Spatio-temporal tube segmentation through a video metrics-based patch similarity measure

Patricia Vitoria[1,2], Vadim Fedorov[1], and Coloma Ballester[1]

[1] *Universitat Pompeu Fabra, Spain and* [2] *Technical University of Munich, Germany*

Abstract

This paper presents a new method for video simplification which identifies moving or static objects in a video by grouping together all the pixels corresponding to homogeneous texture and temporal coherent spatio-temporal regions, the so-called *tubes*. This is achieved through the proposal of video metrics defined on the video domain, which provide patches that intrinsically and automatically adapt its spatio-temporal shape and size, and a patch-based comparison measure that is able to capture the scene similarities. Building upon this video metrics-based patch similarity measure we propose a video simplification method that results in tubes made of the pixels with the same vicinity texture content regardless camera point of view or object position changes. We present experiments analyzing the performance of the proposed approach.

Keywords: patch-based method, supervoxel, tube computation, patch similarity, video segmentation.

1 Introduction

Decomposition and understanding dynamic scenes remains a significant challenge in computer vision. One of the first steps in the scene analysis is the segmentation of it into supervoxels. The output data - supervoxels - has more significant information than each of its pixels and the complexity of the input video is reduced. Changes of position (thus producing affine or projective distortions), illumination, and the interaction of the objects with the surrounding (thus sometimes disappearing or being partially occluded or disoccluded) makes the problem even more challenging.

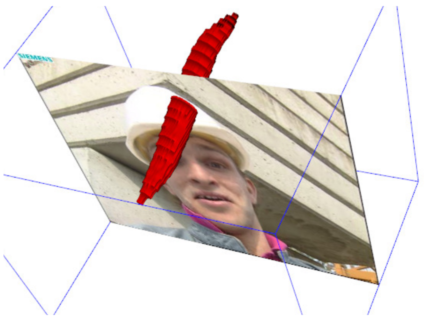
Figure 1: Spatio-temporal adaptive patch (in red) corresponding to the metric tensor of a point close to the helmet boundaries of the well-known Foreman video sequence where the man speaks and moves his head.

Video segmentation is the basis for many applications including video representation, compression, tracking, motion analysis, object recognition and dynamic scene analysis and visualisation.

In this paper, we seek to obtain a spatio-temporal over-segmentation of the video, called *tubes*. The proposed tubes will encode a texture-based and temporal coherent segmentation of the regions in the video that respect boundaries and allow large displacements. It is able to deal with the significant region appearance changes that may happen over the frames due to the suffered perspective distortions caused by changes in the camera viewpoint or object motion. We attain this by, firstly, analysing the video using appropriate varying Riemannian metrics defined on the video domain that capture those distortions and provide adaptive spatio-temporal neighborhoods or patches. Secondly, we use an appropriate video similarity measure which uses these metrics to automatically transform the patches to compare them in an affine invariant manner. These tools are incorporated in our proposal for spatio-temporal tube computation which grows a tube by relevant patch comparison.

The remainder of the paper is organized as follows. In section 2 we revise previous work in supervoxel segmentation and video segmentation. Section 3 presents the proposed approach. Section 4 explains our algorithm and the most important implementation details while some results are presented in Section 5. Finally Section 6 concludes the paper and discusses future work.

2 State of the Art

Superpixel segmentation is an over-segmentation technique that simplifies an input image by grouping together into regions pixels following a similarity criterion. The output data has more compact and meaningful information than a single pixel and can be used as a preprocessed input data for several algorithms. Supervoxels extend the notion of superpixels along frames or over a volumetric object giving a 3-dimensional segmented region, also called *tube*, which aims to represent the trajectory in the scene of each moving object [Grundmann et al., 2010, Lezama et al., 2011, Trichet and Nevatia, 2013, Brendel and Todorovic, 2009]. The main application for supervoxels is video segmentation, although it has also been used for many others including video representation, compression or recognition.

Video segmentation can be associated with labeling or tracking an object or region over time. Some authors addressed video segmentation via analyzing each frame separately. First, individuals frames are segmented independently (superpixel extraction) and second, regions from frame to frame are matched. To find the same region over several frames, the optical flow [Trichet and Nevatia, 2013] or a distance function can be used [Oneata et al., 2014, Kwak et al., 2015]. This is a challenging approach, due to the changes in illumination, point of view, rotation, etc, that can suffer each region from one frame to another, and often gives an inconsistent segmentation. In addition, the estimation of the optical flow to subsequently segment the sequence is seen as a chicken-egg problem. A solution proposed by [Cremers, 2007] is to jointly solve the segmentation and motion estimation problems by minimizing a single functional in a Bayesian inference framework. Other authors use deep learning to face the problem. In [Khoreva et al., 2016], the algorithm learns how to refine the detected masks frame by frame, by using the detection of the previous frame together with optical flow and post-processing with Conditional Random Fields. In [Jampani et al., 2016], the training of Convolutional Neural Networks (CNN) is combined with bilateral filtering. In [Caelles et al., 2016], each frame is segmented independently using a fully CNN architecture using as a input the mask of the first frame.

Last, it is important to remark, that most of the tracking scenarios often use bounding boxes that does not adapts to the contour of the object [Nam and Han, 2016, Hare et al., 2016, Zhong et al., 2012]. Supervoxels, by contrast, seek to adapt the region to the boundary of the object.

3 Proposed model

Video segmentation methods aim at characterizing the spatio-temporal regions of the video domain having homogeneous texture regardless of differences in the point of view or suffered perspective distortion due to the movement and interactions of the objects. To this goal we propose to first analyse and describe the video by using appropriate spatially varying Riemannian metrics defined on the video domain that capture those distortions and provide adaptive spatio-temporal neighborhoods or patches. Morever, we use an appropriate similarity measure which uses these metrics to automatically transform the video patches to compare them in an affine invariant manner. Finally, we propose a method for video simplification and spatio-temporal tube computation. Sections 3.1, 3.2 and 3.3 present, respectively, each of the three ingredients of our proposal.

3.1 A video metric given by appropriate structure tensors

Given a video $u(\mathbf{x}, t)$ defined on a video domain $\mathcal{M} = \{(\mathbf{x}, t) : \mathbf{x} \in \Omega \subset \mathbb{R}^2, t \in \mathbb{R}\}$, our first aim is to endow the video domain $\mathcal{M}$ with a suitable metric depending on the video sequence which incorporates the temporal distances along the visible trajectory of the object pixels and also edge preserving anisotropies. Trajectories are defined by the dense optical flow, that is, a vector field that recovers the apparent motion of two consecutive frames. We build on the video metrics that appeared in [Calderero and Caselles, 2014] for multiscale analyses classification purposes. Structure tensors were identified there as the natural metric obtained by considering the image (or video) as a manifold of patches. In the image processing and computer vision literature, the structure tensor, also referred as the second-moment matrix, has been considered as a metric in the image domain [Brox et al., 2006], [Peyré, 2009], [Weickert, 1998], [Fedorov et al., 2015]. It contains information about the predominant directions of the gradient in a specific neighborhood of a point.

In our work, we stem from a video structure tensor definition and propose an iterative method that enforces affine covariance. Those approximate affine covariant structure tensors will be intrinsically endowed with affine covariant neighborhoods providing adaptive space and time patches. Affine invariance, viewed as a simplified projective invariance, is an essential requirement for the analysis of natural scenes. In the video setting, the affine covariance is again an interesting property: if we turn a camera around its axis with a fixed angle, or if we change the format of the images, we expect to identify the same similarities among the moving objects (up to the discretization problems) [Guichard, 1998].

Let us recall that, in general, if $u : \mathbb{R}^N \to \mathbb{R}$ is any given image defined on $\mathbb{R}^N$, an image-dependent tensor field T_u is defined as a function that associates to each point $x \in \mathbb{R}^N$, a tensor, that is a symmetric, positive semi-definite $N \times N$ matrix $T_u(x)$. The tensor field is said to be *affine covariant* if for any affinity A given by a non-singular $N \times N$ matrix, T_u satisfies $T_{u_A}(x) = A^T T_u(Ax)A$, where $u_A(x) := u(Ax)$ denotes the affined transformed version of u. Several affine covariant structure tensors were proposed in [Fedorov et al., 2015]. Given a tensor $T_u(x)$ we can naturally associate to it an ellipsoidal or elliptical region of "radius" $r > 0$ centered at x

$$B_{T_u}(x, r) = \{y : \langle T_u(x)(y - x), (y - x)\rangle \le r^2\}. \tag{1}$$

When the tensor is affine covariant, we have $AB_{T_u(x,r)} = B_u(Ax, r)$. In other words, the associated regions are affine covariant; they geometrically transform appropriately via an affinity. We refer to these regions as *shape-adaptive* or *ellipsoidal patches*. Figure 1 shows and example for the video case ($N = 3$) for the video structure tensor proposed below. For a point $\mathbf{x}$ close to the helmet boundaries in the *Foreman* sequence, the corresponding shape-adaptive patch is displayed in red.

To present the structure tensor we propose to use, let us denote by $\mathbf{v}$ the dense optical flow of the input video u. That is, $\mathbf{v}(\mathbf{x}, t)$ denotes the apparent motion between a pixel $\mathbf{x}$ on frame at time t and the corresponding at time $t + 1$, for $(\mathbf{x}, t) \in \mathcal{M}$. Let us consider the following video structure tensor [Calderero and Caselles, 2014]

$$T_u(\mathbf{x}, t) = \int_{\mathcal{E}_g((\mathbf{x},t),r)} D_{\mathbf{x}t} u((\mathbf{x}, t) + (\mathbf{y}, \tau)) \otimes D_{\mathbf{x}t} u((\mathbf{x}, t) + (\mathbf{y}, \tau))\mu(\mathbf{y}, \tau)|G|^{1/2}\, d\mathbf{y}d\tau. \tag{2}$$

where $\mathcal{E}_g((\mathbf{x}, t), r) = \left\{Y = (\mathbf{y}, s) : g_{ij}(\mathbf{x}, t)Y^i Y^j \le r^2\right\}$, g is an initial metric on $\mathcal{M}$, $|G|$ denotes the determinant of the symmetric matrix $G = (g_{ij})$ and μ is a weight measure on $\mathcal{E}_g((\mathbf{x}, t), r)$, be either the usual Lebesgue measure or a weighting function. Our initial metric g is given by

$$g(\mathbf{x}, t)\left((\mathbf{y}, \tau), (\mathbf{y}, \tau)\right) = a(\mathbf{x}, t)(\mathbf{y} - \mathbf{v}(\mathbf{x}, t)\tau)^2 + b(\mathbf{x}, t)\tau^2 \quad (\text{equivalently}, G(\mathbf{x}, t) = \begin{pmatrix} aI & -a\mathbf{v} \\ (-a\mathbf{v})^t & b + a|\mathbf{v}|^2 \end{pmatrix}(\mathbf{x}, t)), \tag{3}$$

where $(\mathbf{y}, \tau)$ denote coordinates in an infinitesimal neighborhood of $(\mathbf{x}, t)$ and the functions a and b are defined as $a(\mathbf{x}, t) = \alpha_1 + \alpha_2|\nabla_{\mathbf{x}} u|^p$, $b(\mathbf{x}, t) = \beta_1 + \beta_2(\partial_{\mathbf{v}} u)^p$, with $\alpha_1, \alpha_2, \beta_1, \beta_2 > 0$, $p > 0$ (usually $p = 1,2$), and $\partial_v u = \mathbf{v} \cdot \nabla_x u + u_t$ denotes the convective derivative. The meaning of the metric g is that being at the point $(\mathbf{x}, t)$ of a video a displacement $(\mathbf{y}, \tau)$ of a point is not penalized if it follows the law of motion at $(\mathbf{x}, t)$. g was tested in [Calderero and Caselles, 2014] in the context of video filtering.

The video structure tensor $T_u(\mathbf{x}, t)$ in (2) is computed on the spatio-temporal neighborhood $(\mathbf{x}, t) + \mathcal{E}_G(\mathbf{x}, t)$, which was proven to be equivalent infinitesimally to a motion compensated patch of radius r. We compute it using the compensated video. We propose to enforce affine covariance by an iterative scheme similar to the one used in [Fedorov et al., 2015] for the 2D case. In our case, the computation begins with the initial video region given by the initial metric G and proceeds with alternating computation of the structure tensor $T_u^k(\mathbf{x}, t)$

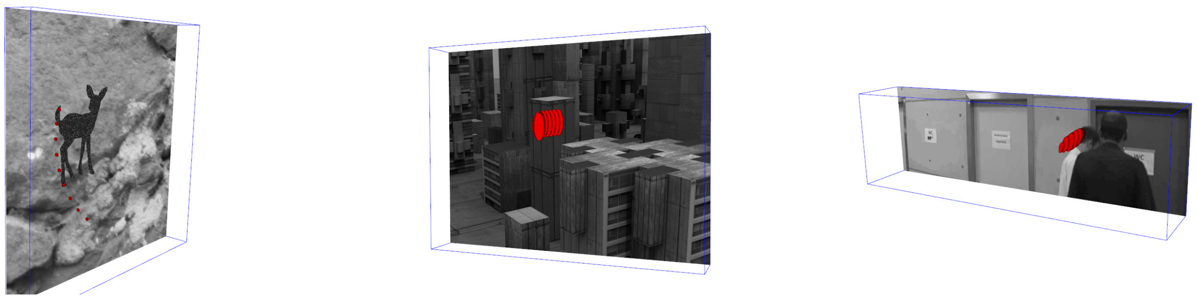

Figure 2: From left to right: Visualization of 3D-shape adaptive patches (in red) corresponding to the tensor on a point on the tail of the dear, on the wall of a building with vertical lines/texture and on the man's head (more details on the sequence movement are given in the text).

and its corresponding region $B_{T_u^k}((\mathbf{x},t),r)$ defined as in (1), where $T_u^k(\mathbf{x},t)$ denotes the structure tensor (2) computed on $B_{T_u^{k-1}}((\mathbf{x},t),r)$, normalized by its volume. In Section 4, we give the details of our algorithm for this $k-$iterative computation. Figure 2 shows some examples of adaptive patches in video. Let us notice, e.g., that the patch *follows the motion* (the deer rotates in the left-most sequence -see also Figure 5- and the camera moves in the *Urban2 Middlebury* sequence in the middle) or *stops* and adapts its shape when there are some occlusions in time (like in the right-most example). Let us remark that r is a free parameter which controls its size. Nevertheless, the size of each of them is also affected by the texture in the vicinity of the corresponding point allowing to an automatic and intrinsic adaptation.

3.2 A video similarity measure

Using affine covariant structure tensors as Riemannian metrics in $\mathbb{R}^N$, explicit formulas for multiscale affine invariant similarity measures were obtained in [Fedorov et al., 2015]. The patch similarity used in our work is an adaptation to the video case of one of them. One of the key points is to exploit the fact that the Riemannian metrics defined on the center point of the two patches in comparison (say $\mathbf{x}$ and $\mathbf{y}$) allow to automatically transform the patches to compare them in an appropriate manner. Moreover, if the local vicinities of $\mathbf{x}$ and $\mathbf{y}$ are related by a local affininity A such that $\mathbf{y} = A\mathbf{x}$, it was shown in [Fedorov et al., 2015] that from the affine covariant structure tensors we can recover the affine distortion up to a rotation. Indeed, $A = T_u(A\mathbf{x})^{\frac{-1}{2}} R T_{u_A}(\mathbf{x})^{\frac{1}{2}}$ for some orthogonal transformation R. This formula is at the basis of the affine invariant multiscale similarity measures studied there. It involves an intuitive idea: In order to compare the image (or video) content on patches centered at points $\mathbf{x}$ and $\mathbf{y}$, we can proceed in three steps: First by applying $T_u(\mathbf{x})^{\frac{1}{2}}$ the shape-adaptive patch, given by (1), is transformed to a disc (or sphere) of radius r. The resulting patch is called *normalized patch*. Second, we rotate the normalized patch using R. Finally, $T_u(\mathbf{y})^{\frac{-1}{2}}$ maps the rotated normalized disc to the shape-adaptive patch at $\mathbf{y}$. The rotation R is determined from the image content at the shape-adaptive patches and, in practice, is split into two parts such that $R = \tilde{R}(\mathbf{y})^{-1}\tilde{R}(\mathbf{x})$, where $\tilde{R}(\mathbf{x})$ and $\tilde{R}(\mathbf{y})$ depends on the image content around only one point, $\mathbf{x}$ or $\mathbf{y}$, respectively.

Let us now introduce the video patch similarity measure used in this work. We present it in the (slightly more general) context of comparing patches of two video sequences u_1 and u_2 defined on $\mathscr{M}_1 = \{(\mathbf{x},t) : \mathbf{x} \in \Omega_1,\, t \in \mathbb{R}\}$ and $\mathscr{M}_2 = \{(\mathbf{x},t) : \mathbf{x} \in \Omega_2,\, t \in \mathbb{R}\}$, respectively. Obviously, we might well have $u_1 = u_2$. Let $T_1(\mathbf{x}_1,t_1)$ be the structure tensor of u_1 at $(\mathbf{x}_{u_1},t_1)$ defined as in previous Section 3.1, for each point $(\mathbf{x}_{u_1},t_1)$ of the video domain $\mathscr{M}_1$. Analogously, let $T_{u_2}(\mathbf{x}_2,t_2)$ defined as in Section 3.1 be the structure tensor of u_2 at $(\mathbf{x}_2,t_2)$. These structure tensor fields provide metrics on $\mathscr{M}_1$ and $\mathscr{M}_2$, respectively. Then, the patch distance to compare the patches of u_1 and u_2 at $(\mathbf{x}_1,t_1)$ and $(\mathbf{x}_2,t_2)$ is defined by

$$\mathscr{D}((\mathbf{x}_1,t_1),(\mathbf{x}_2,t_2),s) =$$
$$\int_{\Delta_s} g_s(y,\tau)\left(u_1\left((\mathbf{x}_1,t_1) + T_1(\mathbf{x}_1,t_1)^{-\frac{1}{2}} R_{u_1}^{-1}(\mathbf{x}_1,t_1)(y,\tau)\right) - u_2\left((\mathbf{x}_2,t_2) + T_2(\mathbf{x}_2,t_2)^{-\frac{1}{2}} R_{u_2}^{-1}(\mathbf{x}_2,t_2)(y,\tau)\right)\right)^2 dy d\tau. \tag{4}$$

where s is the so-called *scale*, g_s is a weighting function (for instance a Gaussian of variance s) and Δ_s denotes a sphere centered at the origin with radius proportional to s and big enough such that g_s has effective support in Δ_s. Let us give a geometrical interpretation of (4): The two shape-adaptive ellipsoidal patches at $(\mathbf{x}_1, t_1)$ and $(\mathbf{x}_2, t_2)$ are first normalized by $R_{u_1} T_1(\mathbf{x}_1, t_1)^{\frac{1}{2}}$ and $R_{u_2} T_2(\mathbf{x}_2, t_2)^{\frac{1}{2}}$, respectively, to spheres of the same size and then compared. Section 4 presents the implementation details.

3.3 Proposed method for spatio-temporal tube computaion

Our tube computation algorithm is based on grouping together all the pixels with similar spatio-temporal local structure. To compute the similarity between two points $(\mathbf{x}_1, t_1)$ and $(\mathbf{x}_2, t_2)$ in the domain of a given video u with optical flow $\mathbf{v}$, we propose to use the patch similarity distance (4) between the shape-adaptive patches defined by the proposed approximate affine covariant structure tensors $T_u(\mathbf{x}_1, t_1)$ and $T_u(\mathbf{x}_2, t_2)$ on those points, and also take into account the Euclidean distance between the corresponding optical flow at $(\mathbf{x}_1, t_1)$ and $(\mathbf{x}_2, t_2)$.

More precisely, given a point $(\mathbf{x}_1, t_1)$, in order to obtain the spatio-temporal tube of this point, we proceed by a region growing procedure where we iteratively check if the neighboring points $(\mathbf{x}_2, t_2)$ (in a neighborhood $U(x_1, t_1)$, see Section 4) belong to the same tube. A point $(\mathbf{x}_2, t_2)$ is considered as belonging to the tube of $(\mathbf{x}_1, t_1)$ if the distance

$$\widetilde{\mathscr{D}}\left((\mathbf{x}_1, t_1), (\mathbf{x}_2, t_2), s\right) = \mathscr{D}\left((\mathbf{x}_1, t_1), (\mathbf{x}_2, t_2), s\right) + \alpha \mathscr{D}_E\left(\mathbf{v}(\mathbf{x}_1, t_1), \mathbf{v}(\mathbf{x}_2, t_2)\right) \tag{5}$$

is below a given threshold $\epsilon > 0$, where $\alpha \geq 0$ is weight factor and s is the scale. We iteratively verify if the neighboring points of the new point of the tube belong to the tube. In the following section we give a summary of all the steps of our algorithm, and the most important details of it.

4 Algorithm

Given a video u with optical flow $\mathbf{v}$, firstly, the normalized video structure tensor for each point $(\mathbf{x}, t)$ is computed with the following iterative process, where $k > 0$ denotes the iteration:

$$NT_u^{(k)}(\mathbf{x}, t) = \frac{\int_{B_{NT_u^{(k-1)}}((\mathbf{x},t),r)} D_{\mathbf{x}t} u((\mathbf{x}, t) + (\mathbf{y}, \tau)) \otimes D_{\mathbf{x}t} u((\mathbf{x}, t) + (\mathbf{y}, \tau)) \mu(\mathbf{y}, \tau) |G|^{1/2} \, d(\mathbf{y}, \tau)}{\text{Volume}\left(B_{NT_u^{(k-1)}}((\mathbf{x}, t), r)\right)} \tag{6}$$

and

$$B_{NT_u^{(k)}}((\mathbf{x}, t), r) = \begin{cases} \left\{(\mathbf{y}, \tau) : \langle G((\mathbf{x}, t)) \left((\mathbf{y}, \tau) - (\mathbf{x}, t)\right), \left((\mathbf{y}, \tau) - (\mathbf{x}, t)\right)\rangle \leq r^2\right\} & \text{when } k = 0 \\ \left\{(\mathbf{y}, \tau) : \langle NT_u^{(k)}(\mathbf{x}, t) \left((\mathbf{y}, \tau) - (\mathbf{x}, t)\right), \left((\mathbf{y}, \tau) - (\mathbf{x}, t)\right)\rangle \leq r^2\right\} & \text{when } k > 0 \end{cases} \tag{7}$$

where G is the initial metric defined in (3). The tensors are computed on the motion compensated neighborhood. Equations (6) and (7), constitute an iterative scheme that provide approximate affine covariant tensors together with their shape-adaptive neighborhoods at each point $(\mathbf{x}, t)$ in the video domain. To simplify the notation, we will denote by $T_u(\mathbf{x}, t)$ the structure tensor $NT_u^{(k)}(\mathbf{x}, t)$ for a fixed number of iterations k and a given value of r.

The algorithm to compute spatio-temporal tubes, each one labeled with a label L, is the following. For fixed parameters $\epsilon > 0$, $1 > \alpha \geq 0$ and $s > 0$, let $(\mathbf{x}_0, t_0)$ be a given seed point in the video domain. Then,
1. Set $(\mathbf{x}_1, t_1) = (\mathbf{x}_0, t_0)$.
2. If $(\mathbf{x}_1, t_1)$ has no label, label it with a *new* label L.
3. Consider all the 26-connected pixels of $(\mathbf{x}_1, t_1)$ **in the motion compensated neighborhood**. We denote by $U(\mathbf{x}_1, t_1)$ this neighborhood.
4. For each point $(\mathbf{x}_2, t_2) \in U(\mathbf{x}_1, t_1)$ which has no label, compute $\mathscr{D}\left(((\mathbf{x}_1, t_1), ((\mathbf{x}_2, t_2), s\right)$ given by (4) (see details below) and the Euclidean distance $\mathscr{D}_E\left(\mathbf{v}(\mathbf{x}_1, t_1), \mathbf{v}(\mathbf{x}_2, t_2)\right)$.
5. If $\widetilde{\mathscr{D}}\left((\mathbf{x}_1, t_1), (\mathbf{x}_2, t_2), s\right) < \epsilon$, then label $(\mathbf{x}_2, t_2)$ with the same label L as $(\mathbf{x}_1, t_1)$.
6. Set $(\mathbf{x}_1, t_1) = (\mathbf{x}_2, t_2)$ and iterate steps 2 to 6.

Figure 3 illustrate the main steps of the proposed algorithm. Finally, let us add some details on the computation of the comparison measure (4) in step 3. We only need the corresponding structure tensors,

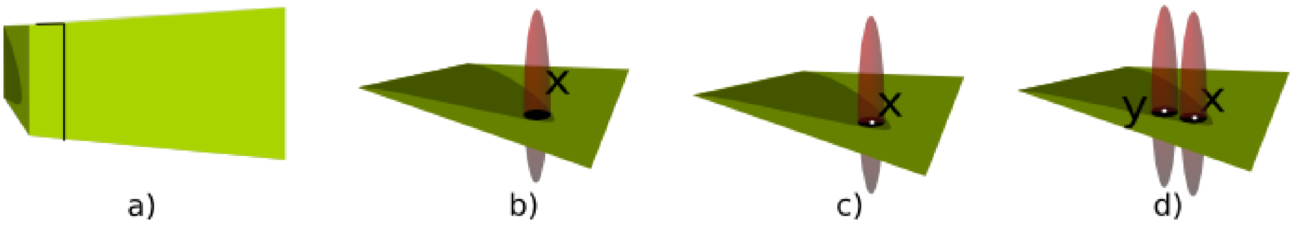

Figure 3: Diagram illustrating the main steps of the algorithm: a) video sequence; b) selected point $\mathbf{x} = (x_1, y_1, t_1)$, c) As $\mathbf{x}$ has no label, label it with a new label L = white; d) Compute the distance between $\mathbf{x}$ and $\mathbf{y} = (x_2, y_2, t_2) \in U(\mathbf{x})$; as they are similar, label $\mathbf{y}$ with same label as $\mathbf{x}$. These steps are performed iteratively.

$T_u(\mathbf{x}_1, t_1)$ and $T_u(\mathbf{x}_2, t_2)$, and the additional orthogonal transformation R which, as explained above, is split as $R = R_u^{-1}(\mathbf{x}_2, t_2) R_u(\mathbf{x}_1, t_1)$. $R_u(\mathbf{x}_1, t_1)$ and $R_u(\mathbf{x}_2, t_2)$ are computed by estimating dominant orientations of the gradient of u in the normalized patches of $(\mathbf{x}_1, t_1)$ and $(\mathbf{x}_2, t_2)$, respectively (i.e., the spheres obtained applying $T_u^{\frac{1}{2}}$) using weighted histograms of gradient orientations. This step is crucial since we align the dominant orientation of both patches to be able to compare them.

5 Experimental Results

Experiments on synthetics and real data sets are presented in this section. Each input sequence has its own resolution size and frame rate, which also helps us to test the robustness of the proposed approach based on video metrics and affine covariance properties to discrete videos. For each sequence, a tube corresponding to a fixed seed point $(\mathbf{x}, t)$ will be computed. Although the presented algorithm provides a supervoxel or tube representation of the full sequence, for simplicity and visualization purposes we will only present, for each sequence, the tube corresponding to a unique seed point $(\mathbf{x}_0, t_0)$.

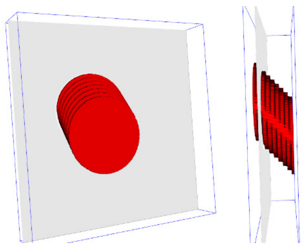

Figure 4: Tube computation from a point inside of the disc that moves, each frame, 10 pixels to the bottom.

5.1 Experiments using synthetic sequences

For the tests, synthetic sequences with a ground-truth optical flow have been created. We work with two types of synthetic sequences where a *disc* and a *deer* are the moving objects in the scene. To create consecutive frames successive translations and rotations are applied to the original frame. Figure 4 shows the tube resulting from a seed point inside of the disc which is translating downwards, while Figure 5 shows the tube of a point inside of the deer which is rotating. We can easily notice that the tube follows perfectly the trajectory of the disc and the deer, respectively, and fits its boundaries regardless of their long trajectories.

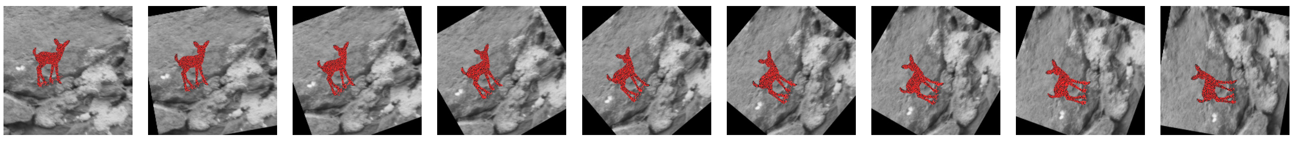

Figure 5: Tube computation starting from a point inside of a deer in a sequence obtained by rotating each frame an angle of 10 degrees counterclockwise.

5.2 Experiments using real sequences

In this section, we show the performance on our algorithm in well-known sequences. The ground-truth optical flow is not available, so an optical flow has been computed between each pair of frames using [Sánchez et al.,

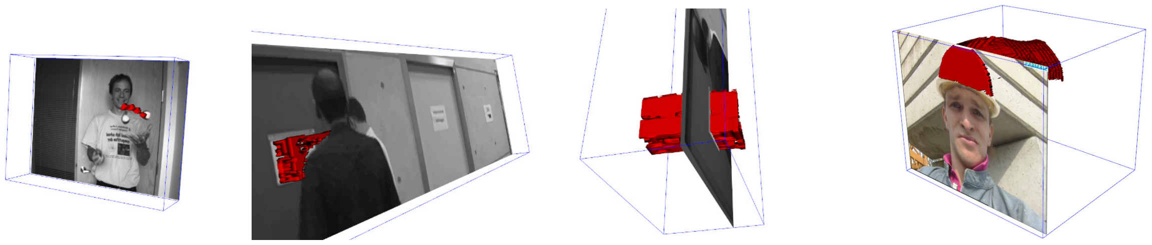
Figure 6: Spatio-temporal tube following a ball (left), the door sign (center) and the helmet (right).

2013]. Figure 6 shows, on the left, the tube corresponding to a seed point inside of the ball in the left hand of the *Beanbags Middlebury* sequence. Although the quality of the estimated optical flow is quite poor, our algorithm is able to follow the ball in the first four frames, losing the trajectory afterwards due to the disparity between the real motion and the optical flow values. In the second example, shown in the two middle images of Figure 6, the algorithm is able to separate the letters from the white background of the sign. In this sequence, the seed point is on the white part of the sign. Notice that is also able to adapt the tube once the sign is getting occluded by the man in the black t-shirt. The change in the shape or area of the tube from one frame to another, reflects the partial occlusions (when the area gets smaller) or disocclusion (when the area gets bigger). In other words, the proposed tubes can be used to estimate occlusion and disocclusions: The temporal boundaries of a tube give information about their birth and death. Lastly, in the right side of Figure 6, a tube computed from a seed point from the helmet of the *Foreman Middlebury* sequence is shown. In this sequence, the tube last along the frames, following the motion corresponding to the movement of the face and adapting the shape to the silhouette of the helmet.

6 Conclusion

We presented a video simplification approach which identifies moving or static objects in video and is able to group together all the pixels into a *tube* or *supervoxel*. Our method grounds on video metrics given by appropriate structure tensors and a video comparison measure. We have shown that, thanks to it, the tubes are recovered regardless of the perspective or affine transformation of the region over frames due to changes on point of view or object motion. We presented examples in synthetic and real sequences, demonstrating that the computed tubes adapt the shape to the boundaries of the region and follow the trajectory of the object.

Some of the problems of our algorithm is the sensitivity to similar color of neighboring regions, being sometimes a problem as it might lead a tube to grow outside the true region of the seed point. This could be solved via a variational optimization strategy building a functional measuring the quality of the decomposition of the original video into tubes with a fidelity data term based on our patch similarity measure over the whole video domain and a smoothness term measuring, for instance, the total area of the boundaries of the tubes. This solution would also achieve a complete partition of the video domain and thus a video segmentation solution.

Acknowledgments

All authors acknowledge partial support by MINECO/FEDER UE project, reference TIN2015-70410-C2-1-R and by GRC reference 2014 SGR 1301, Generalitat de Catalunya.

References

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

[Brox et al., 2006] Brox, T., Boomgaard, R., Lauze, F., Weijer, J., Weickert, J., Mrázek, P., and Kornprobst, P. (2006). Adaptive structure tensors and their applications. *Visual. and Proces. of Tensor Fields*, pages 17–47.

[Caelles et al., 2016] Caelles, S., Maninis, K.-K., Pont-Tuset, J., Leal-Taixé, L., Cremers, D., and Van Gool, L. (2016). One-shot video object segmentation. *arXiv preprint arXiv:1611.05198*.

[Calderero and Caselles, 2014] Calderero, F. and Caselles, V. (2014). Multiscale analysis of images on riemannian manifolds. *SIAM Journal Imaging Sciences*, 7(2):1108–1170.

[Cremers, 2007] Cremers, D. (2007). Bayesian approaches to motion-based image and video segmentation. In *Complex Motion*, pages 104–123. Springer.

[Fedorov et al., 2015] Fedorov, V., Arias, P., Sadek, R., Facciolo, G., and Ballester, C. (2015). Linear multiscale analysis of similarities between images on riemannian manifolds: Practical formula and affine covariant metrics. *SIAM Journal on Imaging Sciences*, 8(3):2021–2069.

[Grundmann et al., 2010] Grundmann, M., Kwatra, V., Han, M., and Essa, I. (2010). Efficient hierarchical graph-based video segmentation. In *CVPR 2010*, pages 2141–2148.

[Guichard, 1998] Guichard, F. (1998). A morphological, affine, and galilean invariant scale-space for movies. *Image Processing, IEEE Transactions on*, 7(3):444–456.

[Hare et al., 2016] Hare, S., Golodetz, S., Saffari, A., Vineet, V., Cheng, M.-M., Hicks, S. L., and Torr, P. H. (2016). Struck: Structured output tracking with kernels. *IEEE-PAMI*, 38(10):2096–2109.

[Jampani et al., 2016] Jampani, V., Gadde, R., and Gehler, P. V. (2016). Video propagation networks. *arXiv preprint arXiv:1612.05478*.

[Khoreva et al., 2016] Khoreva, A., Perazzi, F., Benenson, R., Schiele, B., and Sorkine-Hornung, A. (2016). Learning video object segmentation from static images. *arXiv preprint arXiv:1612.02646*.

[Kwak et al., 2015] Kwak, S., Cho, M., Laptev, I., Ponce, J., and Schmid, C. (2015). Unsupervised object discovery and tracking in video collections. In *ICCV*, pages 3173–3181.

[Lezama et al., 2011] Lezama, J., Alahari, K., Sivic, J., and Laptev, I. (2011). Track to the future: Spatio-temporal video segmentation with long-range motion cues. In *CVPR 2011*, pages 3369–3376.

[Nam and Han, 2016] Nam, H. and Han, B. (2016). Learning multi-domain convolutional neural networks for visual tracking. In *CVPR 2016*, pages 4293–4302.

[Oneata et al., 2014] Oneata, D., Revaud, J., Verbeek, J., and Schmid, C. (2014). Spatio-temporal object detection proposals. In *European conference on computer vision*, pages 737–752.

[Peyré, 2009] Peyré, G. (2009). Manifold models for signals and images. *Computer Vision and Image Understanding*, 113(2):249–260.

[Sánchez et al., 2013] Sánchez, J., Meinhardt-Llopis, E., and Facciolo, G. (2013). TV-L1 Optical Flow Estimation. *Image Processing On Line*, 3:137–150.

[Trichet and Nevatia, 2013] Trichet, R. and Nevatia, R. (2013). Video segmentation with spatio-temporal tubes. In *AVSS 2013*, pages 330–335.

[Weickert, 1998] Weickert, J. (1998). *Anisotropic diffusion in image processing*, volume 1. Teubner Stuttgart.

[Zhong et al., 2012] Zhong, W., Lu, H., and Yang, M.-H. (2012). Robust object tracking via sparsity-based collaborative model. In *CVPR, 2012*, pages 1838–1845.