Parameterization of Triangular Meshes with Virtual Boundaries

Yunjin Lee[1,*] Hyoung Seok Kim[2,†] Seungyong Lee[1,‡]

[1]Department of Computer Science and Engineering
Pohang University of Science and Technology (POSTECH)
Pohang, 790-784, Korea

[2]Department of Multimedia Engineering
Dongeui University
Pusan, 614-714, Korea

Abstract

Parameterization of a 3D triangular mesh is a fundamental problem in mesh processing, such as texture mapping, multiresolution modeling, and smooth surface fitting. The convex combination approach is widely used for parameterization because it has good properties such as fast computation and little distortion of embedded triangles. However, the approach has one drawback: most boundary triangles have high distortion in the embedding compared with interior ones. In this paper, we present an extension of the convex combination approach that resolves the drawback by using a virtual boundary. We construct a virtual boundary near the real boundary of the 3D mesh and fix the virtual boundary onto a given convex polygon. Due to the virtual boundary, the real boundary triangles can better reflect the shape of the corresponding 3D triangles. We also make the shape of the 2D boundary closer to that of the original 3D meshes by projecting the 3D boundary onto an appropriate 2D plane and obtaining its convex hull. With the proposed approach, we can obtain a parameterization of a 3D mesh with less distortion than with the original convex combination approach.

1 Introduction

Parameterization of a 3D triangular mesh is a fundamental process for many applications of meshes, such as texture mapping [7, 8, 6, 10, 9, 5], multi-resolution modeling [1, 4], and smooth surface fitting [2, 3, 12]. In parameterization, a 2D coordinate (s, t) is assigned to each point on the surface of a mesh. This process is accomplished in two steps. The first step is to determine the parameter of each vertex of the mesh such that the topology of the mesh is preserved, i.e., the triangles in the parameter space must not overlap each other. The second step is to find a piecewise linear

*jin@postech.ac.kr, http://home.postech.ac.kr/∼jin
†hskim@dongeui.ac.kr, http://www.dongeui.ac.kr/∼hskim
‡leesy@postech.ac.kr, http://www.postech.ac.kr/∼leesy

function that linearly maps each parameter triangle to the corresponding triangles in the 3D mesh. The procedure of mesh parameterization can be regarded as embedding a 3D mesh into a 2D parameter space. In general, the mesh cannot be flattened over a plane without distortion [7, 8]. In many applications using parameterization, it is important to minimize distortion, which is the primary objective of parameterization. One-to-one mapping is also a very important property of parameterization.

In mesh parameterization, two paradigms exist: energy functional minimization and the convex combination approach. For the first paradigm, several approaches have been developed to define and minimize an energy functional that measures the distortion in the embedded mesh. Maillot *et al*. proposed a method to minimize a norm of the Green-Lagrange deformation tensor based on elasticity theory [8]. The harmonic embedding used by Eck *et al*. minimizes the metric dispersion instead of elasticity [1]. Lévy *et al*. proposed an energy functional minimization method based on orthogonality and homogeneous spacing [6]. A non-deformation criterion (i.e., Dirichlet energy per parameter area) is introduced in [3] with extrapolating capabilities. However, minimizing these energy functionals generally requires a non-linear solver, which may require extensive computation.

The convex combination approach is an extension of the barycentric mapping approach proposed by Tutte [13]. This approach obtains parameterization by fixing the boundary vertices of a 3D mesh onto a 2D convex polygon and solving a linear system to determine the 2D embedded positions of interior vertices. The linear system is constructed by representing each interior vertex as a convex combination of its neighborhood. In this approach, the major problem is how to determine the coefficients of the convex combination for each interior vertex. Floater proposed shape-preserving parameterization [2], where the coefficients are determined by using conformal mapping and barycentric coordinates. The harmonic embedding [1] is also a special case of this approach, except that the coefficients may be negative. The convex combination approach to parameterization is simple and fast, and always creates a one-to-one embedding. However, this approach requires the boundary of a mesh to be fixed on a convex polygon, which causes high distortion near the boundary.

On the other hand, if we reflect the 3D boundary shape of a mesh in determining the 2D boundary in the 2D parameter space, we can decrease the distortion near the boundary of an embedding. Most parameterization methods based on optimization consider all vertices together and obtain a 2D embedding that reflects the 3D boundary shape of a mesh. However, these methods usually require solving non-linear systems, which makes them relatively slow. Lévy proposed a linear system solver for parameterization, which minimizes the gradient of parameters between two adjacent faces [5]. In his approach, the boundary of an embedding needs not be fixed in the parameterization process, but the generated embedding may not be one-to-one.

In this paper, we propose a new parameterization method, which extends the convex combination approach to reduce distortion near the boundary. To overcome the limitation of fixing the boundary in the convex combination approach, we assume a set of virtual vertices attached to the boundary of a mesh and consider it as the fixed part in the parameterization process. That is, we map the virtual vertices to a 2D convex polygon instead of the real boundary vertices of a mesh. Since our method is based on the convex combination approach, it can guarantee a one-to-one mapping to be generated by solving a linear system. The proposed method is simple and fast, and it can generate an embedding with less distortion for a mesh with a complicated boundary than the original approach.

The organization of this paper is as follows. Section 2 introduces the basic idea of the proposed method with a summary of the convex combination approach and the notion of a virtual boundary. Section 3 explains the construction of a virtual boundary and how to determine the coefficients of virtual vertices. Experimental results are shown in Section 4. We give our conclusions in Section 5.

2 Basic Idea

2.1 Parameterization based on convex combinations

Floater proposed the convex combination approach to parameterization [2]. Let $\mathbf{u}_1, \ldots, \mathbf{u}_N$ be the 2D embedded positions of 3D vertices, $\mathbf{v}_1, \ldots, \mathbf{v}_N$, where $\mathbf{v}_1, \ldots, \mathbf{v}_n$ and $\mathbf{v}_{n+1}, \ldots, \mathbf{v}_N$ are the interior and boundary vertices of a mesh, respectively. The convex combination approach determines the values of $\mathbf{u}_{n+1}, \ldots, \mathbf{u}_N$ by mapping the 3D boundary onto a given convex polygon in the 2D parameter space. To obtain the values of $\mathbf{u}_1, \ldots, \mathbf{u}_n$, the approach represents $\mathbf{u}_i$ as a convex combination of $\mathbf{u}_j$, where $\mathbf{v}_j$ are the 1-ring neighborhood vertices of $\mathbf{v}_i$. That is,

$$\mathbf{u}_i = \sum_{j=1}^{N} \lambda_{i,j}\mathbf{u}_j, \;\; i = 1, \ldots, n, \quad \lambda_{i,j} > 0, \;\; (i,j) \in E, \quad \lambda_{i,j} = 0, \;\; (i,j) \neq E, \quad \sum_{j=1}^{N} \lambda_{i,j} = 1, \tag{1}$$

where E is the edge set of the mesh. We can compute the values of $\mathbf{u}_1, \ldots, \mathbf{u}_n$ by solving the linear system in Eq. (2), which is derived from Eq. (1).

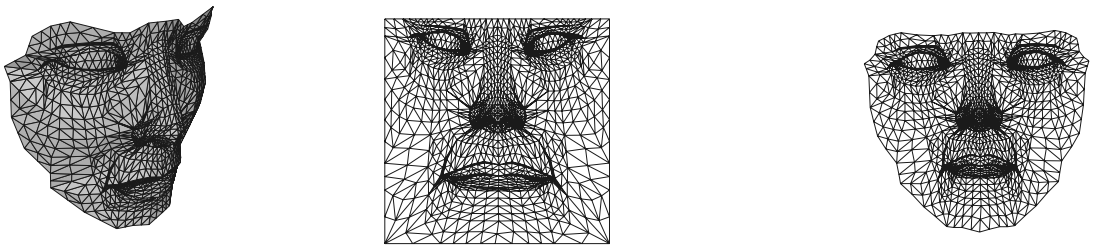
$$\mathbf{u}_i - \sum_{j=1}^{n} \lambda_{i,j}\mathbf{u}_j \;\; = \sum_{j=n+1}^{N} \lambda_{i,j}\mathbf{u}_j, \;\; i = 1, \ldots, n. \tag{2}$$

Floater proved that there always exists a unique solution of the linear system in Eq. (2). He also proved that if mapping between the 3D boundary and the 2D convex polygon is one-to-one, mapping for the interior vertices becomes an embedding without overlaps. The shape of the embedding depends on the coefficients $\lambda_{i,j}$ of the convex combination. Floater proposed three methods to obtain the coefficients: uniform parameterization, chord length

parameterization, and shape-preserving parameterization. Among them, the last method best reflects the shape of a mesh since the method has the affine invariant property.

2.2 Virtual boundary

The major problem of parameterization of a 3D triangular mesh concerns how to embed a mesh onto a 2D parameter space so that the shapes of triangles can be well preserved. Fig. 1(b) is the 2D embedding of the 3D mesh shown in Fig. 1(a) using shape-preserving parameterization [2]. We can ascertain the drawback of the convex combination approach such that high distortion occurs near the boundary and the overall shape of the embedding resembles the shape of the bounding convex polygon, in this case, a square.



(a) 3D triangular mesh (b) shape-preserving parameterization (c) our method with a virtual boundary

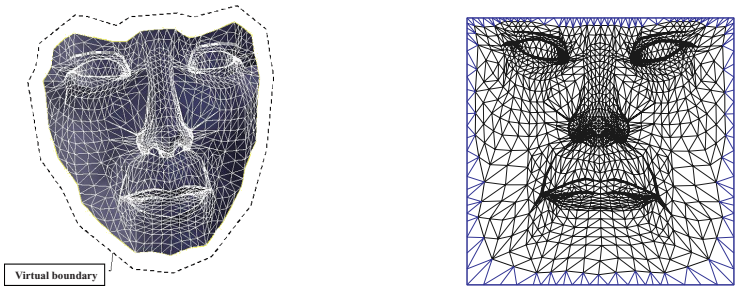
Figure 1: Parameterization of a triangular mesh

Fig. 1(c) is the embedding obtained by the proposed parameterization method. The method combines the concept of a virtual boundary to the convex combination approach and selects a bounding convex polygon that resembles the shape of a 3D boundary. The distortion of triangles near the boundary is much reduced and the overall shape of embedding is closer to that of the 3D mesh.

2.3 Parameterization process

In the proposed method, the parameterization process can be summarized as follows (see Fig. 3).

1. Virtual vertices are attached to the real boundary without 3D geometric information, as shown in Fig. 2(a).
2. The coefficients of convex combinations are computed for interior vertices and real boundary vertices.
3. The shape of the bounding convex polygon is determined in the parameter space, and the virtual boundary is mapped onto the polygon, as shown in Fig. 2(b).
4. The linear system from the convex combinations is solved to determine the embedded positions of the vertices.



(a) mesh and virtual boundary (b) virtual boundary on the parameter space

Figure 2: Virtual boundary in 3D and 2D space

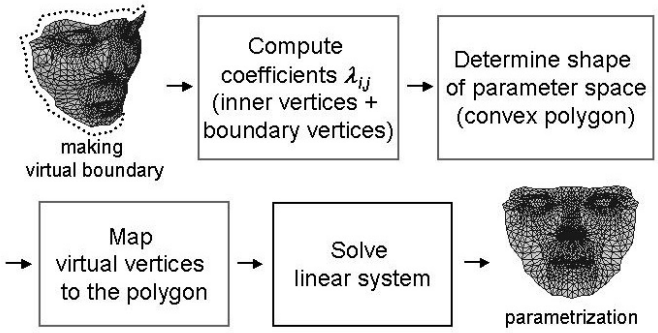


Figure 3: Parameterization process

3 Generation of Virtual Boundary

3.1 Virtual vertices

In this section, we explain how to assign topological information to virtual vertices and how to embed them onto a given bounding convex polygon. The connectivity of the virtual boundary and the real boundary is illustrated in Fig. 4. The vertices on the real boundary and the virtual boundary are denoted by v_i and v'_i, respectively. The number of virtual vertices is twice that of the real boundary vertices, with which we expect the real boundary vertices to move more freely. Each real boundary vertex v_i is adjacent to three virtual vertices, v'_{2i-1}, v'_{2i}, and v'_{2i+1}.

In the convex combination approach, the vertices on the boundary are parameterized on the bounding convex polygon by considering the edge length between two adjacent vertices. In the case of the virtual boundary, we do not have the 3D positions of the virtual vertices. Hence, we map the virtual vertices onto the convex polygon by using edge lengths between the real boundary vertices. We place the vertices v'_i so that the lengths of the edges between

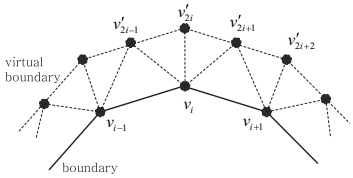


Figure 4: Connectivity between vertices on virtual and real boundary

v'_{2i} and v'_{2i+2} are proportional to the distances between v_i and v_{i+1}, and v'_{2i+1} is the middle point of v'_{2i} and v'_{2i+2}.

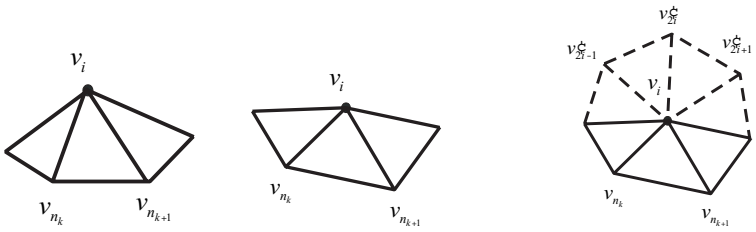
3.2 Coefficient computation

We use shape-preserving parameterization in the coefficient computation process. The coefficients are computed from the local parameterization of a vertex and its 1-ring neighborhood. The conformal mapping is used for local parameterization, and the coefficients are acquired by averaging the barycentric coordinates. Refer to [2] for the details.

In the case of our augmented mesh, we should compute the coefficients for the real boundary vertices as well as the interior ones. Note that the virtual vertices, not the real boundary vertices, are fixed onto the given bounding convex polygon. In the local parameterization of a real boundary vertex, we cannot directly use conformal mapping because the virtual vertices have no geometry. To preserve the local shape of the boundary, we first map a real boundary vertex and its real neighbors onto the 2D parameter space. They can be mapped onto the 2D space without distortion, while the angles and the lengths of the real triangles are preserved. Figs. 5(a) and 5(b) show 3D configuration and 2D mapping of a real boundary vertex and its neighbors, respectively. We then place the virtual vertices v'_{2i-1}, v'_{2i}, and v'_{2i+1} as shown in Fig. 5(c). Here, the distances between v_i and v'_j, $j = 2i-1, 2i, 2i+1$, are the same as the average of lengths of the real edges in the 1-ring neighborhood. The angles around v_i of the virtual triangles are the same as $(2\pi - \theta)/k$, where θ is the sum of angles around v_i of the real triangles and k is the number of the virtual triangles. Finally, we compute the coefficients for the boundary vertices by averaging the barycentric coordinates.

3.3 Shape of virtual boundary

We can use various 2D polygons for the bounding convex polygon, onto which the virtual boundary of a 2D mesh is fixed. Figs. 6(a) and 6(b) show the embedding results obtained when the bounding polygon is a circle and a square, respectively. The examples show that we can obtain better results than Floater's method [2] because the virtual

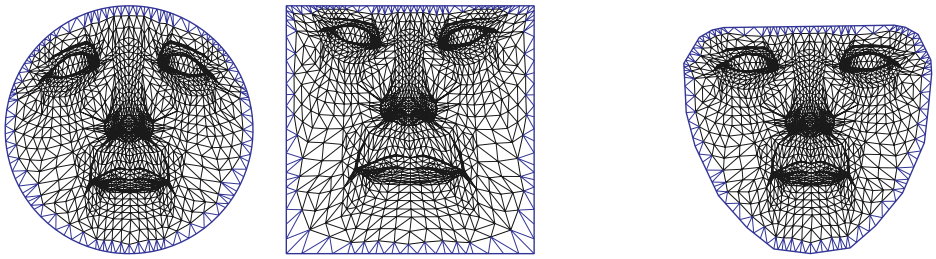


(a) 3D configuration (b) 2D mapping (c) placement of virtual vertices

Figure 5: Local parameterization of a real boundary vertex and its 1-ring neighbors

boundary is fixed instead of the real boundary. However, the shape of an embedded mesh is still strongly influenced by that of the bounding convex polygon. The overall shapes of the real boundary in Figs. 6(a) and 6(b) are close to the circle and the square, respectively.

In Fig. 6(c), the bounding convex polygon is generated from a projected boundary of the mesh. We project the real boundary onto the plane obtained by a least square fitting. Since the projected boundary may be concave, we use the convex hull of the projected boundary as the bounding convex polygon. The embedding result is better than others since the convex hull reflects the 3D geometry of the mesh.



(a) circle (b) square (c) convex hull of the projected boundary

Figure 6: Embedding results with different bounding convex polygons

3.4 Multi-layered virtual boundary

Although we can reduce the distortion near the boundary by using virtual vertices, the shape of the 2D embedding still remains under the strong influence of the bounding convex polygon. In Fig. 6, we observe that the triangle shapes in a region far from the boundary are well preserved. Hence, we try to add more virtual vertices in a multi-layered structure in order to reduce the influence of the bounding polygon. To prevent exponential increase of the number of virtual vertices, each layer is made to contain the same number of virtual vertices. Fig. 7 shows the connectivity of virtual vertices between layers.

Contrary to the single-layered structure, we must compute the coefficients of the convex combinations for virtual vertices, except at the last layer, since their positions are not determined in the mapping of the virtual boundary onto the bounding convex polygon. For simplicity, the coefficients are set to be uniformly $1/d_i$, where d_i is the degree of a virtual vertex v_i. For example, in the case of v'_{2i} in Fig. 7, the coefficients are 1/6.

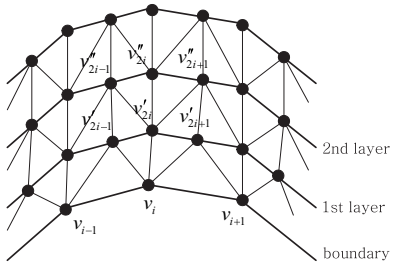
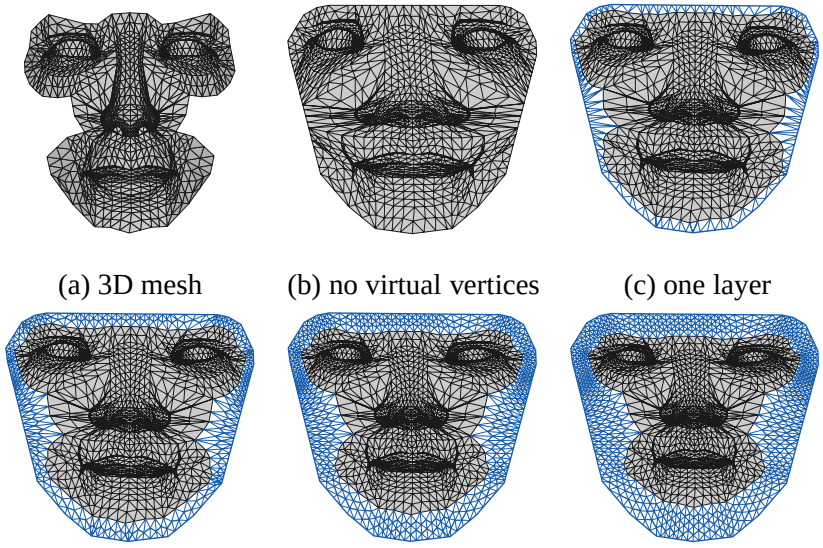


Figure 7: Construction of a multi-layered virtual boundary

Fig. 8 shows the results of our parameterization using the multi-layered virtual boundaries. When the 3D boundary severely runs in and out, as shown in Fig. 8(a), the convex hull of the projected boundary differs considerably from the shape of the 3D boundary. Fig. 8(b) shows the result of 2D embedding when the real boundary is fixed to the convex hull. Due to the concave part of the 3D boundary, the triangles around the nose are stretched. However, the distortion is reduced when we add virtual vertices in a multi-layered structure to the real boundary, as shown in Figs. 8(c) through 8(f).



(a) 3D mesh (b) no virtual vertices (c) one layer

(d) two layers (e) three layers (f) four layers

Figure 8: Effects of multi-layered virtual boundaries

4 Experimental Result

Table 1 shows the comparison of five cases: (1) the real boundary is fixed on a square, (2) the real boundary is fixed on a circle, (3) the virtual boundary is fixed on a square, (4) the virtual boundary is fixed on a circle, (5) the virtual boundary is fixed on the convex hull of the projected boundary. We use the texture stretch metric L_2 and L_∞ defined in [11] as the measures for comparison. The norm L_2 corresponds to the mean stretch over all directions, and the worst-case norm L_∞ is the greatest stretch. From Table 1, we can see that the distortion measurement changes according to the shape of the bounding polygon, and the value decreases when the shape of the bounding polygon is closer to the shape of the 3D boundary.

Shape of Boundary , Type (No. of layers)	L_2	L_∞
square, real (0)	1.1555540562	4.5632829666
circle, real (0)	1.0796248913	5.3720154762
square, virtual (1)	1.1166115999	4.4459357262
square, virtual (2)	1.1067023277	4.3346028328
square, virtual (3)	1.1003003120	4.2538571358
circle, virtual (1)	1.0660164356	5.0777935982
circle, virtual (2)	1.0637238026	4.9618878365
circle, virtual (3)	1.0634766817	4.7662897110
convex polygon, virtual (1)	1.0613353252	4.0639147758

Table 1: Comparison of distortion measurements

Fig. 10 shows the results of texture mapping, where a check-board pattern is mapped onto the mesh in Fig. 9. In the case of a square boundary without virtual vertices, the texture is severely stretched near the boundary, as shown in Fig. 10(a). Figs. 10(b) and 10(c) are the results of circle and polygon boundaries with virtual vertices, respectively. Both have a smaller stretch than Fig. 10(a). However, in Fig. 10(c), the texture is more uniformly mapped than in Fig. 10(b).

Fig. 11 shows an example of parameterizing a more complicated mesh. Fig. 11(a) is the 3D mesh. Figs. 11(b), 11(c), and 11(d) are the embeddings of Fig. 11(a) obtained by fixing the virtual boundary onto a square, a circle, and the convex hull of the projected boundary, respectively. In the figures, we can observe that the embeddings generated by adding virtual vertices have better shapes compared with those obtained by the original convex combination

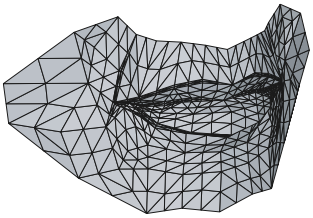
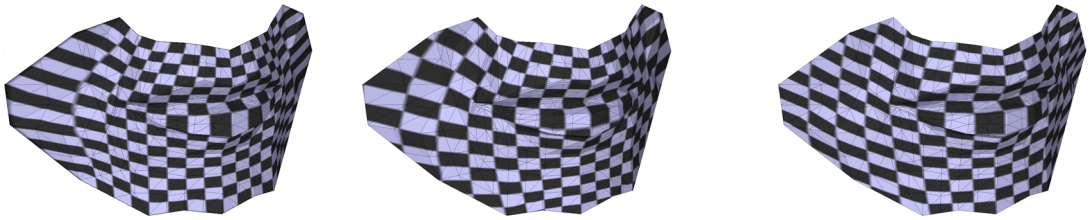


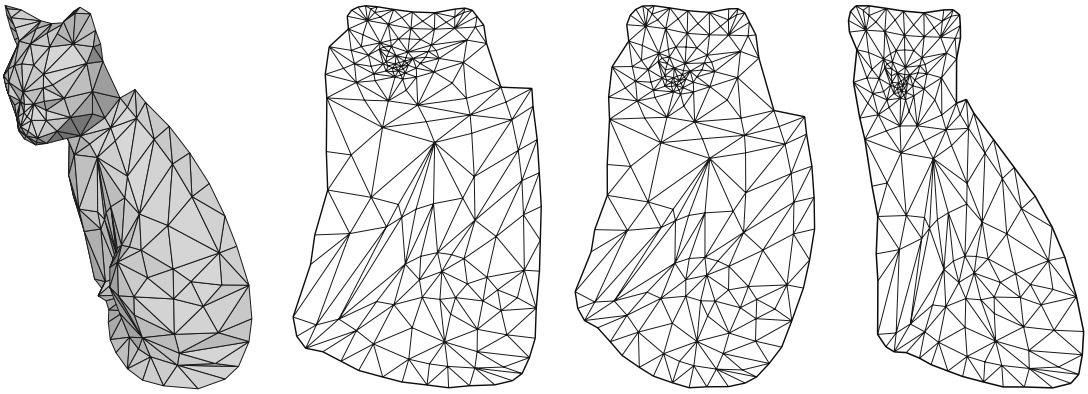
Figure 9: 3D triangular mesh



(a) square (b) circle, three virtual layers (c) convex polygon, one virtual layer

Figure 10: Texture mapping results

approach.



(a) 3D triangular mesh (b) square (c) circle (d) convex hull

Figure 11: Parameterizations with different bounding polygons

5 Conclusion

In this paper, we have proposed an extension of the convex combination approach to reduce the high distortion around the boundary in the parameterization of a 3D mesh. We introduced the virtual boundary of a mesh and fixed it instead of the real boundary to the bounding convex polygon. The multi-layered structure is useful for a mesh with a severely concave boundary. In future, we will analyze the effects of connectivity and coefficients of virtual vertices

on the result of the parameterization. In addition, we will also investigate a speed up technique with a multi-level approach to solve the linear system, such as multi-grid relaxation.

Acknowledgments

The cat model in Fig. 11 is acquired from the 3DCAFE's webpage (http://www.3dcafe.com). This work was supported by the Ministry of Education of Korea through the Brain Korea 21 program.

References

[1] ECK, M., DEROSE, T., DUCHAMP, T., HOPPE, H., LOUNSBERY, M., AND STUETZLE, W. Multiresolution analysis of arbitrary meshes. *ACM Computer Graphics (Proc. of SIGGRAPH '95)* (1995), 173–182.

[2] FLOATER, M. S. Parametrization and smooth approximation of surface triangulations. *Computer Aided Geometric Design 14* (1997), 231–250.

[3] HORMANN, K., AND GREINER, G. MIPS: An efficient global parametrization method. *ACM Computer Graphics (Proc. of SIGGRAPH '99)* (1999).

[4] LEE, A. W. F., SWELDENS, W., SCHRÖDER, P., COWSAR, L., AND DOBKIN, D. Maps: Multiresolution adaptive parameterization of surfaces. *ACM Computer Graphics (Proc. of SIGGRAPH '98)* (1998), 95–104.

[5] LÉVY, B. Constrained texture mapping for polygonal meshes. *ACM Computer Graphics (Proc. of SIGGRAPH 2001)* (2001).

[6] LÉVY, B., AND MALLET, J.-L. Non-distorted texutre mapping for sheared triangulated meshes. *ACM Computer Graphics (Proc. of SIGGRAPH '98)* (1998), 343–352.

[7] MA, S. D., AND LIN, H. Optimal texture mapping. *EUROGRAPHICS '88* (September 1988), 421–428.

[8] MAILLOT, J., YAHIA, H., AND VERROUST, A. Interactive texture mapping. *ACM Computer Graphics (Proc. of SIGGRAPH '93)* (1993), 27–34.

[9] PIPONI, D., AND BORSHUKOV, G. Seamless texture mapping of subdivision surfaces by model pelting and texture blending. *ACM Computer Graphics (Proc. of SIGGRAPH 2000)* (2000), 471–478.

[10] PRAUN, E., FINKELSTEIN, A., AND HOPPE, H. Lapped textures. *ACM Computer Graphics (Proc. of SIGGRAPH 2000)* (2000), 465–470.

[11] SANDER, P. V., SNYDER, J., GORTLER, S. J., AND HOPPE, H. Texture mapping progressive meshes. *ACM Computer Graphics (Proc. of SIGGRAPH 2001)* (2001).

[12] SHEFFER, A., AND DE STURLER, E. Surface parameterization for meshing by triangulation flattening. *Proceddings, 9th International Meshing Roundtable, Sandia National Laboratories* (October 2000), 161–172.

[13] TUTTE, W. How to draw a graph. *Proc. London Math. Soc. 13* (1963), 743–768.