Vasyl Mykhalchuk • Hyewon Seo • Frederic Cordier

AniM-DoG: A Spatio-Temporal Feature Point Detector for Animated Mesh

Abstract Although automatic feature detection has been a long-sought subject in computer graphics and computer vision, there has been little work done on animated meshes. In this paper, we present the first method for automatic detection of spatio-temporal feature points on animated meshes. We first define local deformation characteristics, based on strain and curvature values computed for each point at each frame. Next, we construct multi-resolution space-time Gaussians and difference-of-Gaussian (DoG) pyramids, where each level contains 3D smoothed and subsampled version of the previous level. Finally, we estimate locations and scales of spatio-temporal feature points by using a scale-normalized differential operator. A new, precise approximation of spatio-temporal scale-normalized Laplacian has been introduced, based on space-time Difference of Gaussian. We have experimentally verified our algorithm on a number of examples, and conclude that our technique allows us to detect spatio- and temporal- feature points in a stable and consistent manner.

Keywords Feature detection • Animated mesh • Multi-scale representation, Difference of Gaussian

1 Introduction

With the increasing advances in animation techniques and the capture devices, animation data has become more and more available today. Coupled with this, almost all geometry processing techniques (alignment, reconstruction, indexing, compression, segmentation, etc.) began to evolve towards the new, time-varying data, which constitute active research areas in computer graphics. Many applications such as medicine and engineering benefit from the increased availability and usability of animation data.

V. Mykhalchuk and H. Seo
ICube, University of Strasbourg
300 bd Sébastien Brant 67412 Illkirch, France
Tel.: +33-(0)3-6885-4554
Fax.: +33-(0)3-6885-4455
E-mail: {mykhalchuk,seo}@unistra.fr

F. Cordier
LMIA, University of Haute Alsace
4 rue des Frères Lumière 68093 Mulhouse, France
Tel.: +33-(0)3 8933-6017
Fax.: +33-(0)3-8933-6491
Email: frederic.cordier@uha.fr

The rapid growth in the size and the number of animation data suggests the need for maintaining efficiency in its representation and in the process applied to it. Our goal in this paper is to develop a spatio-temporal feature detection framework for the animated mesh (an ordered sequence of static mesh frames with fixed number of vertices and connectivity). Although feature detection on static mesh has a rich set of literature in the mesh processing [PKG03][LVJ05] [CCF*08][ZBV*09][DK12], there has been little work done on the feature detection in animated meshes.

In this paper, we introduce our algorithm AniM-DoG, which extracts spatio-temporal feature points on animated meshes. Encouraged by the success of scale space representation and Difference of Gaussian (DoG) in the salient point detectors on static meshes, we use the scale space and extend the concept of spatio-temporal DoG. The main contributions of this paper are:

1. Computation of deformation characteristic: Based on a deformation characteristic computed at each vertex in each frame, we build the scale space by computing various smoothed versions of the given animation data.

2. Space-time Difference of Gaussian (DoG) operator: At the heart of our algorithm is a new space-time Difference of Gaussian (DoG) operator, which is an approximation of the spatio-temporal, scale-normalized Laplacian. By computing the local extrema of the new operator in space-time and scale, we obtain repeatable sets of spatio-temporal feature points over different deforming surfaces modeled as triangle mesh animations. To the best of our knowledge, our work is the first that addresses the spatio-temporal feature detector in animated meshes.

We foresee the computation and use of spatio-temporal feature points as important, emerging areas of geometry processing, with applications like animation data retrieval and matching. As we extend existing geometry processing techniques to larger animation datasets, the need for robust, repeatable, and consistent detection of meaningful features from animation data will increase.

The remainder of the paper is organized as follows. In Section 2, we survey related works on local feature extraction in videos and (static) meshes. After recapitulating some basic terminologies and notions in Section 3, we present an overview of the method's pipeline overview in Section 4. Next, we describe the scale space representation and our AniM-DoG algorithm in Section 5.

2 Previous works

Feature extraction is essential in different domains of computer graphics and is frequently used for numerous tasks including registration, object query, object recognition etc. Scale-space representation has been widely used for feature extraction in image, video and triangle mesh data sets [Lin98]. However, almost no research has been done on the feature extraction of deforming surfaces, such as animated meshes.

Interest point detection in images and videos. Perhaps one of the most popular algorithms of feature extraction on images is Harris-Stephens detector [HS88], which uses second moment matrix and its eigen-values to choose points of interest. However, Harris method is not invariant to scale. Lindeberg [Lin98] tackled that problem and introduced automatic scale selection technique, which allows feature point detection at their characteristic scales. As Lindeberg has shown, local scale estimation using the normalized Laplace operator allows to robustly detect interest point of different extents. Mikolajczyk and Schmid [MS01] further developed Lindeberg's idea. As an improvement to the work of Lindeberg, authors proposed to use simultaneously Harris and Laplacian operators to detect interest points in scale-space representation of an image. First, feature point candidates are detected as local maxima of Harris function in the image plane. Further, in order to obtain a more compact representation, only those points are retained where Laplacian reaches maxima over scale space. This approach, however, requires dense sampling over the scale parameters and is therefore computationally expensive. As Lowe [Low04] proposed, Difference of Gaussians (DoG) is a good approximation of Laplacian and hence could be used to reduce computational complexity.

More recently, Laptev and co-authors [LL03] investigated how the notion of scale-space could be generalized to the detection of feature points in space-time data such as image sequences or videos. Interest points are identified as simultaneous maxima of the spatio-temporal Harris corner function as well as extrema of the normalized spatio-temporal Laplace operator. In order to avoid computational burden authors proposed to capture interest points in only sparse scale pyramid and then track these points in spatio-temporal scale-time-space towards the extrema of scale-normalized Laplacian. However, in their method there is no guarantee of convergence. In the work of [BET*08] a novel detector-descriptor scheme SURF (Speeded up robust features) has been proposed. Authors extend existing Hessian-based approaches and introduce Fast-Hessian detector that employs integral images for fast Hessian approximation.

FP(IP) detection on static meshes. There have been several approaches proposed for detecting feature points on 3D meshes. Most of them extend the detectors proposed for images. Pauly et al [PKG03] has used 'surface variation' to measure the saliency of vertices on the mesh, from which they build multi-scale representation. They extract points with high feature response values, which they connect to construct feature lines. Castellani et al [CCF*08] build scale-space over vertices in a mesh with successive decimations of the original shape. The displacements of a vertex throughout the decimation are used as a measure of saliency. Then vertices with high response in its DoG operator (inter-octave local maxima), and with high saliency in the neighborhood (intra-octave local maxima) are selected as feature points.

Zaharescu et al [ZBV*09] use photometric properties associated with each vertex as a scalar function defined on a 3D mesh. A discrete operator named as 'MeshDoG' is applied on this function, on which they apply Hessian operator to detect corner-like feature points. They extend MeshDOG to what they call MeshHOG, a feature descriptor, which essentially is a histogram of gradients in the neighborhood. The extracted features along with their descriptors were used for matching 3D model sequences they obtained from multi-view images. Sipiran and Bustos [SB10] have used 3D Harris operator which is essentially an extension of the Harris corner detector for images. After fitting quadratic patch to the neighborhood, a vertex is treated as an image, on which the Harris corner detector can be been applied. Darom and Keller [DK12] propose a scale-invariant local feature descriptor for the repeatable feature point extraction on 3D mesh. Each point is characterized by its coordinates, and a scale-space is built by successive smoothing of each vertex with its 1-ring neighbors. Local maxima both in scale and location are chosen as features.

All these methods, however, have been concerned with mesh data defined on the spatial domain only. In this work, we propose a new feature detection technique in animated meshes which extends existing methods based on linear scale-space theory to spatio-temporal domain.

3 Preliminaries

At the heart of our algorithm is a scale-space representation. In this section we briefly recapitulate some basic notions that have been previous studied. Later, we develop its extensions to animated mesh data, which are described in section 5.2 and section 5.3.

Scale-space representations have been studied extensively in feature detection for images, and more recently, for videos. The basic idea is to represent an original image $f : R^d \to R$ at different scales as $L: R^d \times R_+ \to R$ by convolution of f with a Gaussian kernel with variance σ:

$$L(x;\sigma) = G(x;\sigma) * f(x), \qquad (1)$$

where

$$G(x;\sigma) = \frac{1}{\left(\sqrt{2\pi}\sigma\right)^d} \exp\left(-\frac{x_1^2 + \cdots x_d^2}{2\sigma^2}\right). \qquad (2)$$

One of the most successful feature detectors is based on DoG (Difference of Gaussians). To efficiently detect feature points in scale space, Lowe [Low04] proposed using convolution of the input image with the DoG functions. It is computed from the difference of two nearby scales:

$$D(x;\sigma) = \big(G(x;\, k\sigma) - G(x;\, \sigma)\big) * f(x),$$

$$= L(x;k\sigma) - L(x;\sigma). \quad (3)$$

where k is a constant multiplicative factor separating the two nearby scales. Note that DoG is particularly efficient to compute, as the smoothed images L need to be computed in any case for the scale space feature description, and D can therefore be computed simply by image subtraction.

The DoG provides a close approximation to the scale-normalized Laplacian of Gaussian [Lin94*], $\sigma^2\nabla^2 G$, which has been proven to produce the most stable, scale-invariant image features [MS02*]. The DoG and scale-normalized LoG are related through the heat-diffusion equation:

$$\frac{\partial G(x)}{\partial \sigma} = \sigma\nabla^2 G(x), \quad (4)$$

where the Laplacian on the right side is taken only with respect to the **x** variables. From this, we see that $\nabla^2 G(\mathbf{x})$ can be computed from the finite difference approximation to $\partial G(\mathbf{x})/\partial\sigma$, using the difference of nearby scales at $k\sigma$ and σ:

$$\frac{\partial G(x)}{\partial \sigma} = \lim_{k\to 1} \frac{G(x;k\sigma) - G(x;\sigma)}{k\sigma - \sigma} = \sigma\cdot\nabla^2 G(x), \quad (5)$$

and therefore,

$$G(x;k\sigma) - G(x;\sigma) \approx (k-1)\cdot\sigma^2\cdot\nabla^2 G. \quad (6)$$

4 Overview

Our goal is to develop a feature detector on animated mesh based on space-time DoG, which has been reported to be efficient approximation of robust Laplacian blob detector in space domain. Note that animated meshes that we are dealing with are assumed to have no clutters or holes, and maintain fixed topology over time, without tearing or changing genus. The spatial samplings can vary from one mesh to another, but it is desirable to have uniform sampling across one surface. The temporal sampling rate can also vary (~30Hz in our experiments), depending on how the animation has been obtained. In any case, the temporal sampling is considered uniform.

The features we want to extract are the corners/blob-like structures, which are located in regions that exhibit a high variation of deformation spatially and temporally. We first define local deformation attributes on the animated mesh, from which we build a multi-scale representation of it. One of the main motivations to base our method on local surface deformation rather than vertex trajectories can be explained by the fact that (1) local deformation on a surface can be effectively measured by some well-defined principles, and that (2) the domain has intrinsic dimension of 2D+time (rather than 3D+time) with some reasonable assumption on the data, i.e. differentiable 2-manifold with time-varying embedding.

We then compute the deformation characteristics at different scales, by defining an appropriate spatio-temporal Gaussian-like smoothing method. However, real Gaussian smoothing on mesh animation is problematic and expensive. Therefore we follow the other alternative and approximate Gaussian low-pass filter by a sequence of spatio-temporal box average filters of fixed width. We obtain different space and time scales of deformation field over animation by varying the number box filtering is applied in space and in time.

To estimate positions and scales of mesh animation feature points, we define a scale-normalized differential operator that assumes simultaneous extrema over space-time and scale neighborhood. Theoretically, it is possible to compute spatio-temporal, scale-normalized Laplacian on every vertex of the animated mesh. For example, one could extend the work by Zaharescu et al and compute the 3D gradient and Laplacian on the animated mesh. However, it would be too much costly as it requires computing the normal plane, on which principal direction should be determined. Therefore, we introduce a new precise approximation of spatio-temporal scale-normalized Laplacian based on space-time Difference of Gaussian. Then local extrema of the space-time DoG operator are captured as feature points. Space-time DoG operator is cheap to compute and allows to robustly detect feature points over mesh animation in a repetitive and consistent manner.

5 Dynamic feature detector (AniM-DoG)

5.1 Deformation characteristics definition

We are interested in quantities that are related to local deformation characteristics associated to each point of the mesh, at each frame. In our work, we base our algorithm on locally computed strain and curvature values computed as follows.

Strain computation. Let $\mathcal{M}$ with *M* frames and *N* vertices be an input sequence of deforming mesh. The mesh at its first frame $\mathcal{M}^0$ is assumed to be a reference, i.e. the rest shape of the mesh before the deformation, which is usually the case. If this is not the case, we manually insert a rest pose at the first frame. We first analyze the deformations associated with each triangle t_i^f for respective frame *f*, analogously to [LCS14], by estimating the deformation gradient tensor F_i^f. With F_i^f being the affine transformation that maps t_i^0 to t_i^f, we obtain principal stretches by the eigen-analysis of $C_i^f = \left(F_i^f\right)^T \left(F_i^f\right)$ and use the largest eigenvalue λ_1 (maximum principal strain) as the in-plane deformation of the triangle. Once the per-triangle deformations have been computed, we estimate the deformation for each vertex by taking the average deformation value of its adjacent triangles.

Curvature computation. Computing the curvature at the vertices of a mesh is known to be non-trivial because of the piecewise-linear nature of meshes. One simple way of computing the curvature would be to compute the angle between two neighboring triangles along an edge. However, such curvature measurement is too sensitive to the noise on

the surface of the mesh because its computation relies on two triangles only. Instead, we compute the curvature over a set of edges as described in [ACD*03]. Given a vertex v_i, we first compute the set of edges E_i whose two vertices are within a user-defined geodesic distance to v_i. Next we compute the curvature at each of the edges of E_i. The curvature at v_i is then calculated as the average of the curvatures at the edges of E_i.

Deformation measure. For each vertex $v_i^f \in \mathcal{M}$ $(f = 1,\dots,M, i = 1,\dots,V)$ on which we have computed strain $s(v_i^f)$ and curvature $c(v_i^f)$, we define the deformation characteristics $d(v_i^f)$ as follows:

$$d(v_i^f) = s(v_i^f) + \alpha \cdot |c(v_i^f) - c(v_i^1)|.$$

The first term is obtained by transferring the above described per-triangle strain values to per-vertex ones, computed at each frame. At each vertex, we take the average strain values of its adjacent triangles as its strain.

The second term encodes the curvature change with respect to the initial, reference frame. Note that $d(v_i^f) \geq 0$ for $\forall v_i^f$, which we use later for the feature detection (section 5.3). We set α typically to 7 in our experiments. Color coded deformation characteristics on a bending cylinder data is shown in Fig. 1.

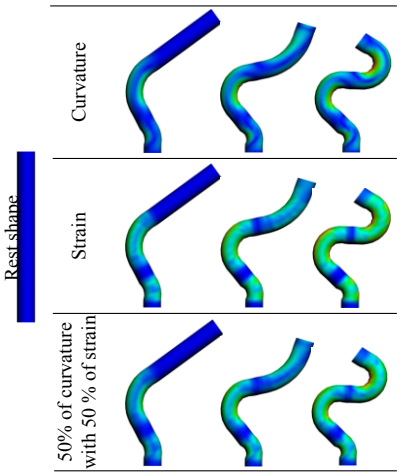


Fig. 1 Local deformation characteristics shown on a bending cylinder mesh.

5.2 Scale space construction (Multi-scale representation)

Given deformation measures d for all vertices of animated mesh $\mathcal{M}$, we re-compute d at K·L different scale representations, obtaining octaves O_{kl} ($k \in \Sigma$=0,…K, $l \in$ T=0,…L) of deformation characteristics at different spatio-temporal smoothing resolutions.

Theoretically, the smoothed versions are obtained by applying an approximated Gaussian filter for meshes. In practice, the approximation consists of subsequent convolutions of the given mesh with a box (average) filter [DK12]. In our work, we define a spatio-temporal average filter on the deformation characteristics of the animated mesh and compute a set of filtered deformation scalar fields, which we call as anim-octaves. As shown in Fig. 2, we define spatio-temporal neighborhood $\mathcal{N}_{st}$ of a vertex in animation as a union of its spacial and temporal neighborhoods. A spatio-temporal average smoothing over $\mathcal{N}_{st}$ is obtained by applying a local spatial filter followed by a local temporal one.

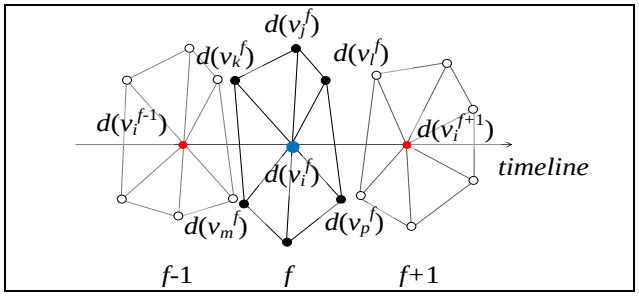


Fig. 2 The smallest possible spatio-temporal neighborhood $\mathcal{N}_{st}$ of a vertex v_i^f (blue dot) is composed of 1-ring spatial neighbors in frame f (black vertices) and 1-ring temporal neighbors (red vertices). Note that considering the temporal neighbors implies considering their spatial neighbors (white vertices) as well.

More specifically, for each vertex v_i^f at an anim-octave of scale (σ_k, τ_l), we compute deformation measures at next spatial octave (σ_{k+1}, τ_l) by averaging deformation measurements in current vertex of current octave $d(v_i^f, \sigma_k, \tau_l)$ and its 1-ring's spatial neighborhood $d(\mathcal{N}_s^1(v_i^f), \sigma_k, \tau_l)$ i.e. at adjacent vertices. For the next temporal octave (σ_k, τ_{l+1}) we repeat similar procedure but this time averaging deformation values in 1-ring temporal neighborhood $\mathcal{N}_t^1(v_i^f)$ as in Fig.2. And for the next spatio-temporal octave, we start from deformations in octave (σ_{k+1}, τ_l) and apply temporal average filter again in the way described above, which yields $d(v_i^f, \sigma_{k+1}, \tau_{l+1})$. We continue this procedure until we build the desired number of spatio-temporal octaves. Fig.3 illustrates our anim-octaves structure. We denote an anim-octave as $O_{kl} = d(\mathcal{M}, \sigma_k, \tau_l)$, where $O_{00} = d(\mathcal{M})$. (We note that although the term octave is widely used to refer to a discrete interval in the scale space, it may be misleading since in a strict sense, our octaves do not represent the interval of half or double the frequency.) In Fig.4, we illustrate the multi-scale deformation characteristics we computed on an animated mesh. The horizontal axis represents the spatial scale σ_k, and the vertical axis the temporal scale τ_l.

octave scale	σ_1	σ_2	…	σ_k
τ_1	O_{11}	O_{12}	…	O_{1k}
τ_2	O_{21}	O_{22}	…	O_{2k}
⋮	⋮	⋮	⋱	⋮
τ_l	O_{l1}	O_{l2}	…	O_{lk}

Fig. 3 Scale space is built by computing a set of octaves of the input animated mesh.

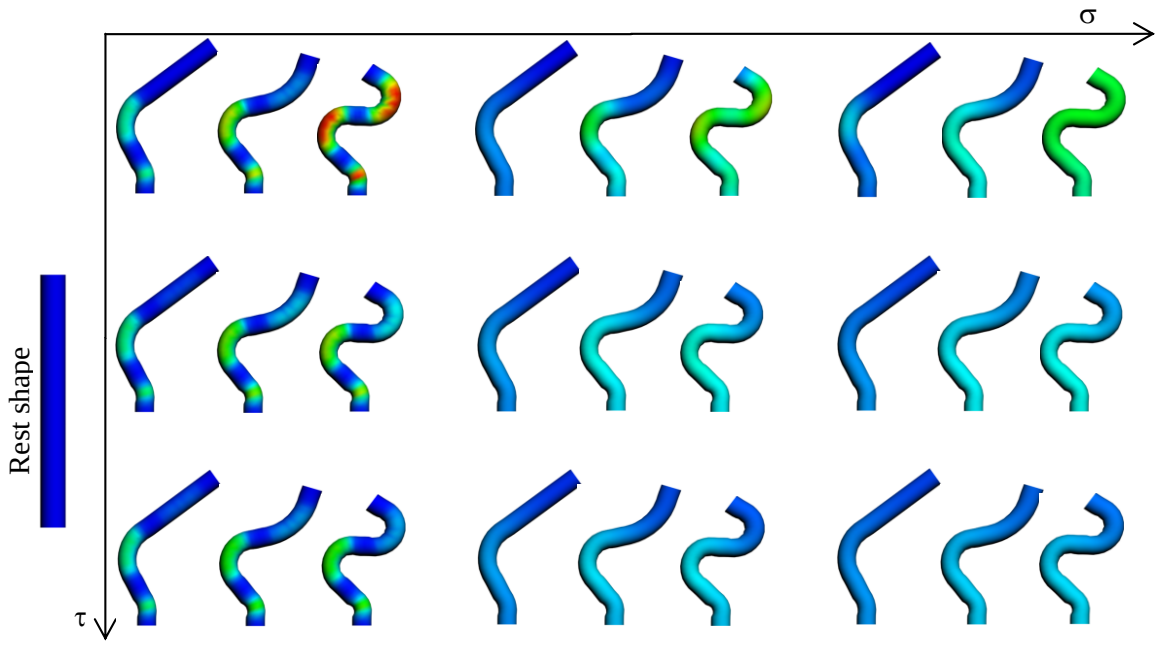


Fig. 4 Multi-scale deformation characteristics on an animated mesh. From left to right, spatial scale σ_j increases, and from top to bottom, temporal scale τ_i increases.

Widths of the average filters. We set the width of the spatial filter as the average edge length of the mesh taken at the initial frame, assuming that spatial sampling of the mesh is moderately regular, and that the edge lengths in the initial frame represent well those in other frames of animation. Note that it can be done in a per-vertex manner, by computing for each vertex the average distance to its 1-ring neighbors, as it has been proposed by Darom and Keller [DK12]. However, since this will drastically increase the computation time for the octave construction stage, we have chosen to use the same filter width for all vertices.

Determining the width of the temporal filter is simpler than the spatial one, as almost all data have regular temporal sampling rate (fps) throughout the duration of animation. Similarly to the spatial case, the inter-frame time is used to set the width of the temporal filter. Instead of averaging over immediate neighbors, however, we consider larger number of frame neighbors, in most cases. This is especially true when the animated mesh is densely sampled in time. The filter widths we used for each dataset are summarized in Table 1.

Maximum number of smoothing. Since an animated mesh can be highly redundant and heavy in size, the memory space occupied by the anim-octaves can be large as the number of scales increases. This becomes problematic in practice. With an insufficient number of smoothing, on the other hand, features of large characteristic scale will not be detected. (Indeed, when the variance of the Gaussian filter is not sufficiently large, only boundary features will be extracted.) Fig. 5 illustrates the principle behind the characteristic scale and the maximum required scale level. Given a spatio-temporal location on the mesh, we can evaluate the DoG response function and plot the resulting value as a function of the scale (number of smoothing). Here, the spatial scale has been chosen as a parameter for the simplicity. The characteristic scales of the chosen vertices are shown as vertical lines, which can be determined by searching for scale-space extrema of the response function. To detect the feature points on the bending region in the middle (v_1), for instance, the octaves should be built up to 12 level. This means the maximum number of smoothing must be carefully set in order to be able to extract feature points of all scales while maintaining moderate number of maximum smoothing.

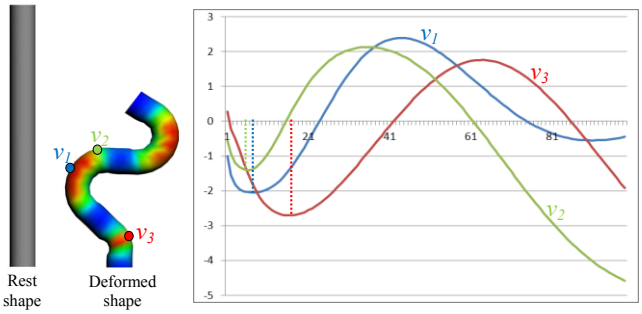


Fig. 5 The DoG response function has been evaluated as a function of the spatial scale (number of smoothings). The characteristic scales of the chosen vertices are shown as vertical lines.

In order to make sure that features representing blobs of large scale are detected, we start by an average filter. Multiple applications of a box (average) filter approximates Gaussian filter [AC92]. More precisely, *n* applications of a box filter of width *w* produce overall filtering effect equivalent to the Gaussian filter with a standard deviation of:

$$\sigma = \sqrt{\frac{n(w^2-1)}{12}}. \qquad (8)$$

When the Laplacian of Gaussian is used for detecting blob centers (rather than boundaries), the Laplacian achieves a maximum response with

$$\sigma = \frac{r}{\sqrt{2}}, \qquad (9)$$

where *r* is the radius of the blob we want to detect.

Now assuming that the maximum radius r_{max} of the blob we want to detect is known, we can compute the required number of average smoothing that is sufficient to detect blob centers from Eq. (8) and Eq. (9):

$$r_{max}^2 = \frac{n(w^2-1)}{6} \quad (10)$$

$$\Leftrightarrow n = \frac{6r^2}{w^2-1}. \quad (11)$$

The maximum number of application of box filter for each dataset is listed in Table 1.

Maximum radius of all possible blobs. Along the spatial scale space, we consider the average edge length of the initial shape as the width of the average filter w, as described above. For the maximum possible radius of a blob, we compare the axis-length change of the tight bounding box of the mesh during animation, with respect to its initial shape. The half of the largest change in axis-length is taken as r_{max}.

Along the temporal scale space, we assume that the maximum radius r_{max} of all possible blobs is the half of the total time of duration of animation. By fixing the maximum number of smoothing to some moderate value, we obtain the desirable box filter width from Eq. (10) or Eq. (11).

5.3 FP detection by DoG

In this section, we extend the idea of scale representation in spatial domain to spatio-temporal domain and adopt it to the case of animated mesh. Next, we propose our feature point detector and discuss some of its implementation related issues.

Spatio-temporal scale space principles. Given time varying input signal $f(\mathbf{x},t)$, $f: \mathbb{R}^d \times \mathbb{R} \to \mathbb{R}$, one could build its scale-space representation $L(\mathbf{x}, t;\ \sigma, \tau)$ by convoluting f with anisotropic Gaussian

$$L(\mathbf{x}, t; \sigma, \tau): \mathbb{R}^d \times \mathbb{R} \times \mathbb{R}_+^2 \to \mathbb{R}.$$

The motivation behind the separate scale parameters in space σ and time τ is that space and time extents of feature points are independent in general [LL03]. On this representation, [SRV98] shows that the spatio-temporal scale space $L(\mathbf{x}, t;\ \sigma, \tau)$ can be defined as a solution of two diffusion equations:

$$\frac{\partial L}{\partial \sigma} = \sum_i \frac{\partial^2 L}{\partial x^i \partial x^i}, \quad (12a)$$

$$\frac{\partial L}{\partial \tau} = \frac{\partial^2 L}{\partial^2 t}, \quad (12b)$$

with the initial condition:

$$\lim_{\sigma \to 0^+ \tau \to 0^+} L(x, t; \sigma, \tau) = f(x, t).$$

In our case, mesh animation $\mathcal{M}$ could be interpreted as 2-manifold with time-varying embedding m(u, v, t) in 3D Euclidean space. Measuring deformation scalar field $d(\mathcal{M})$ in 2-manifold over space and time yields a 3d input signal of the form $d(u, v, t)$, $d: \mathbb{R}^2 \times \mathbb{R} \to \mathbb{R}$, and its scale space form $L_d(u, v, t;\ \sigma, \tau)$: $\mathbb{R}^2 \times \mathbb{R} \times \mathbb{R}_+^2 \to \mathbb{R}$.

Computation of DoG pyramid. Given a scale space representation $L_d(\mathbf{x}, t;\ \sigma, \tau)$ of surface deformation in animation we now proceed with construction of DoG feature response pyramid. To achieve the invariance in both space and time, we first compute a spatio-temporal DoG pyramid, from which features points will be extracted. From the heat-diffusion equations (12a-12b) and equation (6), we see that

$$D(x; \sigma) = L(x; k\sigma) - L(x; \sigma) = \sum_i \frac{\partial^2 L}{\partial x^i \partial x^i}, \quad (13a)$$

$$D(t; \tau) = L(t; k\tau) - L(t; \tau) = \frac{\partial^2 L}{\partial^2 t}. \quad (13b)$$

Hence spatio-temporal Laplacian can be approximated by summing up (13a) and (13b):

$$D(x; \sigma) + D(t; \tau) = \sum_i \frac{\partial^2 L}{\partial x^i \partial x^i} + \frac{\partial^2 L}{\partial^2 t} = \nabla^2 L. \quad (14)$$

Intuitively, Eq.(14) implies that spatio-temporal Laplacian operator can be approximated by the sum of DoGs in space and time scale. However, it is not scale normalized. Laptev and Lindeberg [LL03] has shown that scale normalized Laplacian takes form of

$$\nabla_{norm}^2 L = \sigma^2 \tau^{1/2} \sum_i \frac{\partial^2 L}{\partial x^i \partial x^i} + \sigma \tau^{3/2} \frac{\partial^2 L}{\partial^2 t}. \quad (15)$$

Thus, we achieve the scale-normalized approximation of Laplacian through DoG by multiplying both sides of (13a) with $\sigma^2 \tau^{1/2}$ and (13b) with $\sigma \tau^{3/2}$, obtaining:

$$\sigma^2 \tau^{1/2} D(x; \sigma) = \sigma^2 \tau^{1/2} \sum_i \frac{\partial^2 L}{\partial x^i \partial x^i}, \quad (16a)$$

$$\sigma \tau^{3/2} D(t; \tau) = \sigma \tau^{3/2} \frac{\partial^2 L}{\partial^2 t}. \quad (16b)$$

And from (16a-16b) we see that

$$\nabla_{norm}^2 L = \sigma^2 \tau^{1/2} D(x; \sigma) + \sigma \tau^{3/2} D(t; \tau).$$

On the other hand, we can get a formulation of spatio-temporal Difference of Gaussians which approximates scale-normalized Laplacian

$$D_{st}(x, t; \sigma, \tau) = \sigma^2 \tau^{1/2} D(x; \sigma) + \sigma \tau^{3/2} D(t; \tau).$$

Thus, given the definition of spatio-temporal Difference of Gaussians we can compute feature response pyramid in the following way. For each vertex (u,v,t) in the animated mesh $\mathcal{M}$, and for every scale $(\sigma_k, \tau_l) \in \Sigma \times T$ of the surface deformation pyramid we compute $D_{st}(u, v, t; \sigma_k, \tau_l)$.

FP detection. Once the spatio-temporal DoG pyramid $\{D_{st}(u, v, t; \sigma_k, \tau_l) |\ (\sigma_k, \tau_l) \in \Sigma \times T\}$ is constructed, we extract feature points by identifying local extrema of the adjacent regions in the space, time, and scales. The spatial scale corresponds to the size of the region where some distinctive deformation is exhibited. Similarly, the temporal

scale corresponds to the duration (or speed) of the deformation in which the deformation takes place.

Our interest point detection algorithm proceeds in a way conceptually similar to Mikolajczyk and Schmid [MS01], but with differently constructed pyramid and feature response. Note that our approach requires only DoG, and yet works well and efficiently for identifying blob-like structures. Since our surface deformation function is always non-negative (and consequently its scale-space representation), Laplacian of Gaussian and its DoG approximation reach local minima at centers of blobs, as illustrated in Fig. 6(a). For each scale $(\sigma_k, \tau_l) \in \Sigma \times \mathrm{T}$ of 2D DoG pyramid, we first detect vertices in animation that are local minima in DoG response over their spatio-temporal neighborhood $\mathcal{N}_{st}$:

$$P_{kl} = \{p \in \mathcal{M} |\ \forall p_i \in \mathcal{N}_{st}(p), D_{kl}(p) < D_{kl}(p_k) \text{ and } D_{kl}(p) < \varepsilon_{st}\},$$

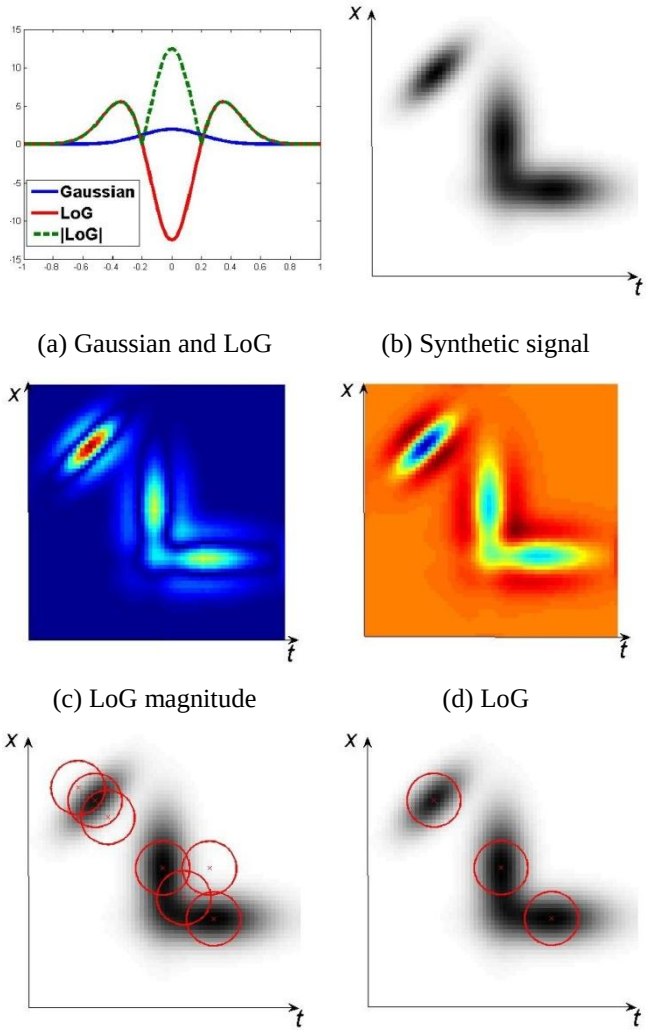
where $\mathcal{N}_{st}(p)$ is a spatio-temporal neighborhood of vertex p in the animation $\mathcal{M}$ (Fig. 2). Then, out of preselected feature candidates P_{kl}, we retain only those vertices which are simultaneous minima over neighboring spatio-temporal scales of DoG pyramid:

$$P = \{p \in P_{kl} | \forall (i,j) \in \mathcal{N}_{\sigma\tau}(k,\mathrm{l}), D_{ij}(p) > D_{kl}(p) \text{ and } D_{kl}(p) < \varepsilon_{\sigma\tau}\},$$

where $\mathcal{N}_{\sigma\tau}(k,\mathrm{l})$ is a set of 8 neighboring scales $\mathcal{N}_{\sigma\tau}(k,\mathrm{l}) = \{D_{(k+1)l}, D_{(k+1)(l-1)}, D_{(k+1)(l+1)}, D_{k(l-1)}, D_{k(l+1)}, D_{(k-1)l}, D_{(k-1)(l-1)}, D_{(k-1)(l+1)}\}$, and ε_{st}, $\varepsilon_{\sigma\tau}$ are user-controlled thresholds.

Dealing with secondary (border) blobs. However, in case we consider local maxima of DoG (LoG) magnitude, we may detect artifacts. Undesirable secondary blobs are caused by shape of Laplacian of Gaussian which yields peaks around the border of real blob (Fig. 6(a)). Consider a perfect Gaussian blob as an input signal. If we assume magnitude (i.e. absolute value) of LoG to be feature response, we get strong peak in the center of the blob and two other secondary peaks around edges, and that is troublesome. In contrast, dealing with signed LoG (not absolute) we observe valleys (local minima) in blob centers and peaks (local maxima) on borders. Hence searching for local minima of LoG, rather than local maxima of LoG magnitude, prevents from detection of false secondary features (Fig. 6(b-f)). The other way around could be to use LoG magnitude but discard local maxima which are not strong enough in initial signal, and therefore are false findings. Though, previous works in feature extraction on images/video/static meshes [MS01, LL03, ZBV*09] often adopt Hessian detector, which does not detect secondary blobs. However, in contrast to DoG detector, estimation of Hessian on a mesh surface is significantly heavier. And even more problematic and challenging to estimate Hessian matrix on animated mesh.

Implementation notes. Often, animated meshes are rather heavy data. As we increase the number of anim-octaves in the pyramid, we can easily run out of memory, since each octave is essentially a full animation itself but at different scale. Consequently, we have to address that issue in the implementation stage. In order to minimize memory footprint, we compute pyramids and detect feature points progressively. We fully load into main memory only space scale dimension of Gaussian and DoG pyramids. As for time scale, we keep only two neighboring time octaves simultaneously, which are required for DoG computation. Then we construct the pyramid from bottom to top by iteratively increasing time scale. On each iteration of Gaussian/DoG pyramid movement along time scale, we apply our feature detection method to capture interest points (if any on current layer). We repeat the procedure until all pyramid scales have been processed.



(a) Gaussian and LoG (b) Synthetic signal

(c) LoG magnitude (d) LoG

(e) Features as maxima of LoG magnitude response (f) Features as minima of LoG magnitude response

Fig. 6. A 2D illustration of our feature detection method. (a) LoG yields valley in blob's center and peaks around the boundary, while magnitude of LoG has peaks in both case. (b) Synthetic input signal consisting of 3 Gaussian blobs in 2d. (c) Response of synthetic 2d signal as absolute value of LoG. (d) Response of the 2d signal computed as LoG. (e) From the LoG magnitude response, we observe several false secondary blobs. (f) Features captured as local minima of LoG response are reliable.

6 Experiments

Deforming meshes used in our experiments include both synthetic animations and motion capture sequences, which are summarized in Table 1. We synthesized a simple deforming Cylinder animation by rigging and used it for initial tests. We also captured two persons' facial expressions using an optical motion capture system [Vicon], and transferred them to their scanned faces. Face1Happy, Face1Surprise, Face2Happy, Face2Surprise contain facial expressions of happiness and surprise of those scanned subjects. Finally, GallopingHorse and GallopingCamel were obtained from results of deformation transfer work [SP04] that are available online [Mesh].

Table 1 The animated meshes used in our experiments

Name	No.vertices/ triangles	No. frames	Filter widths (space/time)	Max. no. smoothings (space/time)
Cylinder	587/1170	40	10.0/0.83	50/100
Face1Happy	608/1174	139	8.96/8.45	118/113
Face1Surprise	608/1174	169	9.39/13.2	96/107
Face2Happy	662/1272	159	9.31/13.2	112/94
Face2Surprise	662/1272	99	8.95/8.45	109/57
GallopingHorse	5000/9984	48	3.48/5.33	77/54
GallopingCamel	4999/10000	48	2.62/5.33	102/54

Fig. 8 shows selected frames of several animated meshes we used in our experiments. Spatio-temporal feature points we have extracted using our algorithm are illustrated as spheres. For the complete sequences along with the extracted feature points, please take a look at our accompanying demo video. The color of a sphere represents the temporal scale (red color corresponds to more fast deformations) of the feature point, and radius of sphere indicates the spatial scale. Vertex color on surfaces corresponds to amount of deformation (strain and curvature change) observed in each of animation frame. During experiments we have discovered that our method captures spatio-temporal scales in a robust manner. For example, surface patches around joints of cylinder (Fig.8:1a-1e) exhibit different amount of deformation that occurs at different speed. The top joint is moving fast and consequently corresponding feature was detected at low temporal scale (red color). However, the mid-joint is deforming for a long time and we identify it at high temporal scale (blue color). Moreover large radii of deforming spheres for both joints make sense and indicate large deforming regions around the features, rather than very local deformation (Fig.8:1c). Second row in (Fig.8: 2a-2e) depicts some of the feature points in Horse mesh animation, and third row (Fig.8:3a-3e) corresponds to Camel animation. Those two meshes deform in a coherent manner [SP04], and eventually we detect their spatio-temporal features quite consistently. In the last two rows (Fig.8:4a-4e, 5a-5e) we present feature points in mocap-driven face animations of two different subjects. Our subjects were asked to mimic of slightly exaggerated emotions during the mocap session. Notice that people normally use different set of muscles when they show up facial expressions, and therefore naturally we observe some variations in the way their skin deforms.

Our algorithm is implemented in C++. All our tests have been conducted on an Intel Core i7–2600 3.4 GHz machine, with 8GB of RAM. The computation time devoted to full pipeline of the algorithm is approximately 2 minutes for most of our example data.

Invariance to rotation and scale. Invariance of our detector to rotation as well as scale is evident from the definition of our deformation characteristics. Both the strain and the curvature measure we use are invariant to rotation and scale of the animated mesh.

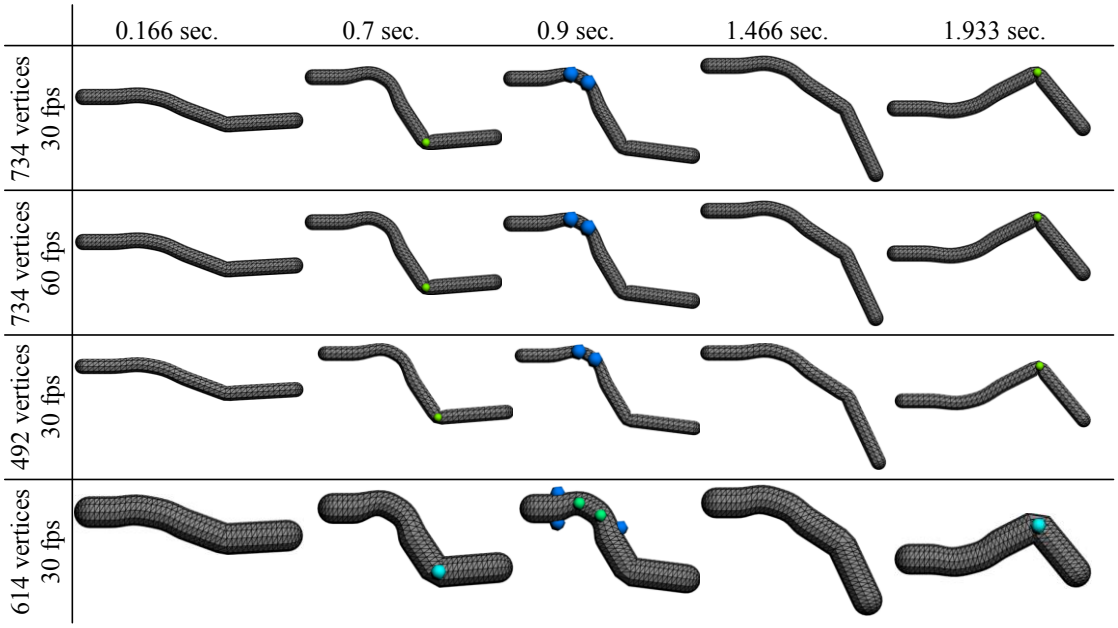


Fig. 7 Results we obtained from on varying datasets of bending cylinder animation showing consistent behavior of our feature detector.

Robustness to changes in spatial and temporal sampling. Robustness of our feature detector to changes in spatial sampling is obtained by the adaptive setting of the widths of the box filters. As described in Section 5.2, we set the width of the spatial filter as the average edge length of the mesh taken at the initial frame. In order to demonstrate the invariance to spatial density of the input mesh, we have conducted comparative experiments on two bending cylinders. These two cylinders have identical shape and deformation; they only differ by the number of vertices and the inter-vertex distance. As shown in the 1st and 3rd rows of Fig. 7, the features are extracted at the same spatio-temporal locations. Robustness to changes in temporal sampling is obtained similarly to the above, i.e. by the adaptive setting of the widths of the box filters. Similar experiments have been conducted by using the two bending cylinders as shown in the 1st and 2nd rows of Fig 7. They are perfectly identical except that the temporal sampling of the first one is twice higher than that of the first one. Once again, the extracted feature points are identical in their locations in space and time.

We have further experimented with datasets of similar animations, but with different shape, spatial- and temporal-samplings (The 4th row of Fig.7, galloping animals and two face models in Fig. 8). Although the extracted features show a good level consistency, they are not always identical. For example, feature points for the galloping horse and camel do not have the same properties (location, time, tau and sigma). Similar results have been observed for the "face" models. This can be explained by the following facts: Firstly, although the two meshes have deformations that are semantically identical, the level of deformation (curvature and strain) might differ greatly. Secondly, most of these models have irregular vertex sampling whereas in our computation of the spatial filter width, we assume that the vertex sampling is regular.

7 Conclusion

We have presented a new feature detection technique on triangle mesh animations based on linear scale-space theory. We introduced a new spatio-temporal scale representation of surface deformation in mesh animations. Furthermore, we developed extension of classical DoG filter to spatio-temporal case. The later allows our method to robustly extract repeatable sets of feature points over different deforming surfaces modeled as triangle mesh animations. We carried out experimental validation of detected features on various types of data sets and observed consistent results. Our approach has shown robustness to spatial and temporal sampling of mesh animation. In our future research, we intend to focus on feature point descriptor that could be useful for applications such as matching between animations.

Descriptors. Our feature detector could be extended to detector-descriptor. One straightforward idea of the feature point descriptor could be the following. Given space –time neighborhood around feature point v_i^f which consists of its *k*-ring space neighborhood over range of [*f*-*l*, …, *f*+*l*] frames. Note, that values of *k* and *l* could be adjusted to reflect characteristic scale of the feature. Next, flattening of *k*-ring regions for each frame of the range produces a volumetric stack of planar mesh patches. Then, proceeding in a spirit of robust 3D SIFT descriptor [SAS07], we can estimate histograms of DoG gradients computed inside the spatio-temporal volume around the feature point. Further, Euclidean or Earth Mover's distance between the histograms could be used for measuring similarities of features within mesh animation or between different animations.

Acknowledgments

We are grateful to the MIRALab, University of Geneva, who has provided us the face scans. This work has been supported by the French national project SHARED (Shape Analysis and Registration of People Using Dynamic Data, No.10-CHEX-014-01).

References

[AC92] R. Andonie, E. Carai: Gaussian Smoothing by Optimal Iterated Uniform Convolutions, Computers and Artificial Intelligence, 11(4): pp. 363-373, 1992.

[ACD*03] P. Alliez, D. Cohen-Steiner, O. Devillers, B. Levy, and M. Desbrun: Anisotropic Polygonal Remeshing, ACM Transactions on Graphics, 2003.

[BET*08] H. Bay, A. Ess, T. Tuytelaars, L. van Gool: Speeded-up Robust Features (SURF), Computer Vision and Image Understanding (CVIU), Vol. 110, No. 3, pp. 346–359, 2008.

[CCF*08] U. Castellani, M. Cristani, S. Fantoni, V. Murino: Sparse points matching by combining 3D mesh saliency with statistical descriptors. Comput. Graph. Forum 27(2): 643–652 (2008).

[DK12] T. Darom, Y. Keller: Scale-Invariant Features for 3-D Mesh Models. IEEE Transactions on Image Processing 21(5): 2758-2769 (2012).

[HS88] C. Harris, M. Stephens: A combined corner and edge detector, In Proc. 4th Alvey Vision Conference, 147–151, 1988.

[LCS14] G. Luo, F. Cordier, H. Seo, Similarity of Deforming Meshes Based on Spatio-temporal Segmentation, Eurographics Workshop on 3D Object Retrieval (3DOR) Strasbourg, France, April 2014.

[LL03] I. Laptev, T. Lindeberg: Interest Point Detection and Scale Selection in Space-Time. Scale-Space 2003: pp. 372–387.

[Lin94] T. Lindeberg, Scale-space theory: A basic tool for analyzing structures at different scales. Journal of Applied Statistics, 21(2): 224-270, 1994.

[Lin98] T. Lindeberg: Feature detection with automatic scale selection. International Journal of Computer Vision, 30, pp.79–116,1998.

[Low04] D. Lowe: Distinctive Image Features from Scale-Invariant Keypoints, 60(2): pp. 91–110, Int'l Journal of Computer Vision, 2004.

[LVJ05] C. H. Lee, A. Varshney, D. W. Jacobs, Mesh Saliency, pp. 659-666, ACM SIGGRAPH 2005.

[Mesh] Mesh data, http://people.csail.mit.edu/sumner/research/deft ransfer/data.html.

[MS01] K. Mikolajczyk and C. Schmid, Indexing based on scale invariant interest points, IEEE International

Conference on Computer Vision (ICCV), pp. 525–531, 2001.
[PKG03] M. Pauly, R. Keiser, M. Gross: Multi-scale Feature Extraction on Point-Sampled Surfaces, Computer Graphics Forum, 22(3), pp. 281–289, 2003.
[SAS07] P. Scovanner, S. Ali, and M. Shah: A 3-Dimensional SIFT Descriptor and Its Application to Action Recognition. Proc. the 15th International Conference on Multimedia, pp. 357-360, (2007).
[SB10] I. Sipiran and B. Bustos: A Robust 3D Interest Points Detector Based on Harris Operator, Eurographics workshop on 3D Object Retrieval (3DOR), pp. 7–14 2010.
[SP04] R. Sumner, J. Popovic: Deformation Transfer for Triangle Meshes. ACM Transactions on Graphics. 23, 3. August 2004.
[SRV98] A.H. Salden, B.M. ter Haar Romeny and M.A. Viergever, Linear Scale-Space Theory from Physical Principles, In Journal of Mathematical Imaging and Vision, Vol. 9, pp. 103-139, 1998.
[Vicon] Vicon motion capture system, http://vicon.com.
[ZBV*09] A. Zaharescu, E. Boyer, K. Varanasi, R. Horaud: Surface feature detection and description with applications to mesh matching, IEEE Computer Vision and Pattern Recognition (CVPR), pp. 373-380, 2009.

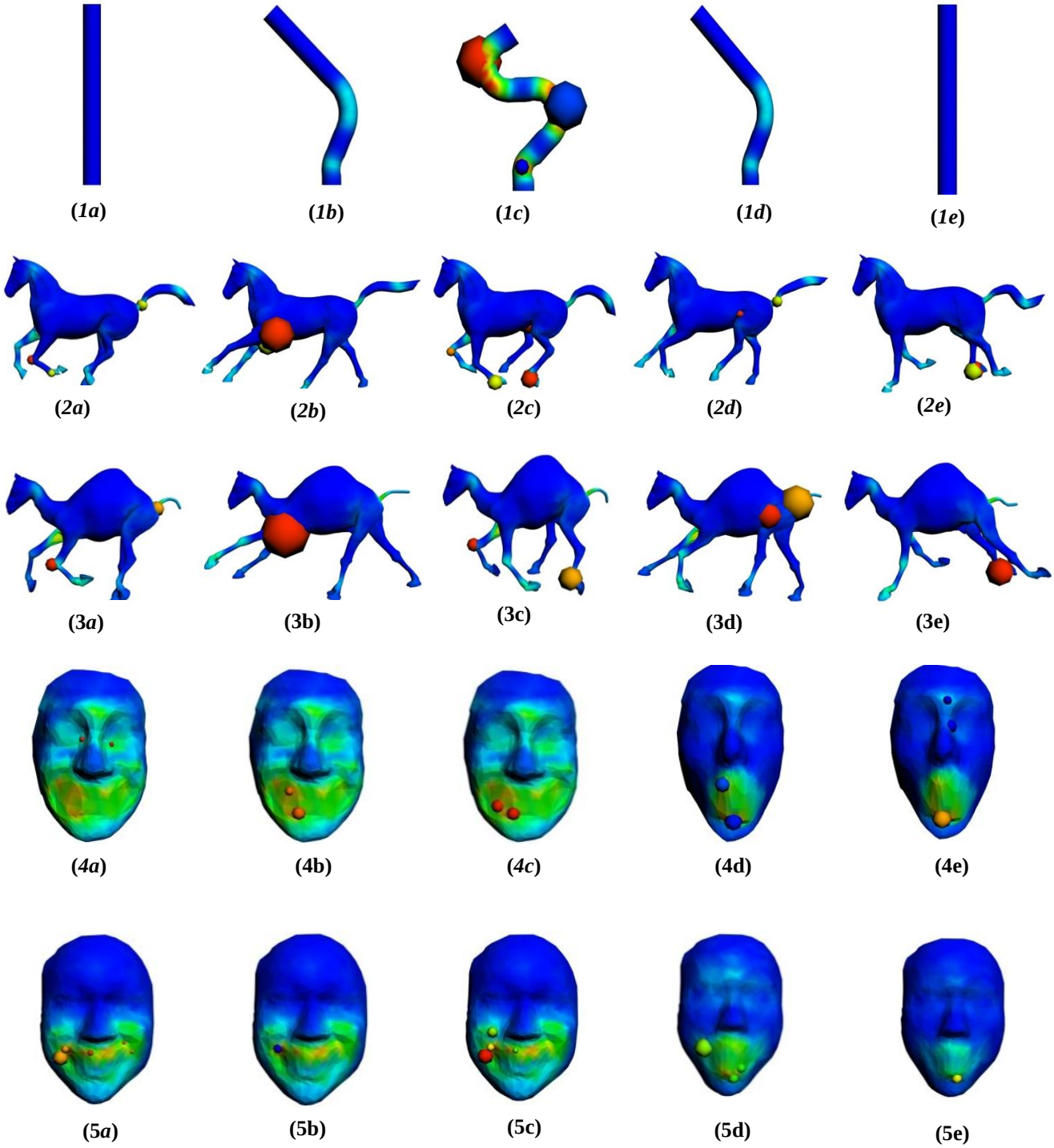


Fig. 8 Dynamic feature points detected by our AniM-DoG framework are illustrated on a number of selected frames of animated meshes. The color of a sphere represents the temporal scale (from blue to red) of the feature point, and radius of sphere indicates the spatial scale.