dimensionality of transformed segment descriptors
$S \geq 1$: number of scales
λ: learning rate of the subgradient descent

Output:
Discriminative projection matrices L_s ($s = 1..S$)

Procedure:
Initialize
$L \leftarrow p$ largest PCA components of all x_i, x'_i in $\mathcal{C}$
$t \leftarrow \frac{1}{2}\left[\ \mathrm{mean}\big(d^2_L(x_i, x'_i),\ y_i{=}{+1}\big) + \mathrm{mean}\big(d^2_L(x_i, x'_i),\ y_i{=}{-1}\big)\ \right]$
Choose S contiguous ranges of a_i from their distribution in $\mathcal{C}$
For each scale $s = 1..S$
Select the subset of constraints $\mathcal{C}' \subset \mathcal{C}$ with $a_i \in \mathrm{range}_s$
Initialize from preceding scale: $L_s \leftarrow L_{s-1}$ and $t_s \leftarrow t_{s-1}$
Loop
Pick constraint i in $\mathcal{C}'$ at random, w/ eq. prob. of $y_i{=}{+1}/-1$
If $y_i\ \big(t - d^2_{L_s}(x_i, x'_i)\big) < 1$ (constraint violated)
$L_s \leftarrow L_s - \lambda\, y_i\, L_s\, (x_i{-}x'_i)\,(x_i{-}x'_i)^\top$
$t_s \leftarrow t_s + \lambda\, y_i\, t_s$
Else Leave L_s, t_s unchanged
Until Obj. (8) no longer decreasing on hold-out validation set
End

6. Results

We evaluate the proposed method on three segmentation tasks that offer clear potential for a supervised approach: dynamic texture segmentation, segmentation of rigid motions, and segmentation of objects from motion boundaries. We compare our results to the respective state of the art for each distinct task. Please consult the supplementary material for additional results and details on the evaluation protocols.

Implementation Our implementation of the segmentation algorithm follows [17]. Color histograms use the Lab space with 3×10 bins. Motion histograms use a filter bank spanning 2 scales, 8 orientations, and 9 speeds between 0 and 2 px/frame, plus a filter that captures flicker. This results in a segment descriptor of size 175 (155 for grayscale videos). The segmentation is bootstrapped at the lowest level using pixel-wise color differences as edge weights, until segments have sufficient size ($a_{\min} = 20$ px) to consider meaningful histograms [18].

Baselines We compare against the respective state-of-the-art methods for each considered task. We also evaluate the value of the proposed contributions against three alternative approaches: (1) unsupervised segmentation using a chi-squared distance between histograms[2] (as in [18]), (2) learning a metric using a simple linear logistic regressor on absolute per-bin differences of histograms (similarly as in [19]), (3) learning a metric with the state-of-the-art ITML algorithm [9], which learns a generalized Mahalanobis distance but without dimensionality reduction.

6.1. Dynamic texture segmentation

We evalute the segmentation of dynamic textures on the SynthDB [3] and Dyntex [31] datasets. The SynthDB dataset [3] consists of collages of K patches of real footage of fire, water, smoke, vegetation, etc. The sequences are very challenging: videos are grayscale, some textures exhibit very similar static appearance (Fig. 4), and adjacent

[2] We use the chi-squared distance as a baseline, which ensures a fair but challenging comparison, as it was consistently shown to be the best-performing among common unsupervised metrics.

textures sometimes have nearly-identical grayscale values. The use of motion is thus crucial to achieve the segmentation of dynamic textures that only differ in their *dynamic* appearance (*e.g.* tree branches moving at different frequencies, or patches of smoke being blown in different directions). We use sequences with $K = 2$ and $K = 3$ for training and testing, respectively. Supervision for training is provided as the ground truth segmentation masks, and semantic labels from 12 classes [30]. We obtained state-of-the-art results, and we detail below variations of our algorithm and training conditions.

Unsupervised Segmentation with intensity histograms alone performs very poorly on such dynamic textures (Fig. 4), which demonstrates the inadequacy of a traditional appearence-based segmentation approach. The proposed filter-based motion features improves the performance significantly, even when performing unsupervised segmentation, which shows that these features capture relevant information.

Learning from manual segmentations We first learn our distance metric using the ground truth segmentation masks, generating our constraints between adjacent segments in the training examples (Section 5). We obtain a consistent improvement over unsupervised segmentation (the Rand index increasing from 72.7% to 89.7%). We plot, in Fig. 4, accuracy against the size of dimension-reduced segment descriptors (p). This figure suggests that dimensionality reduction can be performed aggressively, with a reasonable decrease in performance together with a memory footprint smaller by a factor of 10. This implies that there is a large redundancy in the original descriptors for the present task. Anecdotally, we observed that the dimensions that correspond to the intensity histograms, were given very little importance in the projected descriptors. Moreover, moderate dimensionality reduction may even prove beneficial as a way to avoid overfitting the training data, by discarding irrelevant dimensions from the projected descriptors. This "sweet spot" for the number of dimensions p is easily identifiable by cross-validation on the training data.

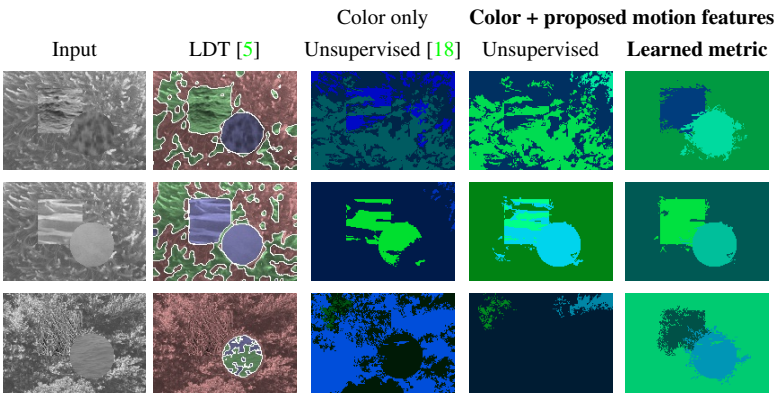
Learning from semantic labels We then use additional constraints from semantic annotations of the training data into 12 classes (*e.g.* grass, river, steam, pond). Constraints are generated between training segments, selected at random, with $y = +1/-1$ for segments with a same/different label (Section 5). The learned distance is now explicitly trained to differentiate textures of different categories. This brings an additional improved in performance at segmenting different types of textures (Rand index of 90.2%). We peform near-perfect segmentation of most of the 100 test sequences (Fig. 4). We surpass all results reported on this dataset, in particular those of [39] that enforced *static* segments (whereas we perform a true video segmentation), and those of [6] that use handcrafted, specially-designed descriptors.

We now evaluate our ability to segment dynamic textures in real videos of the Dyntex dataset [31]. We use the distance learned on the SynthDB dataset (from manual segmentations and semantic labels, p=d) to segment the scenes used by previous authors [13, 6]. As seen in Fig. 5, these sequences are particularly challenging. They include various dynamic textures (steam, water splashes and ripples, smoke, etc.), some recorded with a moving camera. We are able to segment these textures with very precise contours. Athough no ground truth is available, our results are qualitatively as good or better than those of existing methods [13, 6].

6.2. Object boundaries

Next, we consider the task of segmenting objects from their motion boundaries on the CMU dataset [35]. This has previously been addressed, both with unsupervised segmentation approaches [39] and specific methods using classifiers trained on candidate boundaries (edges) in the image. The dataset is challenging, as it includes a mix of rigid and non-rigid motions, caused by parallax at different depths and by intrinsic motion (animals and humans). This dataset contains 30 sequences, which we use to train a similarity metric using segment annotations as in Section 5. We show in Fig. 3 how our optimization algorithm is able to find a distance measure that predicts the probability of merging pairs of adjacent segments (plotted using the ground truth for visualization), with a large margin between the two. This translates into a significant improvement in segmentation over the unsupervised case (Fig. 6, upper). We are able to recover object boundaries with performance close to the state-of-the-art methods of this dataset. Superior methods make use of other cues such as geometric characteristics of the boundaries and the segments. One possibility to explore within our framework would be to extend the learning of the distance measure to a general function that uses characteristics of segments other than arithmetic differences between histograms.

We also evaluate the effect of dimensionality reduction of the segment descriptors (Fig. 6, lower). We observe again the desirable smooth reduction in performance with smaller segment descriptors, that allows significant improvement in memory usage for neglibile or only a modest loss in accuracy. The smaller dimensionality once again proves beneficial in avoiding the over-fitting of the training data. We compare the proposed dimensionality reduction, optimized jointly with the metric, against a trivial projection on the largest PCA components (Fig. 6, lower). It shows a clear advantage for the joint approach that integrates both goals of a small descriptor size and a highly discriminative metric.



Method		Avg. Rand (%)
Features	*Metric*	
Color	Unsupervised	59.9
Color + motion	Unsupervised	72.7
Color + motion	Learned, logistic regression	77.1
Color + motion	Learned, ITML [9]	86.4
Color + motion	Learned from manual segm.	89.7
Color + motion	Learned from manual segm. and semantic labels (p=d)	**90.2**
GPCA [33]		55.4
LDT [5] (with manual initialization)		89.4
DTM (CS) [3] (static segments)		82.5
$(\text{LBP/WLD})_{\text{TOP}}$ [6]		88.4

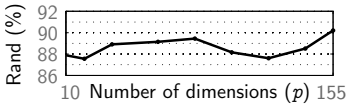


Figure 4. We obtain state-of-the-art results on the segmentation of dynamic textures (SynthDB dataset), as measured by the Rand index averaged over the 100 test sequences. (Lower right) The proposed dimensionality reduction allows only a modest decrease of performance while improving the memory footprint by a factor of more than 10.

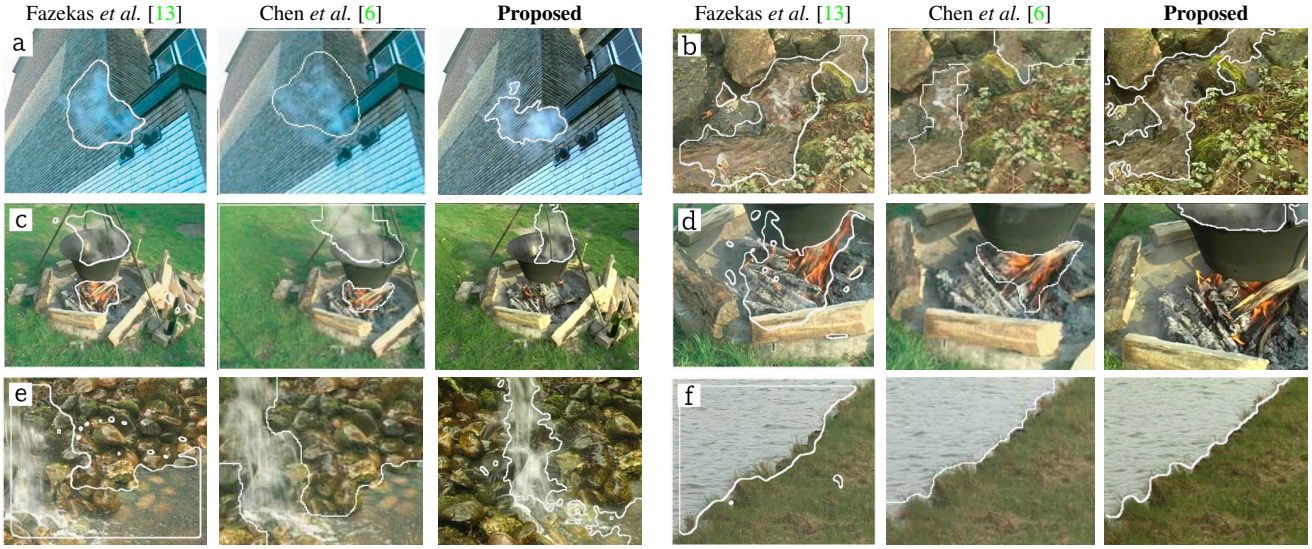
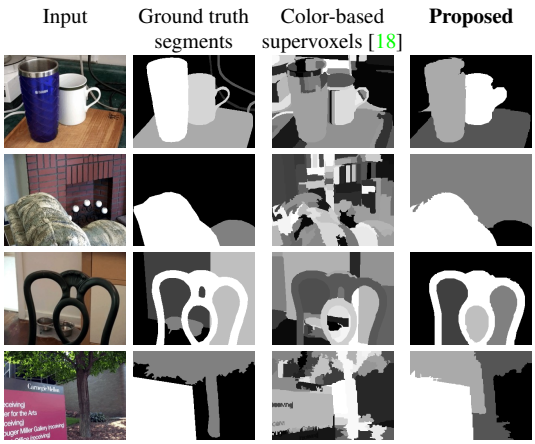


Figure 5. Segmentation of dynamic textures in complex scenes of the Dyntex dataset. We recover much more precise boundaries than existing methods. (a) Steam coming from a ventilation system, (b) A narrow creek winding between pebbles, (c) Steam while cooking on a campfire, (d) Closer shot of the same scene, (e) Water falling over round pebbles and gathering into a pool, (f) A canal and shoreline in Amsterdam. Sequences (a,b,d,f) were shot with a moving camera.

6.3. Rigid motions

We now consider the segmentation of rigid motions with the MIT human-labeled dataset [28]. It features objects with intrinsic motion (*e.g.* car, dog) and parallax-induced motions at different depths. Due to the small size of the dataset (only 9 sequences), we perform leave-one-out evaluation. Segmentation with the learned distance function provides a small improvement over unsupervised segmentation for most sequences of the dataset. We believe the limited advantage of the supervised approach on this dataset is due to the relative ease for segmenting rigid motions using *unsupervised* methods. The small amount of training data makes the learning approach prone to overfitting. The "hand" sequence, for example (Fig. 7), performs poorly, as the learning downweights the appearance features (since training annotations correspond mostly to rigid motions), even though they alone lead to a perfect segmentation in the unsupervised setting. The small training set is also the most likely reason why adapting the metric to multiple scales (Section 5.2) does not result in any improvement. Overall, we still perform better than a number of existing methods [36, 37], and close to the state-of-the-art results reported in [39] (see Fig. 7 and details in supplementary material).

We compare the proposed learning algorithm with a baseline solution using logistic regression, and with the



Method		Boundary F-measure (%)	Boundary AP (%)	Segments Rand (%)
Features	*Metric*			
Color histograms only	Unsupervised	51.1	43.2	60.3
Color + motion	Unsupervised	60.8	61.8	68.3
Color + motion	Learned as proposed, S=1	64.1	65.3	72.1
Color + motion	Learned as proposed, S=2	64.3	65.4	72.3
Color + motion	Learned as proposed, S=3	64.4	65.5	72.4
Stein and Hebert [35]		66.7	63.7	—
Sargin *et al.* [32]		57.0	58.0	—
Sundberg *et al.* [38]		61.0	72.0	—
He *et al.* [21]		47.0	43.0	—
Teney and Brown [39]		60.7	56.3	—

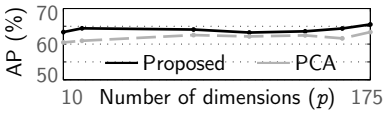
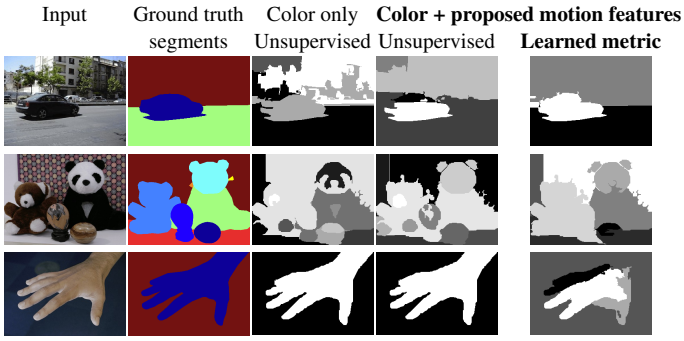


Figure 6. (Left) Object segmentation from occlusion boundaries on the CMU dataset of sequences with camera translation. (Upper-right) Our results, in terms of F-measure of segment boundaries (64.4%), are comparable with the best existing task-specific methods. (Lower-right) The proposed dimensionality reduction, optimized jointly with the metric, performs significantly better than a naive projection on the first d PCA components followed by unsupervised segmentation. This highlights the importance of considering dimensionality reduction and metric learning as a combined objective.



Method		Avg. Rand (%)
Features	*Metric*	
Color histograms only	Unsupervised	68.5
Color + motion	Unsupervised	78.8
Color + motion	Learned, logistic reg.	76.3
Color + motion	Learned, ITML [9]	70.3
Color + motion	**Learned as proposed, S=1**	**83.0**
Color + motion	Learned as proposed, S=2	81.1
Layers++ [36]		77.5
nLayers [37]		82.3
Teney and Brown [39]		**83.2**

Figure 7. Motion segmentation on the MIT human-labeled dataset. Most sequences benefit from the learned approach, but the small size of the dataset increases the risk of overfitting. The annotations, which correspond to rigid motions, tends to downweight the appearance features, even though they were sufficient alone for segmenting the "hand" sequence (last row of images).

state-of-the-art ITML algorithm. From our observations, a distance based on a linear logistic regression does not offer enough degrees of freedom to capture relevant relationships between dimensions of the segment descriptors. This supports the choice of a generalized Mahalanobis as the learned metric. Conversely, the ITML algorithm seemed either prone to overfitting, or unable to model the relevant training constraints, most likely due to the rigid regularizer (using an identity matrix as the reference as in [9]).

7. Conclusions and future work

This paper discussed the segmentation of videos of dynamic scenes using a novel combination of filter-based motion features and a supervised learning approach. We use the responses to a large bank of spatiotemporal filters to capture a wide range of phenomena, including cases where optical flow fails [39]. We use these features within a graph-based video segmentation algorithm, that we extend via a learned metric to measure the dissimilarity between segment. The objective metric learning includes a reduction of dimensionaility of the descriptors, which alleviates the large memory requirements of the original algorithm [18]. We obtained state-of-the-art results on the segmentation of dynamic textures and demonstrated wider applicability to general motion segmentation, with results comparable to the best task-specific methods. One insight brought by this extensive evaluation is that a general-purpose video segmentation framework can be successfully applied to a variety of tasks by incorporating specific rules learned from annotated, manually segmented videos. As noted before [16, 28], there is an obvious bias in human annotations and in each dataset, and it will be interesting in the future to address the learning of more *general* rules and cross-dataset performance. This should involve the annotation or the addition of more dynamic textures in video segmentation benchmarks.

References

[1] S. Alpert, M. Galun, R. Basri, and A. Brandt. Image segmentation by probabilistic bottom-up aggregation and cue integration. In *CVPR*, pages 1–8, 2007. 2

[2] T. Brox and J. Malik. Object segmentation by long term analysis of point trajectories. In *ECCV*, pages 282–295. 2010. 2

[3] A. Chan and N. Vasconcelos. Modeling, clustering, and segmenting video with mixtures of dynamic textures. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(5):909–926, 2008. 5, 7

[4] A. Chan and N. Vasconcelos. Variational layered dynamic textures. In *CVPR*, pages 1062–1069, 2009. 2

[5] A. B. Chan and N. Vasconcelos. Layered dynamic textures. *IEEE Trans. Pattern Anal. Mach. Intell.*, 31(10):1862–1879, 2009. 1, 2, 7

[6] J. Chen, G. Zhao, M. Salo, E. Rahtu, and M. Pietikäinen. Automatic dynamic texture segmentation using local descriptors and optical flow. *IEEE Trans. Image Processing*, 22(1):326–339, 2013. 2, 6, 7

[7] D. Chetverikov and R. Peteri. A brief survey of dynamic texture description and recognition. In *Int. Conf. Computer Recognition Systems*, pages 17–26. Springer, 2005. 2

[8] J. J. Corso. Cvpr tutorial on video segmentation. 2014. 3

[9] J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon. Information-theoretic metric learning. In *ICML*, pages 209–216. ACM, 2007. 5, 7, 8

[10] K. G. Derpanis and J. M. Gryn. Three-dimensional nth derivative of gaussian separable steerable filters. In *ICIP (3)*, pages 553–456, 2005. 3

[11] K. G. Derpanis and R. P. Wildes. Spacetime texture representation and recognition based on a spatiotemporal orientation analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(6):1193–1205, 2012. 1, 2, 4

[12] G. Doretto, P. Pundir, S.Soatto, and Y. N. Wu. Dynamic textures. In *IJCV*, pages 439–446, 2001. 2

[13] S. Fazekas, T. Amiaz, D. Chetverikov, and N. Kiryati. Dynamic texture detection based on motion analysis. *IJCV*, 82(1):48–63, 2009. 6, 7

[14] C. Feichtenhofer, A. Pinz, and R. Wildes. Bags of spacetime energies for dynamic scene recognition. In *CVPR*, 2014. 2

[15] W. T. Freeman and E. H. Adelson. The design and use of steerable filters. *IEEE Trans. Pattern Anal. Mach. Intell.*, 13(9):891–906, 1991. 2, 3

[16] F. Galasso, N. S. Nagaraja, T. J. Cardenas, T. Brox, and B. Schiele. A unified video segmentation benchmark: Annotation, metrics and analysis. In *ICCV*, December 2013. 1, 2, 8

[17] M. Grundmann, V. Kwatra, M. Han, D. Castro, and I. Essa. Graph-based hierarchical video segmentation. In *CVPR Tutorial on Video Segmentation*, 2014. 3, 5

[18] M. Grundmann, V. Kwatra, M. Han, and I. A. Essa. Efficient hierarchical graph-based video segmentation. In *CVPR*, pages 2141–2148, 2010. 1, 2, 3, 4, 5, 7, 8

[19] M. Guillaumin, J. Verbeek, and C. Schmid. Is that you? metric learning approaches for face identification. In *ICCV*, pages 498–505, 2009. 5

[20] M. Haindl and S. Mike. Unsupervised dynamic textures segmentation. In *Computer Analysis of Images and Patterns*, volume 8047 of *LNCS*, pages 433–440. Springer Berlin Heidelberg, 2013. 2

[21] X. He and A. Yuille. Occlusion boundary detection using pseudo-depth. In *ECCV*, pages 539–552. 2010. 8

[22] D. J. Heeger. Model for the extraction of image flow. *J. Opt. Soc. Am. A*, 1987. 2

[23] A. Humayun, O. M. Aodha, and G. J. Brostow. Learning to find occlusion regions. In *CVPR*, pages 2161–2168, june 2011. 2

[24] V. Jain, S. C. Turaga, K. L. Briggman, M. Helmstaedter, W. Denk, and H. S. Seung. Learning to agglomerate superpixel hierarchies. In *NIPS*, pages 648–656, 2011. 2

[25] H. Jung, J. Ju, and J. Kim. Rigid motion segmentation using randomized voting. In *CVPR*, June 2014. 2

[26] A. Khoreva, F. Galasso, M. Hein, and B. Schiele. Learning must-link constraints for video segmentation based on spectral clustering. In X. Jiang, J. Hornegger, and R. Koch, editors, *Pattern Recognition*, Lecture Notes in Computer Science, pages 701–712. Springer International Publishing, 2014. 2

[27] F. Li, T. Kim, A. Humayun, D. Tsai, and J. M. Rehg. Video segmentation by tracking many figure-ground segments. In *ICCV*, 2013. 1

[28] C. Liu, W. T. Freeman, E. H. Adelson, and Y. Weiss. Human-assisted motion annotation. In *CVPR*, 2008. 2, 7, 8

[29] C. D. Manning, P. Raghavan, and H. Schtze. Hierarchical clustering. In *Introduction to Information Retrieval*, chapter 17. Cambridge University Press, Cambridge, UK, 2008. 3

[30] A. Mumtaz, E. Coviello, G. R. G. Lanckriet, and A. B. Chan. Clustering dynamic textures with the hierarchical em algorithm for modeling video. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(7):1606–1621, 2013. 6

[31] R. Péteri, S. Fazekas, and M. J. Huiskes. DynTex : a Comprehensive Database of Dynamic Textures. *Pattern Recognition Letters*. http://projects.cwi.nl/dyntex/. 5, 6

[32] M. Sargin, L. Bertelli, B. Manjunath, and K. Rose. Probabilistic occlusion boundary detection on spatio-temporal lattices. In *ICCV*, pages 560–567, 2009. 8

[33] J. Shi and J. Malik. Motion segmentation and tracking using normalized cuts. In *ICCV*, pages 1154–1160, 1998. 7

[34] K. Simonyan, O. M. Parkhi, A. Vedaldi, and A. Zisserman. Fisher Vector Faces in the Wild. In *BMVC*, 2013. 2, 3, 4

[35] A. N. Stein and M. Hebert. Occlusion boundaries from motion: Low-level detection and mid-level reasoning. *IJCV*, pages 325–357, 2009. 2, 6, 8

[36] D. Sun, E. Sudderth, and M. J. Black. Layered image motion with explicit occlusions, temporal consistency, and depth ordering. *Advances in Neural Information Processing Systems*, 23, 2010. 7, 8

[37] D. Sun, E. B. Sudderth, and M. J. Black. Layered segmentation and optical flow estimation over time. In *CVPR*, pages 1768–1775, 2012. 2, 7, 8

[38] P. Sundberg, T. Brox, M. Maire, P. Arbelaez, and J. Malik. Occlusion boundary detection and figure/ground assignment from optical flow. In *CVPR*, pages 2233–2240, 2011. 8

[39] D. Teney and M. Brown. Segmentation of Dynamic Scenes with Distributions of Spatiotemporally Oriented Energies. In *BMVC*, 9 2014. 2, 4, 6, 7, 8

[40] D. Tsai, M. Flagg, and J. M.Rehg. Motion coherent tracking with multi-label mrf optimization. *BMVC*, 2010. 1

[41] F. Wang and J. Sun. Survey on distance metric learning and dimensionality reduction in data mining. *Data Mining and Knowledge Discovery*, pages 1–31, 2014. 2

[42] K. Weinberger and L. Saul. Distance metric learning for large margin nearest neighbor classification. *The Journal of Machine Learning Research*, 10:207–244, 2009. 3, 4

[43] C. Xu and J. J. Corso. Evaluation of super-voxel methods for early video processing. In *CVPR*, 2012. 1, 2

[44] C. Xu, S. Whitt, and J. J. Corso. Flattening supervoxel hierarchies by the uniform entropy slice. In *ICCV*, 2013. 2

[45] C. Xu, C. Xiong, and J. J. Corso. Streaming hierarchical video segmentation. In *ECCV*, 2012. 2