Offset Triangular Mesh Using the Multiple Normal Vectors of a Vertex

Su-Jin Kim[1], Dong-Yoon Lee[2] and Min-Yang Yang[3]

[1]Korea Advanced Institute of Science and Technology, sujinkim@kaist.ac.kr
[2]Korea Advanced Institute of Science and Technology, yunny@kaist.ac.kr
[3]Korea Advanced Institute of Science and Technology, myyang@kaist.ac.kr

ABSTRACT

This paper introduces and illustrates the results of a new method for offsetting triangular mesh by moving all vertices along the multiple normal vectors of a vertex. The multiple normal vectors of a vertex are set the same as the normal vectors of the faces surrounding the vertex, while the two vectors with the smallest difference are joined repeatedly until the difference is smaller than allowance. Offsetting with the multiple normal vectors of a vertex does not create a gap or overlap at the smooth edges, thereby making the mesh size uniform and the computation time short. In addition, this offsetting method is accurate at the sharp edges because the vertices are moved to the normal directions of faces and joined by the blend surface. The method is also useful for rapid prototyping and tool path generation if the triangular mesh is tessellated part of the solid models with curved surfaces and sharp edges. The suggested method and previous methods are implemented on a PC using C++ and illustrated using an OpenGL library.

Keywords: Offset, Triangular mesh, Multiple normal vectors

1. INTRODUCTION

Offsets are widely used in tool path generation for numerical control machining, rapid prototyping, hollow or shelled model generation, and access space representations in robotics. In a numerical control machining area, 2D and 3D offsets are particularly important and useful for gouge-free and collision-free tool paths [1,2]. In 2D milling, a contour is offset by the size of the cutter radius, and invalid loops are removed for a gouge-free tool path [1]. In 3D milling, a gouge can be removed by computing the cutter location (CL) surface [2].

Since the SLT file format is widely used in the field, the development of an effective offsetting method for triangular mesh is important for numerical control machining and rapid prototyping. To offset triangular mesh, each triangular face is moved by the size of the cutter radius in its corresponding normal direction, which is an exact offset of polyhedron [3]. This process, however, results in intersections or gaps between the offset surfaces of two neighboring triangles, as shown in Fig. 1(a). The problem can be avoided if the vertices, instead of the triangular faces, are offset in their normal direction, as shown in Fig. 1(b) [4,5]. Calculating vertex offset vectors by averaging the normal vectors of triangles connected to the vertex is an inaccurate method for vertices on sharp edges [4]. Thus, a method was developed to calculate offset vectors for vertices by using the weighted sum of the normal vectors of the connection triangles [5]. The vertex offset method works well for small offset values on mesh with smooth edges where local and global intersections do not normally occur. However, the weighted sum of normal vectors at sharp edges is so large that unwanted interference can occur and the shell thickness becomes larger than the smooth edges at the rapid prototyped part.

The triangular mesh offset methods that move faces along the face normal vectors result in intersections or gaps among offset faces; other methods that move the vertices along the single normal direction of vertices are not precise at sharp edges. Consequently, a new offset method for triangular mesh is needed to make no gaps or overlaps between the triangular faces tessellated from the smooth surface and the precisely offset sharp edges.

In this paper, a new offset method for triangular mesh is introduced that moves the vertex to the multiple normal vectors of a vertex computed by the normal vectors of the faces surrounding the vertex. The vertices are moved along the multiple normal directions of a vertex and the gaps at sharp edges along with the vertices are filled by a blending mesh, as shown in Fig. 1(c). The multiple normal vectors of a vertex are introduced and the computation methods that use the normal vectors of the

triangular faces connected to the vertex is explained. The offsetting process of a triangular mesh that uses the multiple normal vectors of a vertex is detailed to create cutter location mesh.

2. MULTIPLE NORMAL VECTORS

The normal vector of a vertex in triangular mesh has been defined to a single vector and the offset method has been using the normal of faces or the single normal vector of a vertex. In this section, the multiple normal vectors of a vertex are introduced and the computation methods that use the normal vectors of the triangular faces connected to the vertex is explained.

To offset triangular mesh, which is an approximation of a solid model with piecewise smooth surfaces and sharp edges, a vertex has no single normal vector but more than one vertex normal vector referred to in this paper as the multiple normal vectors of a vertex. The vertex on smooth geometries has one normal vector calculated by the weighted sum of the normal vectors of the faces around the vertex, while the vertex on the sharp edge has more than two normal vectors that are almost the same as the normal vectors of surrounding faces.

To compute the multiple normal vectors of a vertex, the normal vectors of a vertex are set the same as the face normal vectors surrounding the vertex and replaced to one vector if the cross product of vectors is smaller than allowance. In Fig. 2(a), the vertex normal vector $\mathbf{v}_{1,2,3}$ is computed by averaging the three normal vectors of the faces around the vertex because all the cross products of the normal vectors of the faces are smaller than allowance. In Fig. 2(b), the vertex normal vector $\mathbf{v}_1$ is set the same as the face normal vector $\mathbf{f}_1$ because the direction differs from the other normal vectors of the faces around the vertex. Because the cross product between $\mathbf{f}_2$ and $\mathbf{f}_3$ is smaller than allowance, the vertex normal vectors $\mathbf{v}_2$ and $\mathbf{v}_3$ are replaced to the vector $\mathbf{v}_{2,3}$ which is the average of the two vectors. Vector $\mathbf{v}_{4,5}$ is also the average of vectors $\mathbf{f}_4$ and $\mathbf{f}_5$, which have almost the same values. The three vectors $\mathbf{v}_1$, $\mathbf{v}_{2,3}$ and $\mathbf{v}_{4,5}$, called the multiple normal vectors of a vertex, are computed from five normal vectors of the faces around the vertex. The computation method for the average vector was developed so that the distance from any original triangular surface to the offset surface would be exactly the same as the offset distance, as shown in Fig. 2(c) [5].

$$\mathbf{v}_{1,2} = w_1\mathbf{v}_1 + w_2\mathbf{v}_2 \text{ if } |\mathbf{v}_1 \times \mathbf{v}_2| < \delta \quad (1)$$

$\mathbf{v}_i$ is the normal vector of vertex Vi

δ the allowance

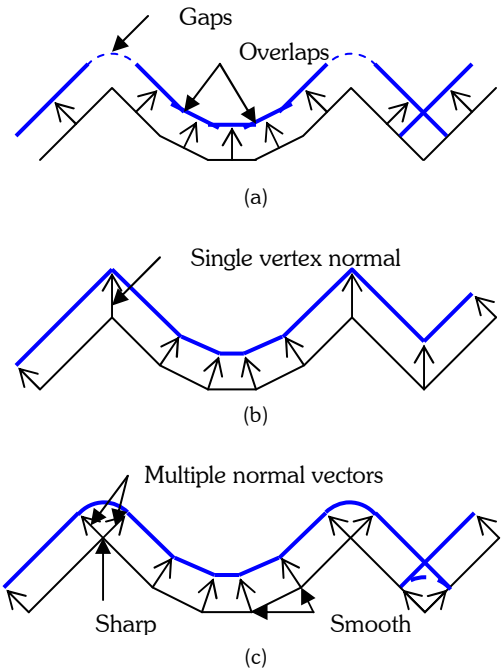
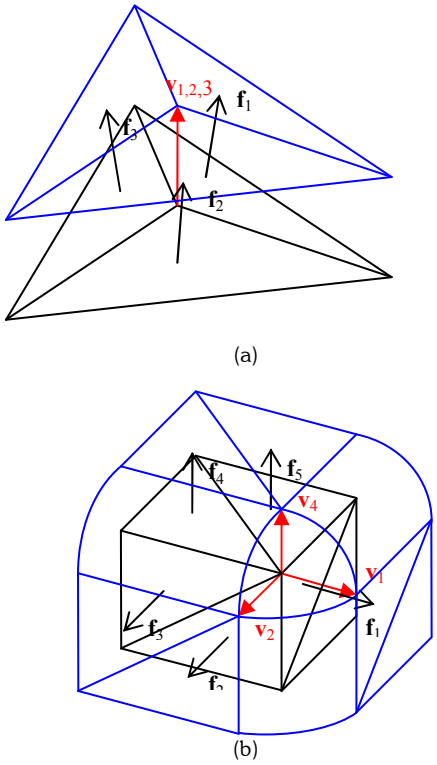
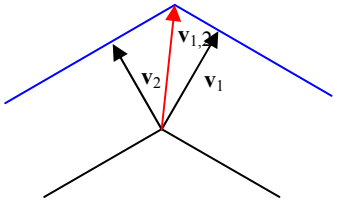


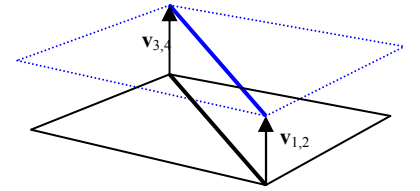
Fig. 1. Offsetting surfaces and vertices: (a) offsetting surfaces along face normal directions; (b) offsetting vertices along the single normal direction of a vertex; (c) offsetting vertices along the multiple normal directions of a vertex.



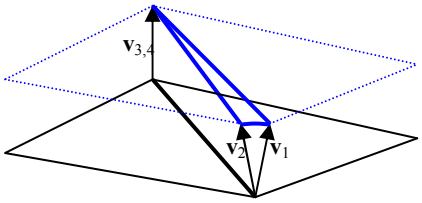


(c)

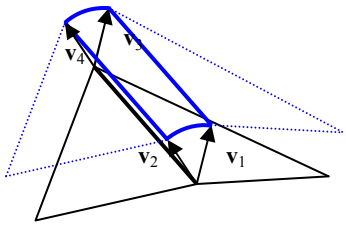
Fig. 2. The multiple normal vectors of a vertex are computed by averaging the face normal vectors: (a) a smooth vertex has one normal vector; (b) three vertex normal vectors computed from five face normal vectors; (c) computation method for the average vector.



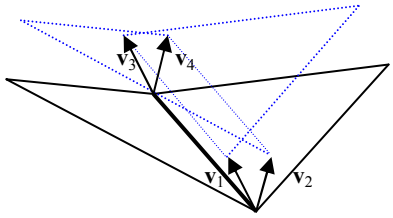
(a)



(b)



(c)



(d)

Fig. 3. Offset edge using the multiple normal vectors of a vertex: (a) smooth edge; (b) the blend surface of three normal vectors; (c) the blend surface fills the gap at the convex edge; (d) the overlap at the sharp concave edge.

3. TRIANGULAR MESH OFFSET

3.1 Offset of faces

For offsetting triangular faces, the vertices of the faces are moved to the multiple normal directions of vertices. No gaps or overlaps appear between the faces with similar normal directions because the vertex surrounded by the faces has a single normal vector. The complexity of offsetting these faces is O(n), where n is the number of vertices on the triangular mesh. The multiple normal vectors with different directions occur at the vertices surrounded by the faces with very different normal vectors. The multiple normal vectors produce a precise offset distance and they do not significantly deform the face of the triangle. The gaps and overlaps are joined by the blend surface, thereby making the shell thickness uniform in layered manufacturing and producing a smooth tool path.

3.2 Offset of edges

The edges that join two adjacent triangular faces are offset by moving two vertices along the normal directions of each vertex. As shown in Fig. 3(a), the smooth edge with the two single vertex normal vectors $\mathbf{v}_{1,2}$ and $\mathbf{v}_{3,4}$ is offset by moving the two end vertices along the normal direction of the vertices. No gap or overlap occurs between the faces because the edge joining the faces is moved along the single direction. The edge shown in Fig. 3(b) with the multiple normal vectors $\mathbf{v}_1$, $\mathbf{v}_2$ and the single normal vector $\mathbf{v}_{3,4}$ is offset by moving the vertices along each direction; the gap near the vertex with the multiple normal vectors is joined by a conic bland surface. The sharp edge shown Figs 3(c) and 3(d) with the two double vertex normal vectors $\mathbf{v}_1$, $\mathbf{v}_2$, $\mathbf{v}_3$, and $\mathbf{v}_4$ is offset by moving each end vertex along two different directions, which makes offset distance precise but creates gap or overlap. The blend surface joins the gap between the two faces if the edge is convex as shown in Fig. 3(c). The blend surface is computed by recursively dividing the two vertex normal vectors at each end point of the edges while the cross product of the vectors is less than allowance.

3.3 Offset of vertices

Since a smooth vertex with the single normal vector $\mathbf{v}_{1,2,3}$ is moved along one direction, no gap or local interference occurs between the faces that surround the vertex shown in Fig. 4(a). As shown in Fig. 4(b), a vertex with the two normal vectors $\mathbf{v}_{1,2}$ and $\mathbf{v}_3$ is moved along each normal direction but no gap is generated because it is filled with the blend surface of the edges. As shown in Figs 4(c) and 4(d), the sharp vertex with the different normal vectors $\mathbf{v}_1$, $\mathbf{v}_2$, and $\mathbf{v}_3$ is divided into three vertices and moved along each normal direction

that makes a gap or overlap between the offset faces. The gap at sharp concave vertex is filled by the blend surfaces. The vertex normal vectors are used to compute the blend surface of the sharp vertices by recursively dividing the three vertices while the cross product of the vertices is less than allowance. The STL model with concave and convex edges and vertices is shaded with the multiple normal vectors of a vertex in Fig. 5(a). All vertices are moved along the multiple normal vectors of the vertices, and the gaps at the sharp vertices and edges are joined by the triangular faces, as shown in Fig. 5(b).

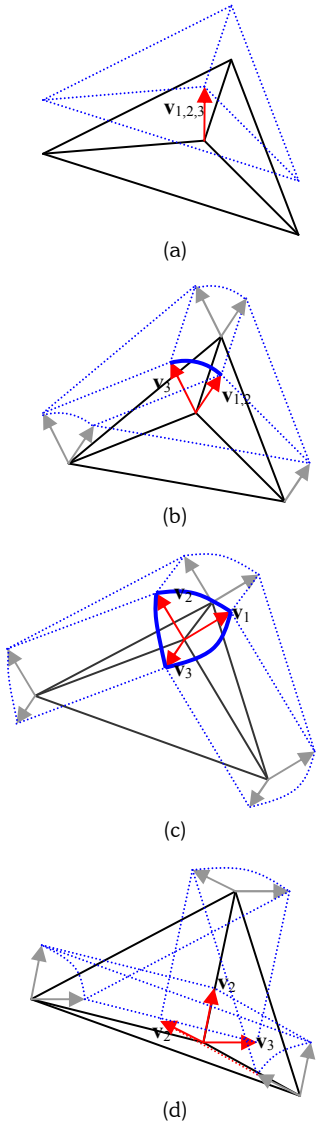


Fig. 4. Offset vertex using the multiple normal vectors of a vertex: (a) a smooth vertex with a single normal vector; (b) a vertex with two normal vectors; (c) the blend surface fills a gap at the convex vertex; (d) the overlap at the sharp concave vertex

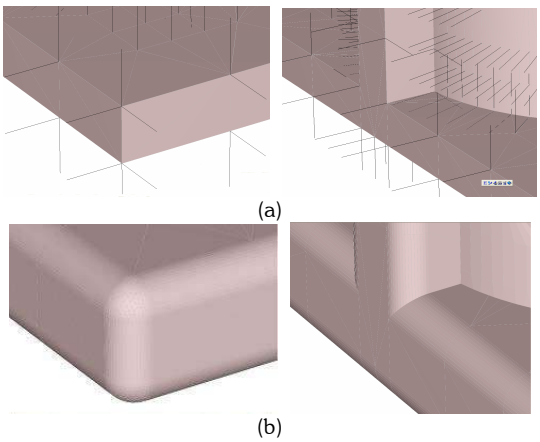


Fig. 5. Offsetting of sharp edges and vertices; (a) enlarged triangle mesh and the multiple normal vectors of a vertex; (b) offset using the multiple normal vectors of a vertex.

3.4 Slicing and loop removal

There are two possible ways to remove the self-intersections on the offset triangular mesh. One complicated way is to remove the self-intersection in 3D spaces. The other way is to remove the self-intersection in 2D spaces after slicing by a series of planes; this way is a suitable method for 3-axis numerical control machining and a layer-based manufacturing process. The offset triangular mesh is sliced by a series of planes to get the tool path lines. Since all the gaps between the faces are joined by the blend surface, the sliced lines of the offset triangular mesh is connected a continuous loop. An invalid loop can be removed by a loop removal method of 2D milling [1].

4. COMPARATIVE EXAMPLES

The C++ language and an OpenGL library were used for the implementation of the proposed offset method and two previous methods. The shaded model in Fig. 6 is triangular mesh generated from a complete solid model by a commercial CAD system. In Fig. 7(a), the face normal vectors of the mesh are shown as black lines at the vertices of the triangular faces. The normal vectors of vertices were computed by averaging the normal vectors of the faces shown in Fig. 7(b). The multiple normal vectors of a vertex are computed by joining face normal vectors surrounding the vertex, as shown at Fig. 7(c). The smooth vertices and edges in the surfaces of the model have a single normal vector. The sharp vertices and edges at the edge of the rectangle and cylinder have the multiple normal vectors of a vertex directing the normal direction of each face.

Using each normal vector (namely, the normal vector of a face, the normal vector of a vertex and the multiple normal vectors of a vertex), the triangular faces and vertices are moved to offset the triangular mesh. In Fig. 7(a), all the faces are moved along the normal direction of the faces by the length of the offset distance [3]. This process produces an exact offset of the triangular faces but with too many gaps and overlaps between the triangular faces, thereby making a long computation time for joining and trimming between the faces. The small faces that join the gaps make the offset mesh and tool path nonuniform. In Fig. 7(b), all vertices are moved along the average vertex normal directions of the normal vectors of the surrounding faces [4, 5]. No gaps occur between the faces but the large geometrical deformation of the triangular faces and the offset errors at the sharp edges are unsatisfactory for numerical control machining. The blended surface at the sharp edge is better than the sharp offset edges for uniform shell thickness and for the tool path of high-speed milling. In Fig. 7(c), all vertices are moved along the multiple normal directions of a vertex. At the smooth vertices and edges, no gap or local interference occurs, and the triangular faces and the block length of the tool path block are uniform. The normal vector of a vertex is computed not by using the simple average of face normal vectors but by using the weighted sum of face normal vectors; in this way, the offset triangular mesh is precise. At the sharp edges and vertices, no large deformation of the faces occurs because the multiple normal vectors of a vertex move the vertices along the face normal directions. The tool path that passes the sharp edges is precise and smooth because it is joined by the blend surfaces. The gaps at the sharp vertices and edges are joined by the blend surface. The sliced lines form loops, enabling easy handling by methods for removing 2D invalid loops. The compared results of three offset methods in Table 1 show that multiple normal methods is better than previous two methods for tool path generation. The offset results of various models in STL formats also are illustrated in Fig. 8. The tool path generated by slicing the offset mesh is verified by NC machining simulation and the results shown in Fig. 8 (d) insure that the proposed offset method is also applicable for NC machining.

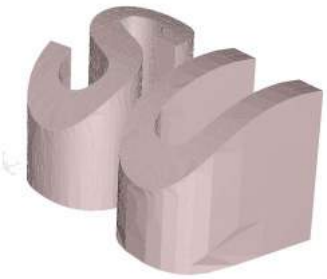
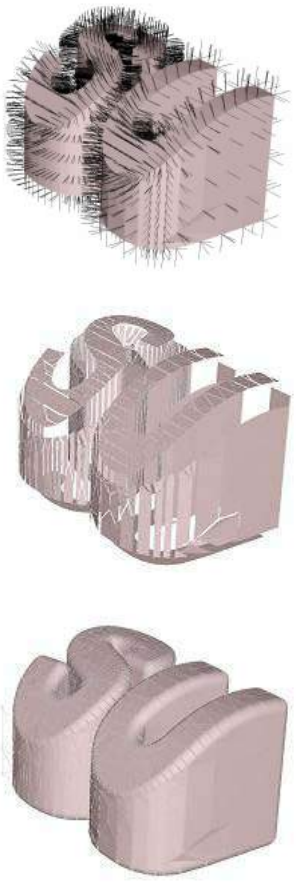
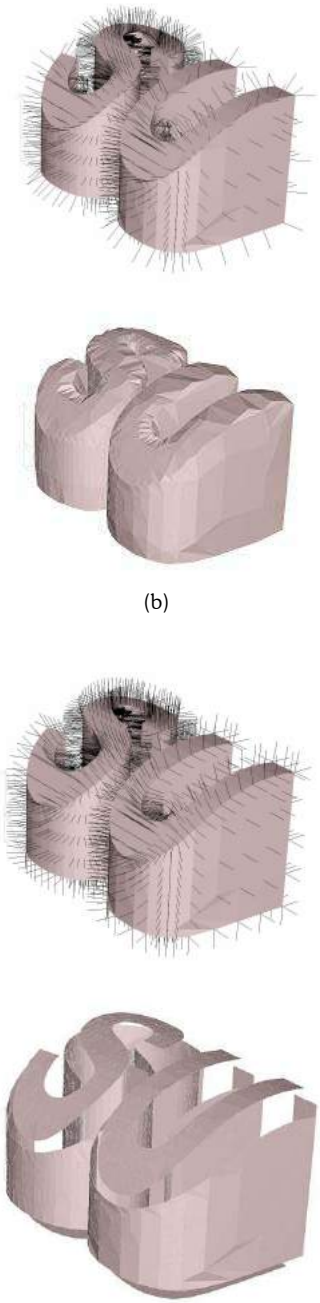


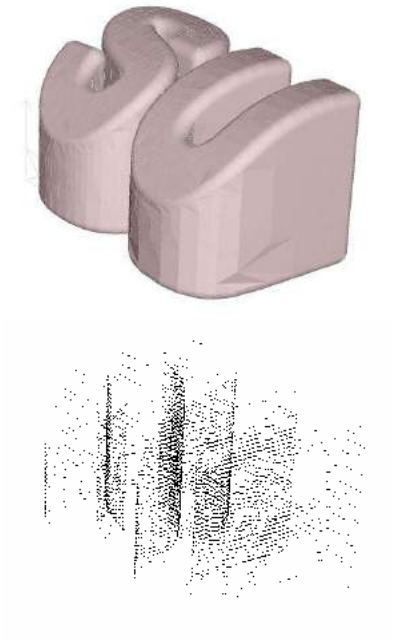
Fig. 6. Shaded view of triangular mesh with 4,200 faces and 0.1mm tolerance.



(a)

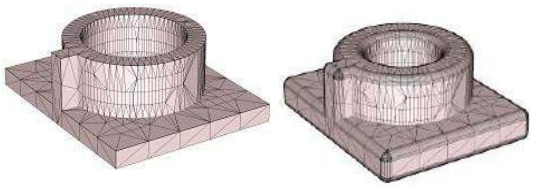


(b)

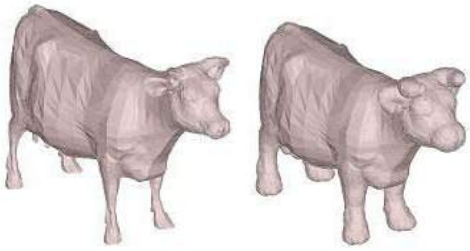


(c)

Fig. 7. Normal vectors of triangular mesh and offset mesh: (a) offsetting surfaces along face normal vectors, 12,600 gaps, 48,842 faces; (b) offsetting vertices along the single normal vectors of a vertex, 3.6mm error; (c) offsetting vertices along the multiple normal vectors of a vertex, 1,676 gaps, 13,923 faces.



(a)



(b)

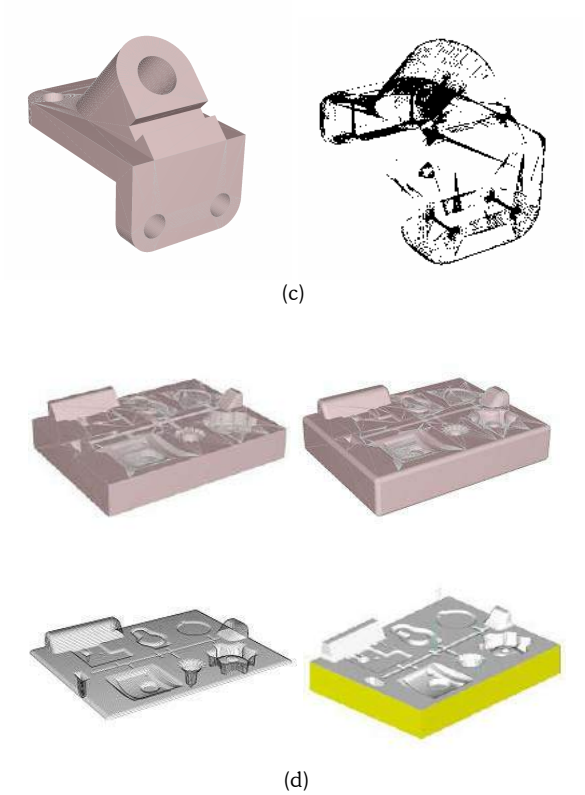


Fig. 8. The offset results of various triangular meshes using the multiple normal vectors of a vertex: (a) STL model and offset mesh of cylinder; (b) Cow; (c) Anchor; (d) STL model, offset mesh, tool path and simulation of injection mold.

	Face normal	*Vertex normal*	*Multiple normal*
Complexity	O(n)+bO(n) [0.375s]	O(n) [0.031s]	O(n)+bO(s) [0.094s]
Number of faces	Increase [48,842]	Same [4,200]	Increase [13,923]
Number of gaps	All edges [12,600]	No [0]	Sharp E. [1,676]
Uniformity	Poor	Good	Good
Precision	Exact [0mm]	Poor [3.5mm]	Allowable [0.1mm]
Tool path	Good	Poor	Better

STL model with 4,200 faces, 0.1mm tolerance
n is the count of faces
b is the time complexity of blending
s is the count of sharp edges & vertices (s<<n)
[ ] is the result of the example, shown in Fig.7

Tab. 1. The comparison of offset methods.

5. CONCLUSION

A triangular mesh offset algorithm that moves vertices along the multiple normal vectors of a vertex is introduced and implemented. The multiple normal vectors of a vertex are set the same as the normal vectors of the faces that surround the vertex, and two vectors with the smallest difference are replaced to an average vector repeatedly until the difference is smaller than allowance. All the vertices are moved along the multiple normal directions, and the gaps at the sharp edges and vertices are filled with blend mesh computed by recursive subdivision.

The offsetting using the multiple normal vectors of a vertex does not create a gap or overlap at the smooth edges, thereby making the mesh size uniform and the computation time short. The method is precise at the sharp edges because the vertices are moved to each face normal direction and joined by the blend surface. The method is useful for rapid prototyping and numerical control machining for triangular mesh that is the tessellated part of solid models with smooth surfaces and sharp edges. The possible disadvantage of this method is that the triangular mesh needs to form a complete solid because the multiple normal vectors of a vertex and the blend surfaces at the sharp edges are calculated using topology between the faces, edges, and vertices.

6. REFERENCES

[1] Hansen, A. and Arbab, F., An Algorithm for Generating NC Tool Paths for Arbitrary Shaped Pockets with Islands, *ACM Transactions on Graphics*, Vol. 11, No. 2, 1992, pp 152-182.

[2] Choi, B.-K., Kim, D.-H. and Jerad, R.-B., C-Space approach to tool path generation for die and mold machining, *Computer-Aided Design*, Vol. 29, No. 3, 1997, pp. 657-669.

[3] Jun, C.-S, Kim, D.-S. and Park, S.-H., A new curve-based approach to triangle machining, *Computer-Aided Design*, Vol. 34, No. 5, 2002, pp 379-389.

[4] Koc, B. and Lee, Y.-S., Non-uniform offsetting and hollowing objects by using biarcs fitting for rapid prototyping process, *Computers in Industry*, Vol. 47, 2002, pp 1-23

[5] Qu, X. and Stucker, B., A 3D surface offset method for STL-format models, Vol. 9, No. 3, 2003, pp 133-141.