Graphical Models 73 (2011) 231–242

Contents lists available at ScienceDirect

Graphical Models

journal homepage: www.elsevier.com/locate/gmod

A robust and rotationally invariant local surface descriptor with applications to non-local mesh processing ☆

A. Maximo [a,*], R. Patro [b], A. Varshney [b], R. Farias [a]

[a] PESC/COPPE, Federal University of Rio de Janeiro, RJ, Brazil
[b] Department of Computer Science, University of Maryland, College Park, USA

ARTICLE INFO

Article history:
Received 20 May 2010
Received in revised form 29 March 2011
Accepted 9 May 2011
Available online 14 May 2011

Keywords:
Local descriptors
Non-local mesh processing
Shape analysis
Similarity processing

ABSTRACT

In recent years, we have witnessed a striking increase in research concerning how to describe a meshed surface. These descriptors are commonly used to encode mesh properties or guide mesh processing, not to augment existing computations by replication. In this work, we first define a robust surface descriptor based on a local height field representation, and present a transformation via the extraction of Zernike moments. Unlike previous work, our local surface descriptor is innately rotationally invariant. Second, equipped with this novel descriptor, we present *SAMPLE* – similarity augmented mesh processing using local exemplars – a method which uses feature neighbourhoods to propagate mesh processing done in one part of the mesh, the *local exemplar*, to many others. Finally, we show that SAMPLE can be used in a number of applications, such as detail transfer and parameterization.

1. Introduction

Surfaces discretized as triangular meshes are ubiquitous in Computer Graphics. The vast majority are generated by artists or scanned from real-world objects. Almost all surfaces, artificial or natural, feature regions with nearly identical properties, such as colour, shape, or texture. In this paper, we are interested in repeated shape patterns and how we can use them to propagate mesh processing.

Symmetries are most commonly detected and described as planar reflections among parts of a mesh [1–3]. In addition to these mirror-like symmetries, resemblances among details on a surface can also be found [4–6]. The concept of self-similarity of a meshed surface is defined by such resemblances, and naturally generalizes the concept of reflective symmetries. We say that two or more parts of a mesh are similar when they share local surface features, either reflective or not. Although the detection and structural analysis of reflective symmetries may be used to improve several mesh processing tasks [7], there have only been a few papers [8,9] dealing with repeated patterns with a close notion of mesh processing propagation as presented by our method.

The goal of this paper is to take advantage of a local surface descriptor over a mesh in order to propagate processing performed on a region to several other similar regions. The idea of similarity-augmented processing has been the target of study for many years in the Computer Vision community [10]. Recently, surface descriptors to measure similarity have draw the attention of the Computer Graphics community, mainly because of a large growth in the mathematical theory underpinning the discrete geometry of meshes; namely, discrete differential geometry [11,12]. Mean and Gaussian curvatures, the Laplace-Beltrami operator, and heat-diffusion processes on a triangular mesh have been used in defining vertex signatures based on their differential properties [4,6,13–15]. At the same time,

☆ This paper has been recommended for acceptance by Jarek Rossignac.

* Corresponding author.
E-mail addresses: andmax@cos.ufrj.br (A. Maximo), rob@cs.umd.edu (R. Patro), varshney@cs.umd.edu (A. Varshney), rfarias@cos.ufrj.br (R. Farias).

1524-0703/$ - see front matter © 2011 Elsevier Inc. All rights reserved.
doi:10.1016/j.gmod.2011.05.002

descriptors relying purely on the positions of vertices on the embedded surface also exist, such as the ones using Euclidean distances [16,17]. In this paper, we are interested in these types of descriptors and present a method to compute a novel local surface descriptor to measure similarity in Section 3; we extend this descriptor to consider surface regions rather than singular vertices in Section 4.1.

To build our surface descriptor, we use the Zernike-basis representation of the heightmap of the surface surrounding a given vertex as its descriptor (see an example of a vertex's heightmap in Fig. 1a). This simple 2D descriptor naturally encodes the surrounding shape, is robust with respect to the mesh triangulation, is efficient to compute and is rotationally invariant (see a more illustrative example in Fig. 2). This approach is similar to the spherical-harmonic representation employed by Kazhdan et al. [18] and to the work of Novotni and Klein[19] that uses 3D Zernike descriptors of the entire surface for shape retrieval. In contrast with these approaches, our descriptors encode a two-dimensional local surface signal rather than a three-dimensional representation of the entire mesh. This results in a similarity measurement which allows us to build a feature-neighbourhood space based on similar local shapes, explained in Section 4.2. We use this neighbourhood to propagate mesh processing operations in Section 4.3, such as mesh parameterization shown in Fig. 1c.

In this paper, we introduce a novel robust surface descriptor based on a local height field representation of a surface, along with a transformation which makes these descriptors rotationally invariant. Equipped with these descriptors, we present a system dubbed *SAMPLE* – similarity augmented mesh processing using local exemplars – which allows for the propagation of computation throughout the mesh, explained in detail in Section 4. The propagation is achieved by employing a feature-neighbourhood space, where similar regions are spatially proximate, illustrated by similar colours in Fig. 1b. We investigate two applications where SAMPLE can be used: detail transfer and mesh parameterization; discussed in Section 5. Nonetheless, any mesh processing task in which replication of processing is suitable can be augmented using our method. Results of the feature-neighbourhood space and replication ideas are presented in Section 6 and discussions about our work and similar approaches is delineated in Section 7. Conclusions and future work directions are given in Section 8.

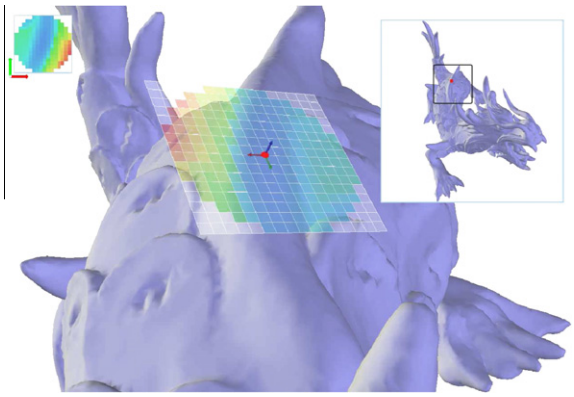

Fig. 2. Surface descriptor example for a given vertex (in red). The descriptor is a heightmap lying on the vertex tangent plane. The red and green arrows are the vertex principal directions, and the blue arrow is the vertex normal. The heights are the Euclidean distances from the plane to the mesh, where the blue and red colours represent low and high values, respectively. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

2. Related work

Measurements of the self-similarity of meshes are employed as a tool in many applications, such as remeshing [20–22], rendering [23], shape matching [9,24], shape retrieval [25,26], and scene understanding [27]. These self-similarity detection techniques generally employ reflection as the base comparison method to determine similar regions. The use of reflection in the majority of these techniques comes from the fact that symmetries in nature tend to have repeated patterns between halves, such as a human face, an animal body, or a plant leaf. However, both natural and artificial objects may also contain more general classes of self-similarities, for instance a computer keyboard has most of its keys sharing one

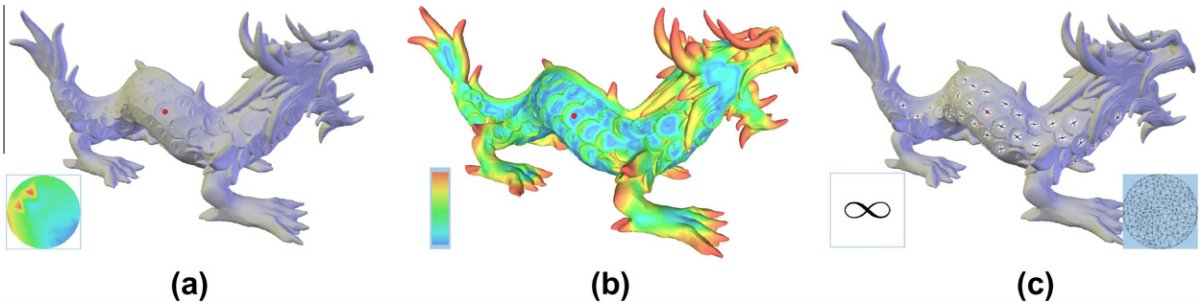

Fig. 1. Illustrative example of SAMPLE: (a) The Asian Dragon model with a vertex (in red) selected on the centre of one of its scales, and the heightmap descriptor of that vertex. (b) The model painted by the degree of similarity between regions and the selected vertex, from most similar (in blue) to least similar (in red). (c) The parameterization done on the surrounding region of the selected vertex (bottom right) is propagated to several other similar dragon scales, the same texture (∞) is mapped to each region. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

identical shape. One of the contributions of this paper is a method to detect self-similarities not limited to the ones based on reflection, which is adequate for scanned or modeled meshes, inspired by either natural or man-made objects.

One of the aspects which makes self-similarity identification especially challenging is the lack of a reasonable measurement tool for local shape comparison. Gatzke et al. [13] present a method to compare different local regions, introducing the *curvature map* of a point. They evaluate the mean and Gaussian curvature as a function of distance, using either neighbourhood rings or geodesic fans as Zelinka and Garland [8], for each point of the mesh. Thereafter, the self-similarity is measured as the difference among these curvature functions. In this work, we define a robust and efficient map of a point based on the heightmap of its surrounding region (see one example in Fig. 1a) instead of the curvature function. Section 7 further enumerates the differences of our work and the work of Zelinka and Garland [8].

Techniques aiming to identify self-similarities mainly depend on the shape properties of a surface. Much recent work in geometry processing considers the coordinate functions, or a more general function, as a signal defined over the meshed surface. One example is the method of spectral compression of meshes proposed by Karni and Gotsman [28]. This method encodes geometry information as a compact linear combination of the orthogonal eigenvectors of the discrete graph Laplacian. Vallet and Lévy's work [29] describes how to use eigenvectors of a geometry-aware formulation of the Laplace-Beltrami operator as a function basis for mesh geometry representation; namely the *Manifold Harmonic Basis*. Ovsjanikov et al. [5] present a method to compute global intrinsic symmetries using eigenfunctions. Their method determines pose-invariant correspondence over the shape, i.e. symmetries that remain intact under isometric deformations. Unfortunately, they restrict their method to reflective symmetries around the principal axes. Sun et al. [6] improve this method by defining a local vertex signature, called the *Heat Kernel Signature* (HKS). The HKS is a multi-scale descriptor of the shape surrounding the vertex based on the temporal evolution of a heat-diffusion process on a meshed surface. Although this method provides a concise descriptor, it does not readily admit a region-based descriptor that we need in our approach.

After the identification of self-similarities on a mesh, the question that remains is how to succinctly describe these similarities. Methods defining signatures per vertex, such as the HKS, are one way to describe similarities. Another way is to use a more global data structure. Simari et al. [30] present an algorithm to compute the *folding tree*, a compact data structure using planar symmetry. Their algorithm, however, is restricted to find symmetry about principal axes. Symmetry detection about more general planes, such as those based on Principal Component Analysis (PCA), has been explored by Kazhdan et al. [1] and Cheng et al. [31]. Perhaps the most general method for symmetry analysis of the entire object is the *planar-reflective symmetry transform* (PRST) [2] that captures reflective symmetries with respect to any plane. Xu et al. [3] improve the PRST idea to allow for the detection of partial intrinsic rotational symmetries.

Surface descriptors, ranging from local (per vertex) to global (the entire mesh), add information about a surface. Golovinskiy et al. [7] present a framework to exploit such information, describing mesh processing tools to detect and preserve symmetries using the PRST and Mitra et al.'s [16] work on partial symmetry detection. The symmetry-aware mesh processing of Golovinskiy and colleagues is guided by the symmetries of a surface, while our method discovers more general similarities on the surface and allows for the processing to be carried out among similar regions.

The usage of a descriptor to aid mesh processing guided by similarity is also explored by Yoshizawa et al. [32]. They use radial basis functions (RBFs) to approximate local vertex neighbourhoods, and describe similarity by the difference among local shapes encoded in these RBFs. These differences are used as weights to remove noise from meshes based on non-local image denoising techniques. Another work aiming to filter noise from meshes is presented by Schall et al. [17]. In contrast with Yoshizawa et al.'s work, they deal with range data, computing a pointwise height difference of the neighbourhood rather than using RBFs. Our method also computes height differences around a vertex; however, it can be used for any type of mesh, not only range image data. In addition, we show how our approach can be applied to several mesh processing tasks, such as parameterization and detail transfer.

3. Robust and rotationally invariant local descriptor

One of the main contributions of this paper is a novel local surface descriptor, as well as a procedure allowing the comparison of these descriptors in a robust and rotationally invariant manner. In this section, we describe both our descriptor and this procedure.

3.1. Local heightmap descriptor

We define the surface descriptor as a heightmap for each vertex, relying simply on Euclidean-distance measurements. The heightmap descriptor is computed either by hardware *Z-buffer* or by shooting rays from the vertex tangent plane to the meshed surface (see an example in Fig. 2). The tangent plane is defined by the vertex position and normal. The normal is computed using the area-weighted average of the surrounding face normals. The heightmap grid lying on the tangent plane is aligned with the two principal curvature directions computed at the vertex. The estimation of such differential geometric properties is not always robust. However, the usage of principal directions does not significantly impact the efficiency of the descriptors, as the alignment of the grid axes can be described by a simple rotation, and the descriptors are compared in a rotationally-invariant manner.

To compute the heightmap, we consider a bounded square sub-region of the tangent plane, with sides of length $\epsilon = 2.5\%$ of the diagonal of the mesh's bounding box. This sub-region is divided into a 16×16 grid, guiding

the casting of rays through each grid cell. We find that these values (grid size and sampling rate) captured the local shape features well in our experiments. Fig. 2 illustrates one example of a heightmap grid on the spine of the XYZ RGB Asian Dragon model.

For each grid cell, we cast one ray perpendicular to the tangent plane in both directions, keeping the closest hit as the height value for that cell. Heightmap cells without a hit value are set to an *infinity* value (1.5ϵ) and cells outside a fixed radius of ϵ are discarded (ignored in the following computations). The result is a 16×16 resolution image (see one example on the top-left corner of Fig. 2) describing the shape surrounding the vertex. This approach is robust to poorly shaped triangles, non-manifoldness and surfaces with holes, since it computes only ray-triangle intersections. A more complex approach using discrete differential geometric properties, such as the HKS [6], could result in a more descriptive signature, but our experiments indicate that such approaches are generally sensitive to triangle quality and topological singularities.

3.2. Achieving rotational invariance

Once we have the heightmap of each vertex, we can measure similarity between two vertices by computing the pixel-wise difference of the two heightmap images. Unfortunately, this leads to poor results. The main problem is that the principal directions of each vertex do not always align the heightmap properly (see Fig. 3). We address this problem by making use of a basis function which has proved useful in providing rotational invariance in the field of Computer Vision; the orthogonal Zernike polynomial functions [33].

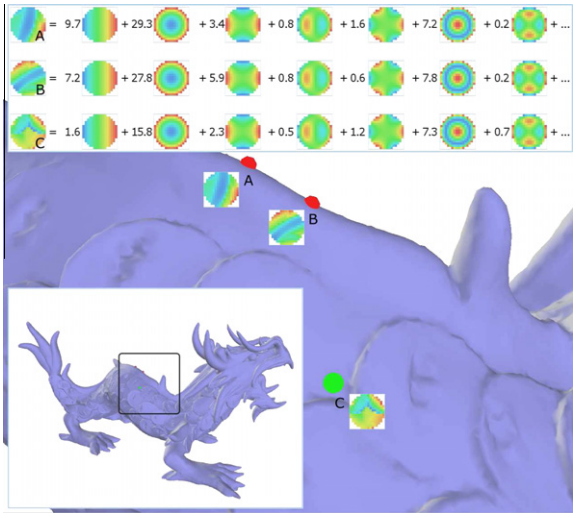

Fig. 3. Zernike expansions for heightmaps. Only the magnitudes of the complex values for the coefficients (top three row numbers) and Zernike polynomials (images after the equal signs) are shown. Although vertices A and B (in red) are similar, their heightmaps have improper alignment based on principal directions. By converting them to Zernike coefficients, the heightmaps are correctly compared. Vertex C (in green) is also correctly classified as different than vertices A and B. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

The straightforward approach for the alignment problem is to compute the difference for every rotation and choose the minimum value as the similarity. While this brute-force approach would yield the correct result, it is very inefficient. Instead of this approach, we convert each heightmap to Zernike coefficients and compare these coefficients rather than the image pixels. The Zernike polynomials constitute an orthogonal basis for functions defined on the unit disk. Each Zernike polynomial, V_p^q, has an associated order p and repetition q, and they are defined over the domain $D = \{(p,q) | q \in \mathbb{Z}, p \in \mathbb{Z}^{\geqslant 0}, |q| \leqslant p, |p-q| \in \mathbb{Z}^{\text{even}}\}$ as follows:

$$V_p^q(\rho,\theta) = R_p^q(\rho) e^{iq\theta} \tag{1}$$

where R_p^q is the radial polynomial given by:

$$R_p^q(\rho) = \sum_{\substack{k=|q| \\ |p-q| \text{even}}}^{p} \frac{(-1)^{\frac{p-k}{2}} \frac{p+k}{2}!}{\frac{p-k}{2}! \frac{k-q}{2}! \frac{k+q}{2}!} \rho^k \tag{2}$$

To obtain the Zernike coefficients of a function $f(x,y)$, we apply:

$$z_p^q(f) = \frac{p+1}{\pi} \int\int_{x^2+y^2 \leqslant 1} \left(\overline{V}_p^q\right)(x,y) f(x,y) dx\, dy \tag{3}$$

where $(\overline{V}_p^q)$ denotes the complex conjugate of (V_p^q). The Zernike polynomials form a basis upon which an image f can be projected. The result of this projection is the Zernike moments of the image, the magnitudes of which are invariant to rotation [33]. In practice, each of the Zernike basis functions is represented as a set of discrete samples on a $k \times k$ grid. However, analogously to the continuous case, the samples take on a value of zero outside a circular region centred at $[c_x, c_y] = [\frac{k}{2}, \frac{k}{2}]$. Each Zernike basis function is supported in the region $\mathbf{S} = \left\{[x,y] | \sqrt{(x-c)^2 + (y-c)^2} \leqslant \frac{k}{2}\right\}$. In this way, we can define the projection of an image f onto the Zernike basis function $\overline{V}_p^q$ as:

$$z_p^q(f) = \frac{p+1}{\pi} \sum_{(x,y) \in \mathbf{S}} \left(\overline{V}_p^q\right)[x,y] f[x,y] \tag{4}$$

For our heightmap images (with 16×16 resolution), we project $f(x,y)$ onto 25 Zernike polynomials (corresponding to non-negative repetitions and up to the 8th polynomial order) resulting in a vector $\mathbf{z}_i$ of Zernike moments for each vertex v_i. We have noticed that 25 coefficients are sufficient to represent the vertex's descriptor. Fig. 3 shows an example of two heightmap images (A and B) with an improper alignment using the principal directions, whereas the Zernike coefficients of the two images have close values, correctly classifying as similar the two vertices on the spine of the Asian Dragon model. The processing time spent in computing the brute-force approach is three orders of magnitude greater than the computation using Zernike coefficients. More specifically, employing Zernike coefficients reduces the similarity measurement time of entire meshes from hours to seconds.

4. Similarity augmented mesh processing

The main idea behind SAMPLE – similarity augmented mesh processing using local exemplars – is to provide a method to propagate computations from an exemplar region to many other similar regions of a mesh. The concept of similarity in this scenario depends on the mesh properties needed by the mesh processing task. For example, if the processing takes into account the local shape properties, such as normals and curvatures, the surface descriptor should encode these features. When propagating mesh processing, the descriptor is used to establish feature neighbourhoods where the computation can be replicated throughout the mesh.

We use two spaces to accomplish similarity augmented mesh processing: *geometry* and *feature* spaces. The geometry space defines the regular neighbourhood of a local surface patch, given by the mesh connectivity and geometry, and the feature space defines the similarity neighbourhood, given by the distance between surface descriptors. The geometry space is widely used by many existing mesh processing tasks, such as Laplacian smoothing [34]. The feature space, on the other hand, is scarcely used in mesh processing, and it is normally employed as a feature neighbourhood for averaging values [32,17]. We use the feature space in a more comprehensive manner. In our case, the feature neighbours define correspondences across regions in addition to the geometry neighbours, aiding in the replication of processing done on an exemplar region to other close regions in the feature space.

Once the geometry and feature spaces are established, the non-local propagation of mesh processing can be carried out. We illustrate our idea by applying the propagation algorithm in transferring the same details to many similar regions seamlessly, and by reusing the parameterization computed for a patch over the mesh to others similar patches. Both applications are well-suited for propagation through a similarity-based descriptor using local shape features, as explained in detail in the next section. However, additional mesh processing tasks could use different descriptors, e.g. planar-reflective symmetry [2] or a saliency-based descriptor [35].

4.1. Establishing the feature space

With the technique explained in Section 3.2, we have a robust similarity measurement among vertices. Typically, the feature neighbourhood is established by considering the point-wise self-similarity of a triangulated mesh, i.e. distances between the shape descriptors of its constituent vertices. However, the feature space resulting from computing descriptors of isolated vertices may have misplaced similarity correspondences for nearby vertices, and it may ignore distinctive shape features in favour of large flat regions (see example in Figs. 4 and 5, left). We solve this problem by considering entire neighbourhoods rather than single vertices when building the feature space. For each vertex, we apply Gaussian weights to the Zernike coefficients, adding the coefficients of nearby vertices within a fixed radius of ϵ. The Gaussian filter cut-off is set to be at a distance of 2σ, where σ is set to $\epsilon/2$ to include all vertices inside the region considered by the heightmap.

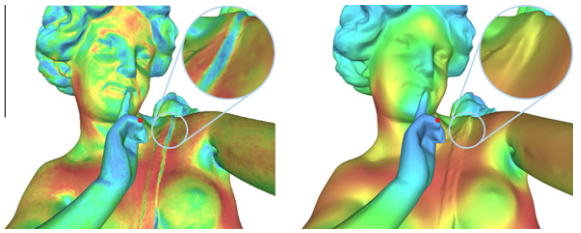

Fig. 4. Difference between vertex-to-vertex (left) and region-to-region (right) comparison methods. The similarity measurement based on individual vertices yields the incorrect comparison of the selected vertex (in red) to the strap region close to the shoulder (top-right corner). The model is painted from most similar (in blue) to least similar (in red). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

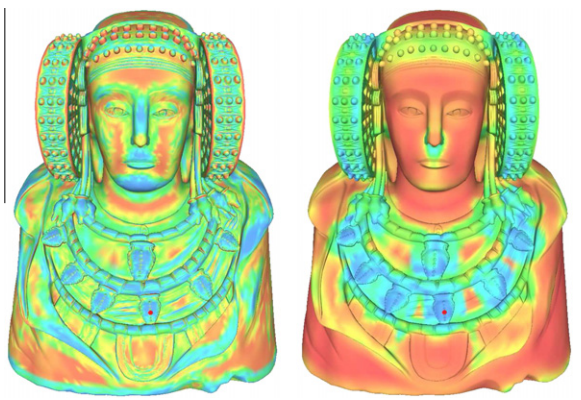

Fig. 5. The feature space of the Dama model further illustrates the difference from vertex-based (left) to region-based (right) comparisons. Excessive blue colour in the left image shows that the vertex-based technique is not as discriminative as the region-based technique shown in the right image. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Recall that we have defined $\mathbf{z}_i$ to be the vector of Zernike coefficients for the vertex v_i. Our region-based comparison method uses the geometry space of v_i, denoted by $\mathcal{N}(v_i, \sigma)$, which includes vertices within a distance σ of v_i. To apply our Gaussian-weighted comparison scheme, we consider a new vector of Zernike coefficients for each vertex defined as follows:

$$\mathbf{z}_i^{\sigma} = \frac{\sum_{v_j \in \mathcal{N}(v_i, 2\sigma)} \mathbf{z}_i e^{-\frac{|v_i - v_j|^2}{2\sigma^2}}}{\sum_{v_j \in \mathcal{N}(v_i, 2\sigma)} e^{-\frac{|v_i - v_j|^2}{2\sigma^2}}} \tag{5}$$

The $\mathbf{z}_i^{\sigma}$ is defined as a Gaussian-weighted average of the Zernike coefficient vectors of all vertices within a radius 2σ of v_i. These newly formed vectors of Gaussian-weighted Zernike coefficients replace the original vectors, resulting in the generation of a smooth and high quality feature space. Note that this new vectors are still defined over each vertex of the mesh, though now they include information about other vertices in the surrounding region. Figs. 4 and 5 illustrate the 3DScanCo Angel and Dama models

using vertex-based comparison and the Gaussian-weighted region method.

It is important to note that this region-based comparison benefits greatly from the rotational invariance of our descriptor. While this property is not strictly necessary to define a descriptor, it allows for the agglomeration of local descriptors without explicit concern of their alignment. In the absence of the rotational invariance property, an attempt to build a region-based descriptor from an agglomeration of vertex-based descriptors (e.g. geodesic fans) must address the challenging question of how these descriptors should be aligned and combined. A brute-force approach is difficult, because attempting to find the best alignment between all vertex descriptors in the region is a significantly more difficult problem than simply finding the minimum distance alignment between two descriptors.

4.2. Similarity measurement

After extracting the descriptors as explained in Section 3 and accounting for local regions as explained in Section 4.1, we obtain 25 coefficients for each vertex, which constitute its embedding coordinates in the feature space. Our feature space can simply be viewed as $\mathbb{R}^{25}$, where similar vertices can be found by searching for nearest neighbours in this space, under a Euclidean metric, for any given vertex.

The similarity distance s in the feature space between vertices v_i and v_j is defined as follows:

$$s(v_i, v_j) = d\left(\mathbf{z}_i^{\sigma}, \mathbf{z}_j^{\sigma}\right) \tag{6}$$

where $d(\cdot,\cdot)$ denotes the Euclidean distance. The mesh feature space depends only on the Gaussian-weighted Zernike coefficients of each vertex, $\mathbf{z}_i^{\sigma}$, which can be pre-computed and stored in an auxiliary data structure for similarity measurements. The feature neighbours of a vertex v_i are defined as the set of vertices within a distance τ, where $s(v_i, v_j) < \tau$. Fig. 6 shows a number of feature space neighbours of one vertex on the scale of the Asian Dragon model, the vertices in the geometry neighbourhood are illustrated for comparison.

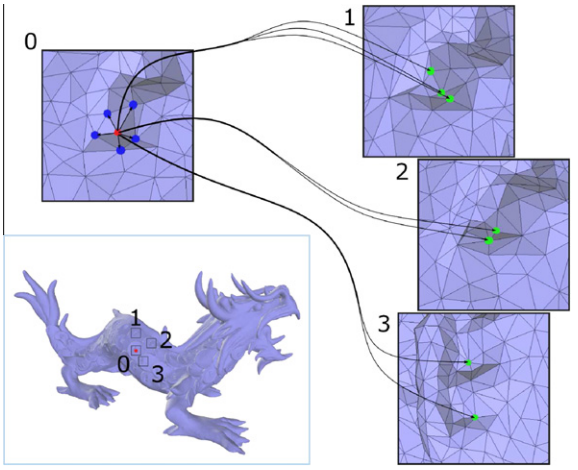

Fig. 6. Illustration of the geometry and feature spaces. Several geometry neighbours of the selected vertex (in red) on a dragon scale are shown in blue (box 0). While some of the feature neighbours are shown in green (boxes 1–3). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

4.3. Propagating mesh processing

The SAMPLE algorithm propagates mesh processing by using the feature space and a *local exemplar*. An exemplar region is any region of the mesh with desired features, which is used to guide the propagation of a target processing task. For example, to paint all the scales of the Asian Dragon model with the same drawing, one of the scales can be selected as the exemplar region and, as the texture painting occurs, all regions locally similar to the exemplar are painted with the same pattern. The disadvantage of this approach is that it depends on the size of the local exemplar, which is directly related to the size of the heightmap descriptor explained in the previous section. If the exemplar region is much smaller or larger than the heightmap, then the feature space fails to capture the distinctive features of the local shape, providing the incorrect similar neighbours upon which the processing is propagated. In these cases, the feature space has to be recomputed considering an ϵ value chosen to match the size of the desired region and yield the correct result.

The feature space plays the main role in the propagation of mesh processing. It is responsible for describing the similarity correspondences among vertices, which are used to propagate computation across similar regions. These regions are defined using the geometric neighbourhood of similar vertices. After a local exemplar is chosen, we can determine its feature neighbours by finding vertices whose feature space embedding reside within a user-defined distance threshold τ from the embedding of the exemplar's centre vertex. Then, the operations performed on the exemplar can be carried out on these other similar regions through the centre vertices. The similarity threshold defines which regions are affected by the propagation. Smaller threshold values affect fewer regions, while larger values affect more regions.

Although the comparison method used to build our feature space considers entire mesh regions, the feature neighbourhood is defined among mesh vertices. The feature neighbours of a given vertex may also be close to this vertex in the geometry space. This scenario can compromise the propagation of mesh processing. We solve this problem by considering only neighbouring vertices in the feature space whose geometric regions do not overlap each other. That is, the size of the exemplar region defines the minimum distance in the geometry space that we will require between any pair of vertices before we consider them for the propagation of processing.

After these considerations, we can now describe how to propagate mesh processing between the local exemplar and a target region. We define a local coordinate frame for each vertex undergoing the propagation in order to replicate processing consistently. This frame is based on the same vectors used to compute and align the heightmap descriptor over the vertex tangent plane, and it is used to

assign local (u,v,w) coordinates for each vertex inside the affected regions. These coordinates replace the original (x,y,z) coordinates while processing similar regions, creating a common coordinate system throughout the propagation. The size of each affected region is defined by the exemplar region, where the desired local shape resides. Since our similarity measurement is rotationally invariant and the heightmaps of the affected regions may not be aligned, we rotate the (u,v) coordinates by an angle (α) which is chosen to achieve the minimum similarity difference between the affected and exemplar regions. Although a technique exists to retrieve the rotational angle using Zernike moments [36], in our experiments, this method yields several local minima for α, degrading the propagation. Since we generally have a small number of regions among which to propagate results, we compute the α angle using the brute-force approach discussed in Section 3.2, i.e. evaluating the differences for all discrete rotations and choosing the angle resulting in the minimum value.

Finally, the local coordinates are used to find the nearest vertices in the exemplar region and interpolate an approximate result value of any processing done on the local exemplar for the vertex on the target affected region. The quality of this approximation depends on the processing performed and the quality of the mesh samples. A more regularly sampled mesh generally yields better results when propagating processing operations than a poorly sampled one. In the next section, we illustrate this idea using our SAMPLE algorithm in two applications: detail transfer and mesh parameterization.

5. Applications

We present two illustrative applications of our SAMPLE method in order to demonstrate the feature space and the propagation ideas. The first replicates the parameterization computed for one exemplar region to several other similar regions, preserving scale and local orientation of the parameterization domain. The second propagates a given detail mask to many similar regions, preserving local consistency.

The first application aims to reuse a local mesh parameterization taking advantage of the feature space to replicate the computed parameters. We perform a simple parameterization, using the *Floater Mean Value Coordinates* algorithm [37], on the selected exemplar region, and subsequently visit a number of similar regions which reside within a given similarity distance threshold in the feature space. For each similar region, we assign values from the exemplar parameterization corresponding to the local coordinates (u,v,w) explained in Section 4.3. In this way, the same parameterization domain is shared among regions with similar local shape. Fig. 7 shows the replication process applied to the XYZ RGB Asian Dragon model.

Note in Fig. 7 that a small number of scales are not affected by the process, since they consist of vertices whose feature space embedding reside beyond the specified distance threshold from the exemplar scale. We chose this distance threshold because it allows us to select most of the similar scales without producing false positives, i.e. similar non-scale regions, which tend to occur at greater similarity distance thresholds. Also, note that although the replication procedure induces a maximum error displacement of 0.047 units on the parameterization disk domain, the textures are not visibly distorted.

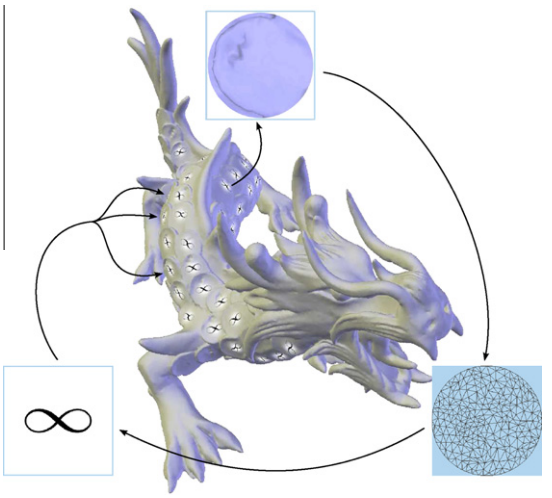

Fig. 7. SAMPLE application: mesh parameterization. One of the dragon scales is elected to be the exemplar region (top); the exemplar is parameterized using a standard procedure (bottom right); the infinity logo image (bottom left) is used to texture the similar dragon scales seamlessly.

The second application aims to use the feature space to propagate mesh editing throughout meaningful regions. We use a simple editing mask, a Gaussian-weighted extrusion of a fixed radius (shown in the bottom-left corner of Fig. 8), applied to the same exemplar dragon scale used in the first application. We also use the same similarity distance threshold to affect the identical scales, illustrating that our SAMPLE algorithm can be employed in a similar manner for a broad range of mesh processing tasks. In this

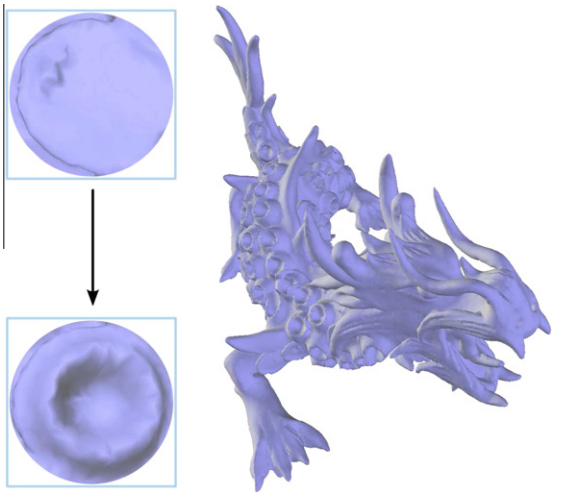

Fig. 8. SAMPLE application: detail transfer. One exemplar region (top left) of a dragon scale is edited to resemble a crater (bottom left). Our similarity augmented algorithm propagates the editing result naturally over the similar scales.

application, the mesh that results from the propagation of the editing performed on the single exemplar region is identical to the mesh which would result if all of the similar regions had been edited manually. This illustrates how the SAMPLE system can be used to reduce the burden on an artist, who would otherwise have to manually edit each of the similar regions to obtain the same result.

Note that in Figs. 8 and 11 we choose to use, respectively, rotationally-symmetric displacements and textures. This is done mainly because the local coordinate frame alignment may lead to unintended effects globally, even though the replication is locally coherent.

6. Results

We have introduced the concept of the feature space of a mesh, defining a novel local surface descriptor and how to use it to embed vertices in the mesh feature space. In addition to the use of this space to augment mesh processing tasks exploring similarity, the Euclidean distances among points in this space can also reveal reflective symmetries. These symmetries can be viewed as a subset of the more general class of similarities described by our SAMPLE method. Fig. 9 depicts the Stanford Armadillo model and its feature space with respect to one vertex. Note the symmetric patterns which appear naturally when colouring vertices by the distance order in feature space.

Another interesting result is the 3DScanCo Angel model, shown in Fig. 10. Unlike the Armadillo, the Angel does not exhibit a simple plane of symmetry, yet it still contains an underlying symmetric structure. The feature space for this model also highlights the symmetric patterns when colouring the vertices by similarity distance.

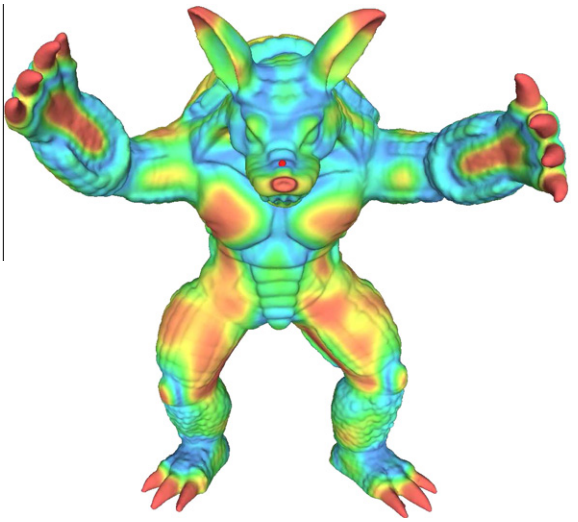

Fig. 9. Feature space example for a given vertex (in red). The surrounding vertex region is used to map the entire mesh using the similarity neighbourhood. Regions with wrinkles similar to the selected vertex on the nose are painted with blue, such as the ones in the arms and feet, while more dissimilar regions are gradually painted from green to red. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

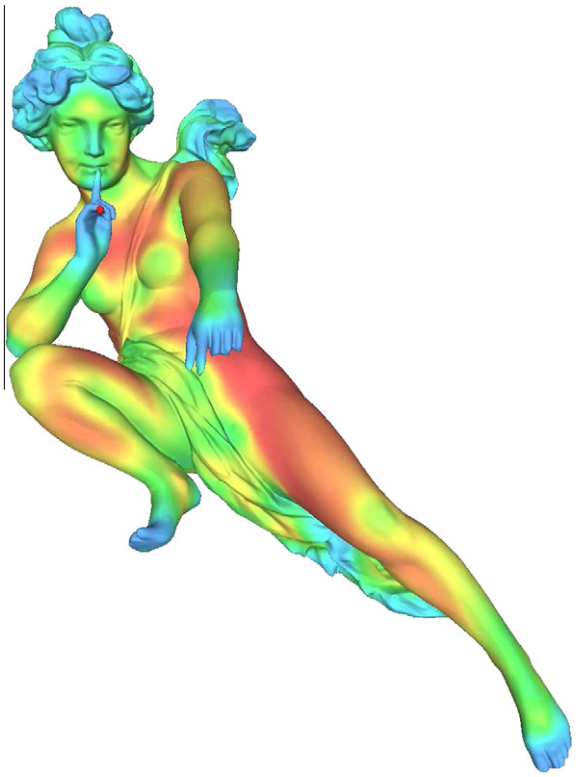

Fig. 10. Symmetry example using the feature space. For a vertex on the right hand (in red), the model is painted from close (in blue) to far (in red) regions in the feature space. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

While the feature space can reveal global symmetries, considering only the k nearest neighbours in this space is an interesting special case. In particular, when $k = 1$, the feature space provides a result which closely matches one's intuition. Fig. 11 illustrates how the mesh parameteriza-

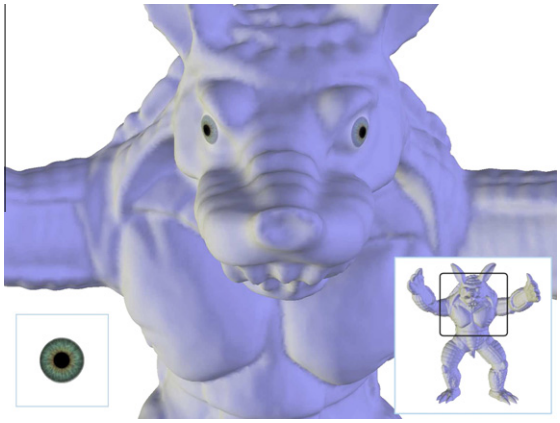

Fig. 11. Application example of the immediate feature neighbour. The vertices in the centre of each eye are immediate neighbours in feature space, choosing either vertex and applying our similarity augmented mesh parameterization leads to the automatic mapping of a texture (bottom left) to both eyes.

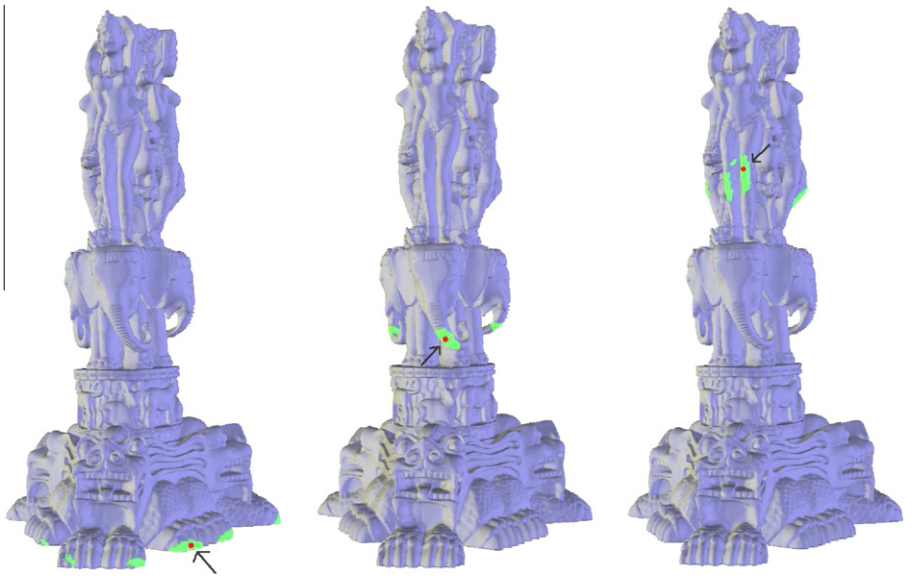

Fig. 12. Nearest similarities (in blue) of the XYZ RGB Thai Statue model. Three vertices (in red) are used in this example: on the bottom of the statue (left) finding several toes; finding the three elephants' trunk in the middle; and on the top (right) finding several knees. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Table 1
Computation time for establishing the feature space. The first column shows the number of vertices (#v) and faces (#f), while the last three columns show the time spent in each pre-computation step of our algorithm – Heightmap descriptor (H), Zernike expansion (Z) and Gaussian average (G).

Model	#v/#f (K)	H (min)	Z (s)	G (min)
Dama	106/207	3.20	5.53	2.47
Dragon	108/216	2.56	5.54	1.83
Statue	125/250	2.98	6.58	2.07
Armadillo	172/345	6.12	9.01	5.34
Angel	237/474	17.76	12.15	18.45

tion of one eye of the Armadillo model can be duplicated to the other eye.

Analyzing the k nearest neighbours in feature space is also interesting to reveal symmetric regions in highly-detailed models. Take for example the XYZ RGB Thai Statue model shown in Fig. 12. Even with a large number of details on the surface, our feature space is able to capture small similarities using $k = 0.5\%$ of the total number of vertices.

The pre-computation time required to build the feature space is shown in Table 1 and depends on the following computations: the heightmap descriptor, which is a standard procedure linear in the number of vertices; the expansion in Zernike coefficients of each heightmap image; and the computation of Gaussian-weighted Zernike coefficients, which is a fixed neighbourhood summation also linear in the number of vertices. The timings for the heightmap descriptor and Gaussian average computations are only preliminary, since we do a serial ray-casting approach to compute the heightmaps and a straightforward regular grid to find neighbours for the Gaussian averaging. In practice, this step can be almost completely parallelized across the different vertices. The computation of Zernike coefficients requires, for the XYZ RGB Asian Dragon model (with 108 K vertices), 5.54 s on a 2.33 GHz Intel Xeon E5345 CPU with 4 GB RAM. Moreover, the XYZ RGB Thai Statue model (with 125 K vertices), the Stanford Armadillo (with 172 K vertices) and 3DScanCo Angel (with 237 K vertices) models require 6.58, 9.01 and 12.15 s, respectively, providing empirical validation that the computation of the Zernike coefficients also scales linearly in the number of vertices.

The run time to compute feature neighbours for a given vertex using the pre-computed feature space depends on the evaluation of similarity differences and nearest neighbourhood searching. We sort the similarity differences, making the search trivial. The computation of the differences and the sorting operations for a selected vertex require 0.02 s for the Asian Dragon model. The propagation time, on the other hand, mainly depends on the mesh processing task to propagate. In our experiments, the parameterization of one dragon scale consumed 0.01 s, while the propagation to the 57 similar scales consumed 0.26 s. Finally, the detail transfer applied to the same scales consumed 0.29 s.

7. Discussion

In this section we enumerate the main differences, advantages and disadvantages of our method and other approaches.

The descriptor provided by the HKS method [6] can be used to build a feature space similar to our approach, as shown in Fig. 13. For this example, we use the same logarithmically scaled temporal domain as the original paper to build the HKS for each vertex (with 100 sampled values), and choose a range for the time parameter t to

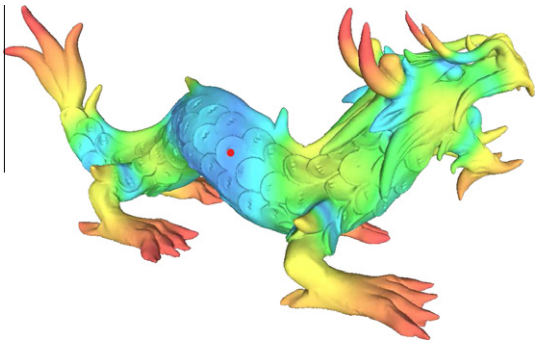

Fig. 13. Feature space using HKS. The colours show the Euclidean differences between the HKS of the selected vertex (in red) and all other vertices, considering the time parameter t of the heat-diffusion process to be in range [12,32] over the same sampled scale of the original paper [6]. The difference increases as the colour changes from blue to green to red. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

capture the shape of one dragon scale. Unfortunately, there is no direct relation between the time parameter and the geometric neighbourhood, making a region-based similarity comparison based on HKS difficult.

The feature space in Fig. 13 is illustrated in the same way as presented in our method, by colouring the model from most similar (in blue) to least similar (in red). Although the HKS method is able to find several similar dragon scales in the middle, it loses several others and incorrectly classifies as similar regions in the spine and ears of the Asian Dragon model.

The pioneering work of Zelinka and Garland [8] introduced the idea of similarity based (non-local) mesh processing in earnest. They developed a local surface descriptor, based on geodesic fans, capturing a notion of similarity which largely agrees with intuition. It is important to note that while our work is motivated by the success of Zelinka and Garland's approach, our descriptors provide several benefits over geodesic fans. In particular, we would like to draw two distinctions between these works. First, the geodesic fan concept relies upon the accurate computation of geodesic strips, which may be difficult or impossible to compute in cases of non-manifoldness or holes in the mesh. Also topological inconsistencies, which result in regions with similar appearance but distinct topology, will not be discovered by the geodesic-fan descriptor. However, the heightmap descriptor introduced in this paper is robust to such holes and topological artifacts (see Fig. 14). As a result, we find that our descriptor also behaves in an intuitive manner, and that regions presenting visual similarities are discovered as similar by our approach.

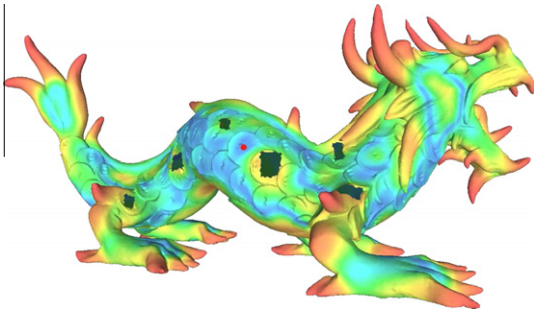

Fig. 14. Test of our feature space using the Asian Dragon model with holes and inconsistencies. While the dragon scales with holes are classified as different than the selected scale, the other scales remain correctly classified.

The second distinction is that the geodesic fans require alignment to compute similarity. To make their representation rotationally invariant, Zelinka and Garland consider storing all possible (discrete) rotations of their descriptor linearizing each fan into a high-dimensional feature vector. Then, an indexing structure, such as a kd-tree, may be used to quickly perform a similarity search. The problem of this approach is that it increases the storage burden considerably. To offset the negative effects on storage, they show that the resultant feature vectors can then be reduced in dimensionality by applying tree-structured vector quantization (TSVQ). In contrast with Zelinka and Garland's approach, our method avoids such complexity by directly providing a rotationally invariant representation of our surface descriptor. Once we extract the Zernike moments of our descriptors, only a single inner product is required to compute similarity between two descriptors. Though we have not found the addition of TSVQ and indexing steps necessary, we would like to note that it is possible to further increase the comparison performance of our local surface descriptors and their effective size by applying the same type of fast indexing methods exploited by Zelinka and Garland. Finally, our descriptors admit a natural extension to a region-based descriptor, while such an extension using geodesic fans is not clear.

8. Conclusion

We have introduced a new robust and rotationally invariant surface descriptor for similarity and SAMPLE, a framework for similarity augmented mesh processing, and shown how processing and editing operations performed on one region of a meshed surface can be automatically transferred to other similar regions. One of the primary constituents of our framework is a novel mesh surface descriptor. For a given vertex, we build a heightmap by considering the distance from its tangent plane to the meshed surface. Moreover, we show how these descriptors can be effectively compared. First, we compute the Zernike polynomial expansion of the heightmap descriptor, the coefficients of which yield a rotationally invariant representation. Second, we devise a region-based descriptor, rather than vertex-based, by applying Gaussian weights to the Zernike coefficients of nearby vertices.

After defining a similarity measurement, we introduce the idea of feature space. We define this space in which mesh vertices are embedded using the Gaussian-weighted Zernike coefficients of their corresponding descriptors. This space provides a powerful tool for a new form of similarity-driven mesh processing. We show how the feature space can be used to propagate processing across similar regions, using parameterization and editing as two

examples. The latter tool can be used to replicate mesh editing operations automatically across similar regions; for instance, reducing the burden on artists of repeatedly performing the same editing operation.

We believe that our similarity augmented mesh processing framework shows promising results and presents numerous avenues for future research. One future direction of research is to consider different and more robust ways to compute the vertex normals, since it has a direct impact in our descriptor. Another consideration is to study other formulations for the local surface descriptor. The heightmap we currently use can be viewed as a scalar function parameterized over the tangent plane of the centre vertex. It may be possible to form more informative descriptors by considering different parameterizations, such as authalic or conformal, for the region surrounding the centre vertex. A third possibility is to consider different scalar functions, such as curvature or saliency, to be evaluated over the parameter domain. One final interesting avenue for future research is to extend our surface descriptor to take into account surface regions at multiple scales. The success of multi-scale descriptors in the field of image processing suggests that such an approach might have significant benefits over the current single-scale implementation of our SAMPLE method.

Acknowledgments

We would like to acknowledge the grant of the first author provided by Brazilian agency CNPq (National Counsel of Technological and Scientific Development). This work has been supported in part by the NSF Grants: CCF 04-29753, CNS 04-03313, CCF 05-41120 and CMMI 08-35572. We also gratefully acknowledge the support provided by the NVIDIA CUDA Center of Excellence award to the University of Maryland and constructive discussions with David Luebke at NVIDIA Research. Any opinions, findings, conclusions, or recommendations expressed in this article are those of the authors and do not necessarily reflect the views of the research sponsors. We would also like to acknowledge the Stanford Graphics Lab, Cyberware Inc. and XYZ RGB Inc. for providing the models used in this paper.

References

[1] M. Kazhdan, B. Chazelle, D. Dobkin, T. Funkhouser, S. Rusinkiewicz, A reflective symmetry descriptor for 3D models, Algorithmica 38 (1) (2003) 201–225, doi:10.1007/s00453-003-1050-5.

[2] J. Podolak, P. Shilane, A. Golovinskiy, S. Rusinkiewicz, T. Funkhouser, A planar-reflective symmetry transform for 3D shapes, in: SIGGRAPH '06: Proceedings of the 33th Annual Conference on Computer Graphics and Interactive Techniques, ACM, New York, NY, USA, 2006, pp. 549–559. doi:10.1145/1179352.1141923.

[3] K. Xu, H. Zhang, A. Tagliasacchi, L. Liu, G. Li, M. Meng, Y. Xiong, Partial intrinsic reflectional symmetry of 3D shapes, in: SIGGRAPH Asia '09: ACM SIGGRAPH Asia 2009 papers, ACM, New York, NY, USA, 2009, pp. 1–10. doi:10.1145/1661412.1618484.

[4] R.M. Rustamov, Laplace–Beltrami eigenfunctions for deformation invariant shape representation, in: SGP '07: Proceedings of the Symposium on Geometry Processing, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 2007, pp. 225–233.

[5] M. Ovsjanikov, J. Sun, L.J. Guibas, Global intrinsic symmetries of shapes, Computer Graphics Forum 27 (5) (2008) 1341–1348, doi:10.1111/j.1467-8659.2008.01273.x.

[6] J. Sun, M. Ovsjanikov, L.J. Guibas, A concise and provably informative multi-scale signature based on heat diffusion, Computer Graphics Forum 28 (5) (2009) 1383–1392, doi:10.1111/j.1467-8659.2009.01515.x.

[7] A. Golovinskiy, J. Podolak, T. Funkhouser, Symmetry-aware mesh processing, in: Proceedings of the 13th IMA International Conference on Mathematics of Surfaces, Springer-Verlag, Berlin Heidelberg, 2009, pp. 170–188. doi:10.1007/978-3-642-03596-8_10.

[8] S. Zelinka, M. Garland, Similarity-based surface modelling using geodesic fans, in: SGP '04: Proceedings of the Symposium on Geometry Processing, ACM, New York, NY, USA, 2004, pp. 204–213. doi:10.1145/1057432.1057460.

[9] R. Gal, D. Cohen-Or, Salient geometric features for partial shape matching and similarity, ACM Transactions on Graphics 25 (1) (2006) 130–150, doi:10.1145/1122501.1122507.

[10] D. Gavrila, L. Davis, 3-D model-based tracking of humans in action: a multi-view approach, IEEE Computer Society Conference on Computer Vision and Pattern Recognition (1996) 73.

[11] M. Meyer, M. Desbrun, P. Schröder, A.H. Barr, Discrete Differential-Geometry Operators for Triangulated 2-Manifolds, Springer-Verlag, 2002. pp. 35–57.

[12] B. Levy, Laplace–Beltrami eigenfunctions towards an algorithm that understands geometry, International Conference on Shape Modeling and Applications (2006) 13.

[13] T. Gatzke, C. Grimm, M. Garland, S. Zelinka, Curvature maps for local shape comparison, International Conference on Shape Modeling and Applications (2005) 244–253.

[14] M. Pauly, N.J. Mitra, J. Wallner, H. Pottmann, L.J. Guibas, Discovering Structural Regularity in 3D Geometry, in: SIGGRAPH '08: ACM SIGGRAPH 2008 Papers, ACM, New York, NY, USA, 2008, pp. 1–11. doi:10.1145/1399504.1360642.

[15] A. Itskovich, A. Tal, Surface partial matching and application to archaeology, Computers & Graphics 35 (2) (2011) 334–341, doi:10.1016/j.cag.2010.11.010.

[16] N.J. Mitra, L.J. Guibas, M. Pauly, Partial and approximate symmetry detection for 3D geometry, ACM Transactions on Graphics 25 (3) (2006) 560–568, doi:10.1145/1141911.1141924.

[17] O. Schall, A. Belyaev, H.-P. Seidel, feature-preserving non-local denoising of static and time-varying range data, in: SPM'07: Proceedings of the ACM Symposium on Solid and Physical Modeling, ACM, New York, NY, USA, 2007, pp. 217–222. doi:10.1145/1236246.1236277.

[18] M. Kazhdan, T. Funkhouser, S. Rusinkiewicz, Rotation invariant spherical harmonic representation of 3D shape descriptors, in: SGP '03: Proceedings of the Symposium on Geometry Processing, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 2003, pp. 156–164.

[19] M. Novotni, R. Klein, 3D zernike descriptors for content based shape retrieval, in: Proceedings of the Eighth ACM Symposium on Solid Modeling and Applications, SM'03, ACM, New York, NY, USA, 2003, pp. 216–225. doi:10.1145/781606.781639.

[20] J. Podolak, A. Golovinskiy, S. Rusinkiewicz, Symmetry-enhanced remeshing of surfaces, in: SGP '07: Proceedings of the Symposium on Geometry Processing, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 2007, pp. 235–242.

[21] J. Palacios, E. Zhang, Rotational symmetry field design on surfaces, ACM Transactions on Graphics 26 (3) (2007) 55, doi:10.1145/1276377.1276446.

[22] N. Ray, B. Vallet, W.C. Li, B. Lévy, N-symmetry direction field design, ACM Transactions on Graphics 27 (2) (2008) 1–13, doi:10.1145/1356682.1356683.

[23] Y. Kim, C.H. Lee, A. Varshney, Vertex-transformation streams, Graphical Models 68 (4) (2006) 371–383.

[24] M. Kazhdan, T. Funkhouser, S. Rusinkiewicz, Shape matching and anisotropy, ACM Transactions on Graphics 23 (3) (2004) 623–629, doi:10.1145/1015706.1015770.

[25] M. Kazhdan, T. Funkhouser, S. Rusinkiewicz, Symmetry descriptors and 3D shape matching, in: SGP '04: Proceedings of the Symposium on Geometry Processing, ACM, New York, NY, USA, 2004, pp. 115–123. doi:10.1145/1057432.1057448.

[26] R. Liu, H. Zhang, A. Shamir, D. Cohen-Or, A part-aware surface metric for shape analysis, Computer Graphics Forum (Proceedings of Eurographics) 28 (2) (2009) 397–406.

[27] A. Berner, M. Bokeloh, M. Wand, A. Schilling, H.-P. Seidel, A graph-based approach to symmetry detection, in: Symposium on Volume and Point-Based Graphics, Eurographics Association, Los Angeles, CA, 2008, pp. 1–8.

[28] Z. Karni, C. Gotsman, Spectral compression of mesh geometry, in: SIGGRAPH '00: Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques, ACM Press/Addison-

Wesley Publishing Co., New York, NY, USA, pp. 279–286. doi:10.1145/344779.344924.
[29] B. Vallet, B. Lévy, Spectral geometry processing with manifold harmonics, Computer Graphics Forum (Proceedings of Eurographics) 27 (2) (2008) 251–260, doi:10.1111/j.1467-8659.2008.01122.x.
[30] P. Simari, E. Kalogerakis, K. Singh, Folding meshes: hierarchical mesh segmentation based on planar symmetry, in: SGP '06: Proceedings of the Symposium on Geometry Processing, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 2006, pp. 111–119.
[31] Z.-Q. Cheng, G. Dang, S.-Y. Jin, A meaningful mesh segmentation based on local self-similarity analysis, 10th IEEE International Conference on Computer-Aided Design and Computer Graphics, 2007, pp. 288–293, doi:10.1109/CADCG.2007.4407896.
[32] S. Yoshizawa, A. Belyaev, H.-P. Seidel, Smoothing by example: mesh denoising by averaging with similarity-based weights, in: International Conference on Shape Modeling and Applications, IEEE Computer Society, Washington, DC, USA, 2006, p. 9. doi:10.1109/SMI.2006.38.
[33] A. Khotanzad, Y.H. Hong, Rotation invariant pattern recognition using zernike moments, in: 9th International Conference on Pattern Recognition, vol. 1, 1988, pp. 326–328, doi:10.1109/ICPR.1988.28233.
[34] G. Taubin, A signal processing approach to fair surface design, in: SIGGRAPH '95: Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques, ACM, New York, NY, USA, 1995, pp. 351–358. doi:10.1145/218380.218473.
[35] C.H. Lee, A. Varshney, D.W. Jacobs, Mesh saliency, ACM Transactions on Graphics 24 (3) (2005) 659–666, doi:10.1145/1073204.1073244.
[36] J. Revaud, G. Lavoue, A. Baskurt, Improving zernike moments comparison for optimal similarity and rotation angle retrieval, IEEE Transactions on Pattern Analysis and Machine Intelligence 31 (2008) 627–636, doi:10.1109/TPAMI.2008.115.
[37] M.S. Floater, Mean value coordinates, Computer Aided Geometry Design 20 (1) (2003) 19–27, doi:10.1016/S0167-8396(02)00002-5.