Optimal adaptive polygonal approximation of parametric surfaces

LUIZ VELHO[1] LUIZ HENRIQUE DE FIGUEIREDO[2]

[1]IMPA – Instituto de Matemática Pura e Aplicada
Estrada Dona Castorina 110, 22460-320 Rio de Janeiro, Brazil (lvelho@visgraf.impa.br)

[2]Computer Systems Group, Department of Computer Science, University of Waterloo
Waterloo, Ontario, Canada, N2L 3G1 (lhf@csg.uwaterloo.ca)

Abstract. We present a new method for adaptive polygonization of parametric surfaces. The method combines recursive simplicial subdivision of the domain and point sampling along curves on the surface. We avoid cracks in the polygonal mesh by determining the optimal sampling rate along the edges of a cell *before* subdividing it. The method is suitable for surfaces with low variations, such as bicubic patches, as well as for surfaces with high variations, such as height fields.

1 Introduction

The polygonization of parametric surfaces is a classical problem in computer graphics and geometric modeling that has many practical applications. The problem is computing a piecewise linear approximation for a continuous surface described by parametric functions.

A polygonal approximation is the simplest form of surface description and therefore is the representation of choice in the implementation of a large number of algorithms. Moreover, existing graphics hardware and libraries have special support for polygonal meshes, specially triangular meshes. Thus, despite the existence of more sophisticated standard forms for surface description, such as NURBS, there is always a need to convert surfaces to polygonal form.

In this paper, we describe a method that builds good polygonal approximations while keeping the number of polygons low.

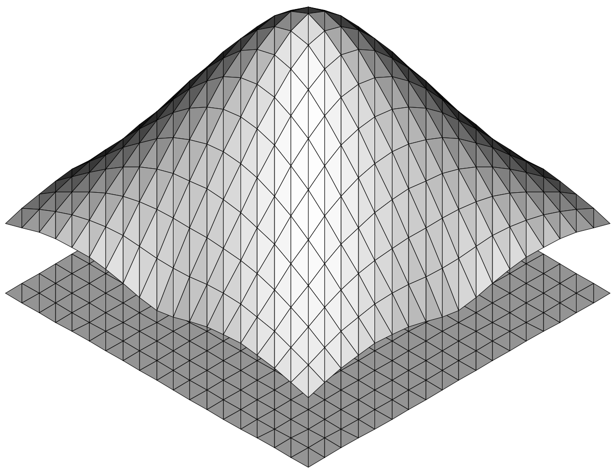

Figure 1: Polygonal approximation of a surface from uniform decomposition of its domain into triangles.

1.1 Methods for polygonal approximation

The simplest polygonization method is *uniform decomposition*: The parameter domain of the surface is decomposed into a regular rectangular grid and the surface is sampled at the nodes of this grid. The connectivity of the grid provides the structure for a quadrilateral mesh approximating the surface. Frequently, however, the rectangles in the grid are further subdivided diagonally into triangles, because triangular meshes are easier to process (Figure 1).

The main drawback of uniform decomposition is that it requires high sampling rates to approximate complex surfaces accurately. Uniform decomposition using very high sampling rates produces meshes with too many polygons. Since it is difficult to choose an adequate mesh size, except by trial and error, uniform decomposition typically produces approximations that oversample regions of low curvature and undersample regions of high curvature.

A better alternative to uniform decomposition is *adaptive decomposition*, in which the sampling rate varies across the parameter domain according to the complexity of the surface, as measured by the variation of its curvature. Adaptive decomposition methods ideally sample the domain finely in regions of high curvature and coarsely in regions of low curvature, thus producing only the minimum number of polygons required to approximate the surface within a prescribed accuracy. However, finding the *best* polygonal approximation with the *exact* minimum number of polygons required for the given accuracy is probably a very difficuly problem (i.e., NP-hard), and we must instead rely on good heuristics.

Adaptive methods must solve two fundamental problems: how to perform optimal sampling, and how to ensure global consistency. Optimal sampling guarantees a faithful and efficient geometric approximation, and is based on the adaptation criteria. Global consistency

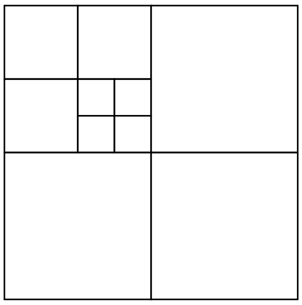

Figure 2: Quadtree domain decomposition.

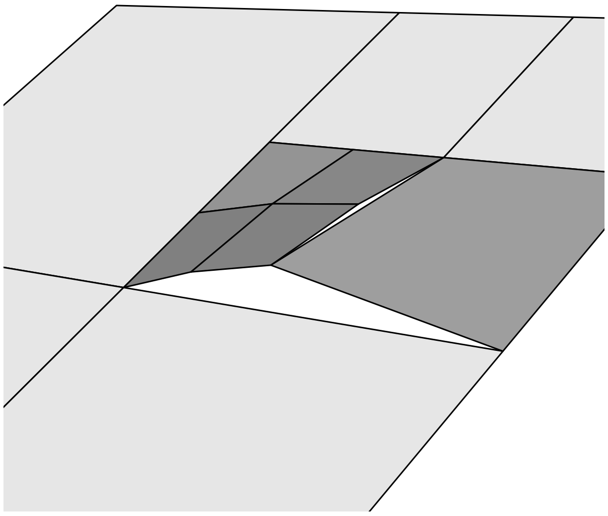

Figure 3: Surface cracks due to topological inconsistency in the quadtree decomposition shown in Figure 2 [2].

guarantees the correct topology of the polygonal mesh, and depends on the method used for structuring sample points. Adaptive polygonization algorithms can be classified according to how they solve these two problems.

1.2 Previous work

Previous algorithms for adaptive polygonal approximation of parametric surfaces perform sampling and subdivision in a single step. Surface patches are recursively subdivided by splitting edges at their midpoints and creating internal edges to connect these sample points, until the surface patches are flat within a tolerance. Figure 2 shows a typical quadtree decomposition of a rectangular domain that can result from such subdivision schemes.

The domain subdivision must be performed carefully, and well integrated with the sampling, to guarantee the global topological consistency of the polygonal mesh, and avoid the creation of *cracks*, which may appear when adjacent cells are not subdivided to the same depth. In this case, cracks show up in the surface where the piecewise approximation is not continuous (Figure 3). This problem is caused by a topologically inconsistent domain decomposition (Figure 2).

Global consistency and elimination of cracks can be achieved in several ways: by moving inconsistent vertices to flat edges, thereby guaranteeing continuity by imposing edge coherence [1]; with hierarchical spatial data structures, such as restricted quadtrees, over which further triangulation is used to force consistency [2]; or even a posteriori [3].

The main deficiency of these algorithms is in the control of the adaptation process: coupling sampling and subdivision imposes restrictions on the structure of the decomposition, often resulting in sub-optimal polygonal meshes. Moreover, recursive subdivision is controlled by flatness tests for surface patches that only exist for special classes of surfaces [4-11] (but see [12] for a general method).

2 Path-based adaptation algorithm

We now describe a *path-based* adaptive polygonization algorithm that combines adaptive curve sampling with simplicial subdivision: we use complete edge sampling to avoid cracks, and area scanning to guide the subdivision. This strategy allows better adaptation, trivially ensures global consistency, and produces meshes with an optimal number of polygons. The basic algorithm is:

1. [Initialization] Start with a coarse uniform simplicial decomposition of the parameter domain. This can be simply the subdivision of the domain along its diagonal into two triangular cells.
2. [Curve generation] Sample the edges of all cells in the initial decomposition adaptively to construct an efficient polygonal approximation of the corresponding curves on the surface.
3. [Test for flatness] For each cell, test the corresponding surface patch for flatness, based on the curvature of its edge curves.
4. [Cell subdivision] Subdivide each cell whose patches are not flat by constructing internal edges, based on the number of points on the external edge curves. Repeat the sampling in step 2 for each new internal edge.
5. [Recursion] Repeat steps 3 and 4 for each new cell.

Unlike previous methods, edge sampling is completely done at each subdivision step, while creating new edges in step 4. Further subdivisions respect this sampling. This is the key factor for optimal sampling and global consistency. Because edge curves are generated first, in a single operation, it is possible to find the minimum number of sample points that produces the desired approximation. Moreover, global consistency is automatically guaranteed, because edge curves are shared by adjacent cells. Thus, by construction, no cracks are possible.

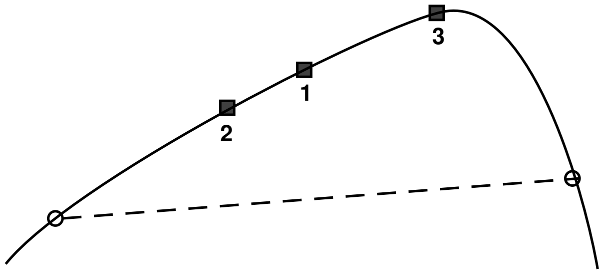

Figure 4: Stochastic search for critical points.

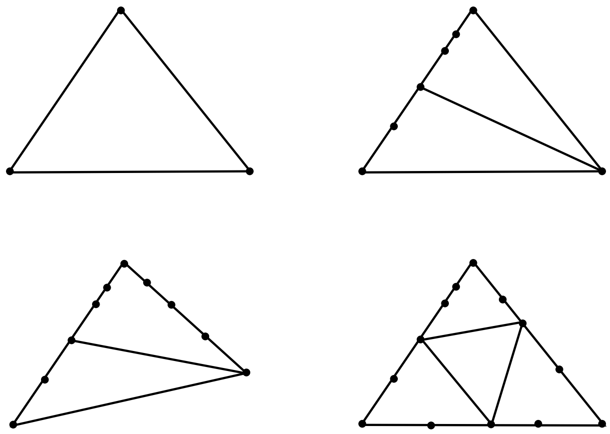

Figure 5: Four subdivision cases: three simple edges, two simple edges, one simple edge, no simple edge.

2.1 Curve generation

The edges in the domain decomposition are straight line segments, but the corresponding curves on the surface are not. The goal of the sampling in step 2 is to build a polygonal approximation of these curves that is adapted to their geometry on the surface.

There are many methods for adaptive sampling of parametric curves [10, 13-16]. We have chosen *multiple random probing*, an extension of single random probing [16], because it is easy to implement and has many degrees of freedom to help achieve a good sampling.

Adaptive sampling with multiple random probing works as follows: The curve is probed at a series of random points on the edge, starting around the midpoint with a small deviation from it, and increasing the range of deviation at each subsequent random point. At each probe, the curve is tested for flatness by checking whether the probe point lies close to the line segment defined by the two edge extremes. If the edge passes all flatness tests, then the corresponding curve on the surface is considered flat. Otherwise, the edge is split at the first probe point that fails the flatness test, and the two parts are sampled recursively. We use depth-first search, recursing on the left half first, to generate sample points in the order that they occur on the curve, and thus a correct polygonal approximation.

By varying the center and range of random perturbation for probing, we not only avoid aliasing but also choose a “good” point for splitting the edge, because, in effect, we are performing a stochastic search to find the point on the curve that lies fartherst away from the line segment connecting the edge endpoints. Therefore, there is a high probability of splitting the edge at the point of highest curvature on that segment of the curve (Figure 4). Such a subdivision scheme is specially important in the early stages of the subdivision, when the edges are long and the surface can oscillate significantly from one extreme to the other. The whole procedure can be regarded as a heuristic for finding the critical points of the curve, e.g., the points of maximum local curvature.

2.2 Cell subdivision

Cell subdivision in step 4 is determined by an analysis of the complexity of the external edge curves. A *simple edge* is composed of a single linear segment; it corresponds to a flat curve segment on the surface. Cells are subdivided by splitting complex external edges at an internal point and joining them to either the opposite vertex in the cell or to another splitting point. There are four possible cases, depending the number of simple edges in a cell (Figure 5):

1. *three simple edges*: Recursive subdivision terminates and the procedure outputs one triangle, corresponding to a flat surface patch.
2. *two simple edges*: The cell is divided into two subcells by generating a new internal edge from the complex edge to the opposite vertex.
3. *one simple edge*: The cell is divided into two subcells by generating a new internal edge from one complex edge to the opposite vertex.
4. *no simple edge*: The cell is divided into four subcells by generating new internal edges between each pair of adjacent complex edges.

The test for flatness in step 3 is simply deciding which of these four cases apply to a cell; the corresponding patch is considered flat only in case 1.

When new internal edges are to be created in cases 2, 3, and 4, we perform an exhaustive search and choose the edge that can be sampled with the least number of points. (It is in this sense that our method is an optimization method.) This local area scanning is the key factor for global optimal sampling. The motivation here is to find the best subdivision of each patch, both in terms of the accuracy of the approximation and in terms of the number of polygons needed. In cases 2 and 3, we seek the curves of lowest curvature inside the patch. In case 4, we seek a triplet of curves with total minimum curvature that fit well inside the patch.

A complete analysis of special cell subdivision methods for Bézier surfaces can be found in [7].

3 Examples

We now show four examples of the method in action. For each example, we show the parameter domain decomposition and a three-dimensional image of the corresponding polygonal approximation, rendered with Gouraud shading. In all four cases, the algorithm starts with the simplest simplicial decomposition: a rectangular domain divided along its diagonal into two triangles.

Figure 6 shows the cylinder given by:

$$x = \cos u, \quad y = \sin u, \quad z = v$$
$$u \in [0, 3.12], \quad v \in [0, 1].$$

Note how the polygonization is built along lines that are parallel to the main axis of the cylinder. These lines correspond to paths of minimum curvature on the surface, which are straight line segments in this case. The other lines—the boundaries and the main diagonal—are part of the initial coarse triangulation.

Figure 7 shows the "saddle" given by:

$$x = u, \quad y = v, \quad z = (uv)^3$$
$$u \in [0, 1], \quad v \in [0, 1].$$

Note how the polygonization adapts to the geometry of surface. The flat center is covered by only a few triangles. The curved sides form a ruled structure in which the triangles are aligned transversally to the steepest directions.

Figure 8 shows the "mountain" given by:

$$x = u, \quad y = v, \quad z = (\sin u \sin v)^4$$
$$u \in [1.5, 2.7], \quad v \in [0.75, 1.65].$$

Here we can see that the polygonization follows contour lines of the surface as in a topography map. These curves are level sets parallel to the base xy-plane. Besides their importance in geometric approximation, contour lines are perceptually insightful because they depict the surface's height variations.

Figure 9 shows the "needles" given by:

$$x = u, \quad y = v, \quad z = 0.8 \sin u \sin v$$
$$u \in [1.33, 11.33], \quad v \in [10.25, 21.25].$$

This surface has both high frequency detail and sharp variations. Even though the algorithm started with just two triangles covering the entire parametric domain, it was able to capture all important features in the surface. Moreover, the polygons are uniformly distributed around those features.

4 Discussion

The method is a heuristic optimization process that tries to minimize the number of triangular faces necessary to approximate a surface within a given tolerance. This global optimization is achieved by a combination of domain decomposition and local search; it also has a stochastic component, due to edge sampling by random probing. The method gives not only a good geometric approximation (sampling), but also a good polygonal decomposition (structuring).

As can be seen in the examples, specially in the cylinder example, the polygonal approximations are built "around" the lines of principal curvature [17]. The lines of minimal principal curvature are the union of the low curvature curves selected for cell subdivision.

In the case of height field surfaces, such as the "mountain", the adaptation tracks down the level sets (i.e., curves of same height), which are locally orthogonal to the terrain gradient.

5 Conclusion

We have presented a new, general method for adaptive polygonization of parametric surfaces. The method combines recursive simplicial subdivision of the domain with point sampling along curves on the surface. We avoid cracks in the polygonal mesh by determining the optimal sampling rate along the edges of a cell before subdividing it. The method is suitable for surfaces with low variations, such as bicubic patches, as well as for surfaces with high variations, such as height fields.

The method described here is also suitable for computing trimmed surfaces: First, sample the trimming curves as described in Section 2.1. Then, start the adaptation process from a triangulation of the trimmed domain that respects this sampling.

One drawback of the method is that currently it is slow, due to the exhaustive search for curves of low curvature inside each patch. A dynamical programming approach could probably be used to avoid recomputing candidates edges in later subdivisions, resulting in increased overall speed.

On the other hand, the cost of the method as described here may be appropriate for applications that do not change the geometry of the surface, such as high-quality rendering, specially if multi-resolution models are used. Note that multi-resolution models are obtained with no additional cost, simply by keeping the polygonal approximations corresponding to all intermediate decompositions, and using one that is suitable for a given rendering scale.

We plan to perform an experimental comparision of the performance of this method with previous methods [11, 18, 19].

Acknowledgements. The figures in Section 3 were generated with Geomview [20]. The authors are partially supported by research grants from the Brazilian Council for Scientific and Technological Development (CNPq). The second author received financial support from the University of Waterloo for presenting preliminary results of this research at the Fourth SIAM Conference on Geometric Design, in November 1995.

References

[1] J. H. Clark. A fast algorithm for rendering parametric surfaces. In K. I. Joy, C. W. Grant, N. L. Max, and L. Hatfield, editors, *Tutorial: Computer Graphics: Image Synthesis*, pages 88–93. Computer Society Press, 1988.

[2] B. Von Herzen and A. H. Barr. Accurate triangulations of deformed, intersecting surfaces. In *Computer Graphics (SIGGRAPH '87 Proceedings)*, volume 21, pages 103–110, July 1987.

[3] M. Tamminen and F. W. Jansen. An integrity filter for recursive subdivision meshes. *Computers and Graphics*, 9(4):351–363, 1985.

[4] J. Lane and L. Carpenter. A generalized scan line algorithm for the computer display of parametrically defined surfaces. *Computer Graphics and Image Processing*, 11:290–297, 1979.

[5] J. Lane and R. Riesenfeld. A theoretical development for the computer generation and display of piecewise polynomial surfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2(1):35–46, 1980.

[6] P. A. Koparkar and S. P. Mudur. Computational techniques for processing parametric surfaces. *Computer Vision, Graphics, and Image Processing*, 28(3):303–322, 1984.

[7] D. Filip. Adaptive subdivision algorithms for a set of Bézier triangles. *Computer Aided Design*, 18(2):74–78, 1986.

[8] R. D. Clay and H. P. Moreton. Efficient adaptive subdivision of Bézier surfaces. In D. A. Duce and P. Jancene, editors, *Eurographics '88*, pages 357–371. North-Holland, September 1988.

[9] D. R. Forsey and R. V. Klassen. An adaptive subdivision algorithm for crack prevention in the display of parametric surfaces. In *Proceedings of Graphics Interface '90*, pages 1–8, May 1990.

[10] M. Kosters. Curvature-dependent parametrization of curves and surfaces. *Computer Aided Design*, 23(8):569–578, 1991.

[11] J. W. Peterson. Tessellation of NURB surfaces. In P. Heckbert, editor, *Graphics Gems IV*, pages 286–320. Academic Press, 1994.

[12] S. P. Mudur and P. A. Koparkar. Interval methods for processing geometric objects. *IEEE Computer Graphics and Applications*, 4(2):7–17, 1984.

[13] U. Ramer. An iterative procedure for the polygonal approximation of plane curves. *Computer Graphics and Image Processing*, 1:244–256, 1972.

[14] M. Crampin, R. Guifo Guifo, and G. A. Read. Linear approximation of curves with bounded curvature and a data reduction algorithm. *Computer Aided Design*, 17(6):257–261, 1985.

[15] R. E. Chandler. A recursive technique for rendering parametric curves. *Computers and Graphics*, 14(3/4):477–479, 1990.

[16] L. H. de Figueiredo. Adaptive sampling of parametric curves. In A. Paeth, editor, *Graphics Gems V*, pages 173–178. Academic Press, 1995.

[17] M. do Carmo. *Differential Geometry of Curves and Surfaces*. Prentice Hall, 1976.

[18] V. Vlassopoulos. Adaptive polygonalization of parametric surfaces. *The Visual Computer*, 6(5):291–8, 1990.

[19] S. Z. Li. Adaptive sampling and mesh generation. *Computer Aided Design*, 27(3):235–240, 1995.

[20] Software written at the Geometry Center, University of Minnesota. Available at `http://www.geom.umn.edu/software/`.

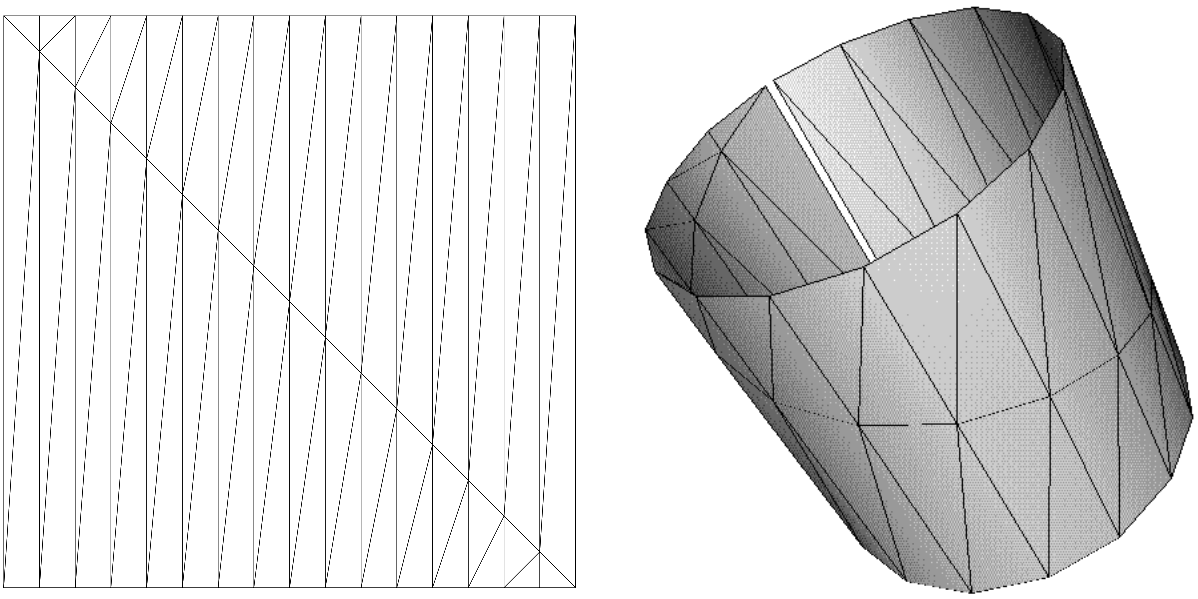

Figure 6: Domain decomposition (left) and polygonal approximation (right) for the cylinder given by $(\cos u, \sin u, v)$, where $u \in [0, 3.12]$ and $v \in [0, 1]$.

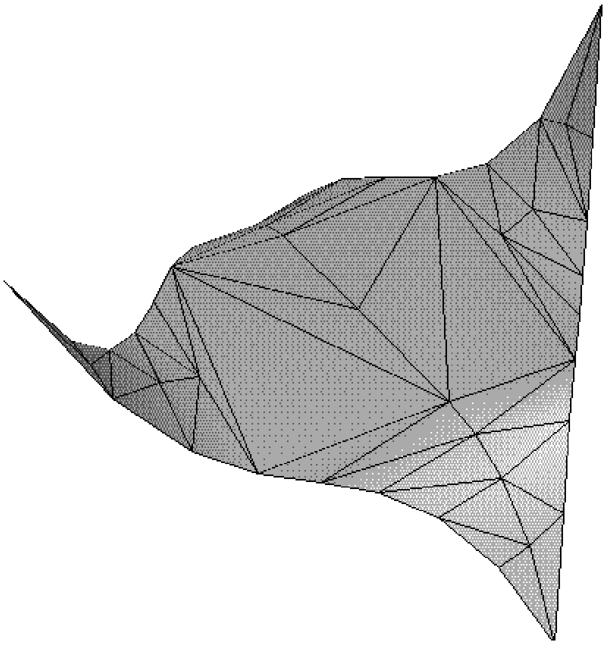

Figure 7: Domain decomposition (left) and polygonal approximation (right) for the saddle given by $(u, v, (uv)^3)$, where $u \in [0, 1]$ and $v \in [0, 1]$.

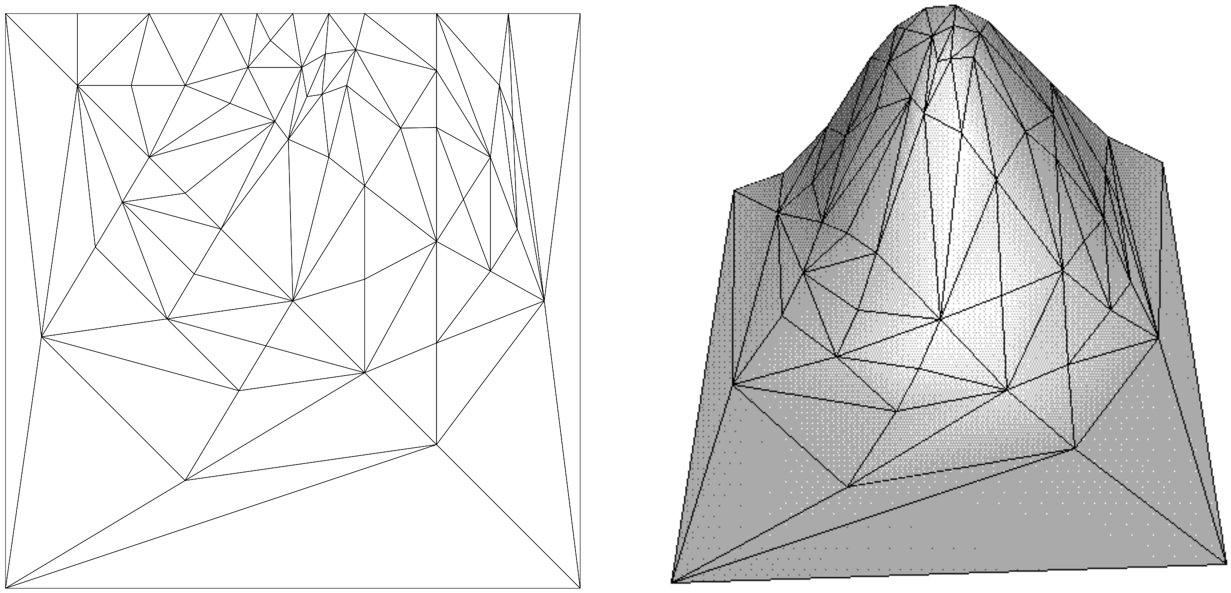

Figure 8: Domain decomposition (left) and polygonal approximation (right) for the mountain given by $(u, v, (\sin u \sin v)^4)$, where $u \in [1.5, 2.7]$ and $v \in [0.75, 1.65]$.

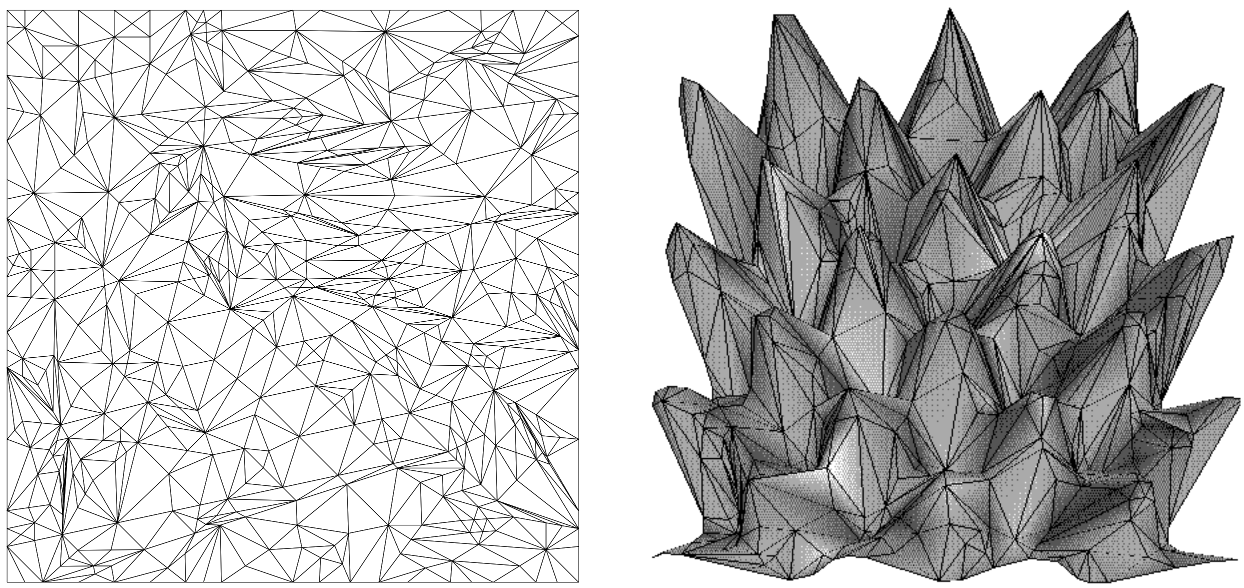

Figure 9: Domain decomposition (left) and polygonal approximation (right) for the needles given by $(u, v, 0.8 \sin u \sin v)$, where $u \in [1.33, 11.33]$ and $v \in [10.25, 21.25]$.