




Relief Pattern Segmentation Using 2D-Grid Patches on a Locally Ordered Mesh Manifold

Claudio Tortorici¹ , Denis Vreshtazi², Stefano Berretti²  and Naoufel Werghi¹ 

¹Khalifa University, Abu Dhabi, UAE

²University of Florence, Florence, Italy

Abstract

The mesh manifold support has been analyzed to perform several different tasks. Recently, it emerged the need for new methods capable of analyzing relief patterns on the surface. In particular, a new and not investigated problem is that of segmenting the surface according to the presence of different relief patterns. In this paper, we introduce this problem and propose a new approach for segmenting such relief patterns (also called geometric texture) on the mesh-manifold. Operating on regular and ordered mesh, we design, in the first part of the paper, a new mesh re-sampling technique complying with this requirement. This technique ensures the best trade-off between mesh regularization and geometric texture preservation, when compared with competitive methods. In the second part, we present a novel scheme for segmenting a mesh surface into three classes: textured-surface, non-textured surface, and edges (i.e., surfaces at the border between the two). This technique leverages the ordered structure of the mesh for deriving 2D-grid patches allowing us to approach the segmentation problem as a patch-classification technique using a CNN network in a transfer learning setting. Experiments performed on surface samples from the SHREC'18 contest show remarkable performance with an overall segmentation accuracy of over 99%.

CCS Concepts

• *Computing methodologies* → *Shape analysis; Shape representations; Mesh geometry models;*

1. Mesh Resampling

The rationale of investigating a mesh-resampling method is three-fold: 1) Dealing with a geometric texture analysis, where texture is defined as repetitive shape patterns on the mesh, we deemed that such repetition aspect would be better represented when the mesh is regular; 2) A regular mesh resampling reduces the nuisance effects of the irregular tessellations in the computation of local surface descriptors; 3) We noticed that there is a lack in the literature of mesh resampling techniques that ensure the best trade-off between computation efficiency and preservation of the geometric texture.

We wanted to design a mesh tessellation algorithm, capable of producing an *ordered equilateral triangular mesh*, i.e., a mesh with equilateral facets ordered with respect to a seed point. We want also to keep as much as possible the geometric texture of the surface. To this end, we start by observing that the simplest geometry that can contain a close set of equilateral triangles is the *hexagon*. Such hexagon shape can be “propagated” adding a new layer of equilateral triangles of the same size. It is possible to demonstrate that each new layer, or ring of facets, that propagates the hexagon shape follows an arithmetic progression with the cardinal of the ring-facets at the ℓ^{TH} ring writing as $\#E(\ell) = 6(2\ell - 1)$.

Algorithm 1 summarizes our Circle-Surface Intersection Ordered Resampling (CSIOR) method.

Algorithm 1 CSIOR

```
1: Initialize:  
    $FacetIn \leftarrow$  First Hexagon  
2: repeat  
3:   for  $fin$  in  $FacetIn$  do  
4:      $FacetOut \leftarrow$  CIRCLE INTERSECTION( $fin$ )  
5:   end for  
6:   for  $fout$  in  $FacetOut$  do  
7:     if  $fout$  and  $fout + 1$  consecutive facets then  
8:        $FacetIn \leftarrow$  Connect  $fout$  and  $fout + 1$   
9:     end if  
10:  end for  
11: until Mesh fully resampled
```

The initial hexagon, or set of $FacetIn$, can be generated by projecting six equidistant points of a circle on the mesh surface. Once an initial hexagon composed of six ordered equilateral triangles (initial set of $FacetIn$) is defined, the process evolves iteratively in two steps: (i) *Circle Intersection* that generates new facets from the previous ring keeping shape and relief patterns in the original mesh, while propagating the order of the previous ring; and (ii) *Facet Connection* that connects the facets obtained at the first step, keeping the hexagon arithmetic progression and the facets ordering.

2. Mesh-grid and Mesh Images

With circular ordering of the triangle facets in the hexagon rings produced by the CSIOR algorithm, we can set arithmetic progression patterns allowing us to derive *mesh-grids* from the ordered structure of the mesh. To determine such a pattern, the propagated hexagon is divided into six slices, numbered from 1 to 6 starting from top-left to bottom-right, as shown in Figure 1(a). Since facets are ordered ring-wise, the grid is defined by two matrices, one for the ring number Ψ , and one for the facet location Φ within the ring; so, the pair $\langle \Psi(i, j), \Phi(i, j) \rangle$ determines the facet at position (i, j) of the grid. Observing the ordered structure, we identified similar behaviors in central slices and in the external ones: 1) In central slices (in green in Figure 1(a)), each row of the grid belongs to the same ring, thus ψ is constant at each row, while the facets location ϕ shifts by two in a counter-clockwise manner; 2) In the external slices (in yellow), each row is composed of elements from different rings, thus ψ increases by 1 going outward with respect to the seed point, while ϕ increases by 0, 6, or 12, depending on the slice, in the same direction. The final grid has a number of rows equal to $\lfloor \frac{R}{2} \rfloor$, where R is the number of hexagon rings. As shown in Figure 1(b), the pattern repeats for any number of hexagon ring, thus obtaining an image-like grid over the resampled mesh (c), without the need to compute any local or global additional ordering.

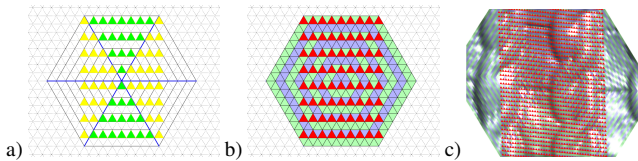


Figure 1: Image-like grid: (a) The grid pattern over the central and external slices (colored in green and yellow, respectively) of the hexagon; (b) Grid obtained over 8 rings, and (c) over the mesh of a relief pattern.

The grid-like structure on the mesh manifold allows us to extract local images using descriptors directly computed on the mesh itself. Indeed, descriptors like Gaussian/mean curvature, shape index, normal, *etc.*, can be computed at each facet of the grid, to form a kind of local geometric image. Moreover, these images can be grouped by triplets into an RGB image format, which we dubbed "mesh-image", providing thus an early-fusion format of geometric descriptors on the mesh. This 2D image representation allows also performing convolution on the mesh at different scales. Examples of mesh-images computed on the manifold are reported in Figure 2.

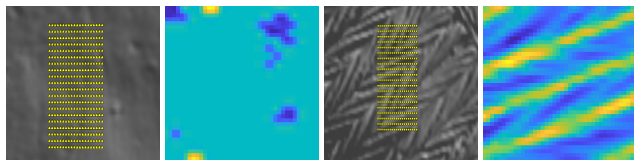


Figure 2: A mesh-grid and its associated mesh-image computed with the local depth descriptor, respectively, for a non-textured surface (left) and a textured surface instance (right).

3. Segmentation of Relief Patterns on the Mesh

Our aim here is to perform an automatic mesh surface segmentation whereby the surface is partitioned into regions exhibiting distinct geometric texture. To the best of our knowledge, this problem has not been approached before. In a first stage, we propose to segment the mesh into textured and non-textured regions. We designed a patch-based approach, where a mesh patch is represented by an $m \times m$ mesh-grid and its associated mesh-image computed at the neighbourhood of each triangle facet on the mesh. We consider each patch belonging to one of the following classes: (1) Textured patches, which represent relief patterns; (2) Non-Textured patches, which represent regions without any geometric textures; and (3) Border patches, which represent surface regions where the two previous categories meet each other.

The mesh-image representation that encodes the shape of a mesh patch in an image, allows us to classify surface patches using a Convolutional Neural Network. We performed our experiments using a pre-trained ResNet50 architecture in a transfer learning setting, where the final layers have been reshaped to the number of classes in our case. The mesh-images have been scaled to the ResNet input size of 224 by 224.

We experimented our method on the SHREC'18 dataset that includes 3D models of archaeological artifacts showing different geometric patterns over model surfaces. From this dataset, we extracted and resampled using CSIOR, ten different surfaces exhibiting textured and non-textured areas. We computed mesh-image patches for *Local Depth* (LD), *Normal Elevation* (NE), and *Normal Azimuth* (NA), surface descriptors. First, we segmented manually the mesh surfaces using the MeshLab editor, so that each triangle facet in the mesh is labeled as textured or non-textured. Then, at each facet, we extracted a mesh-grid, which we labeled as textured if the portion of textured facets is above 80%, non-textured if it is below 20%, and edge if it is in-between.

Results in Table 1 show best performance is obtained with a grid size of 32×32 and the *Normal Azimuth* (NA) surface descriptor.

Grid size	14 × 14				32 × 32			
	LD	NA	NE	All	LD	NA	NE	All
Accuracy	96.4	95.7	95.2	97.0	98.5	98.6	97.7	98.4

Table 1: Classification rate obtained with different settings.

We have also experimented a multi-oriented version of our local representation, where three mesh-grids are generated locally, one grid for each edge of a facet. A mesh image is then constructed for each of them. Using these multi-oriented images can be seen as a sort of data augmentation for the local description of the mesh. Results for the multi-oriented solution reported in Table 2 show a general improvement with a classification of over 99%.

Grid size	multi-oriented 32 × 32			
	LD	NA	NE	All
Accuracy	98.9%	99.2%	99.2%	99.2%

Table 2: Classification obtained with multi oriented 32 × 32 grids.