Title:
On Reduction of Data Redundancy in STL File by Generating Higher Order Polygons

Authors:
Mayur Vispute, mayur1218@gmail.com, Research Scholar, IIITDM Jabalpur
Narendra Kumar, kumarn@nitj.ac.in, Assistant Professor, NIT Jalandhar
Prashant K Jain, pkjain@iiitdmj.ac.in, Associate Professor, IIITDM Jabalpur

Keywords:
Additive Manufacturing, 3D printing, STL, Redundancy, Slicing, Polygons

DOI: 10.14733/cadconfP.2020.179-183

Introduction:
Additive manufacturing (AM) fabricates complex geometries directly from CAD model by adding material layer-by-layer in a required manner [5]. The AM process flow starts with the CAD model generation followed by the standard tessellation language (STL) conversion, slicing [6], tool path generation and part fabrication. STL conversion is an essential step in additive manufacturing especially for slicing operation. Slicing is an operation of capturing 2D contour information by performing an intersection of the z-plane with STL model. The STL file contains the triangulated surface data of the CAD model. On the one hand, STL file format has been become the de-facto standard in the additive manufacturing due to its capability of storing information in human readable format. On the other hand, the large file size and inherent redundancy in STL data have always been a research concern, which needs the attention of researchers. It contains the information of triangles and their orientation [1]. STL file can be exported in the form of binary or ASCII, depending on the requirement of the user. In ASCII format, the triangle information is stored in the form of three vertices and facet normal [2]. As at least one edge of the triangle is shared with adjacent triangle, redundancy in the STL data comes into picture which ultimately increases the file size [4]. This problem in STL is clearly explained through an example in which one face of the tessellated cube model is considered. It can be seen that considered face of the cube has two triangles, sharing a common edge marked by red colour, as shown in Fig. 1(a). Corresponding STL for the considered face is represented by Fig. 1(b). It can be seen that data points corresponding to common edge are repeated twice in STL file, as presented in red color. This repetition of points increases the file size of STL.

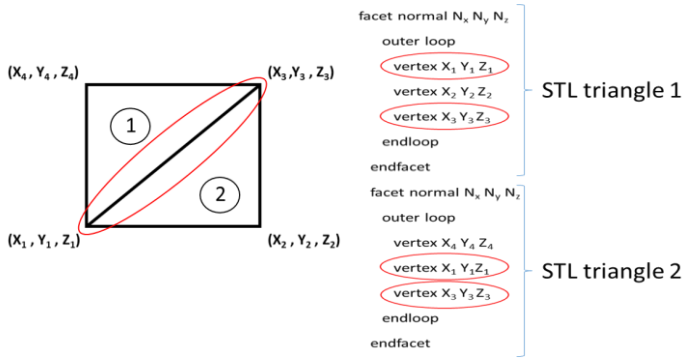

Fig. 1: (a) One face of STL cube model (b) Text format of STL model.

Due to the large file size, slicing of STL file takes more time during the contour extraction at different z height [3], Fig. 2. shows the STL model of a cube containing 12 triangles in which each triangle is sharing at least one edge with another triangle. When this STL file is sliced at particular z height, two points are obtained at the intersection of a triangle with slice plane. In this case, total eight triangles are present, therefore, sixteen points will be obtained at particular z height to generate the closed contour. Square shape contour will be generated when all sixteen points are joined together. These repeated sixteen points can be reduced to eight points if all coplanar triangles are converted into higher polygon by deleting common edge. The file size and the redundancy of STL can be reduced by removing the excess data points presented in its structure. In the present study, a novel approach has been proposed to reduce the file size and redundancy by replacing coplanar triangles of STL into higher order polygons using elimination of sharing edges approach. The MATLAB software has been used to implement the approach. The proposed approach has been validated by considering different case studies.

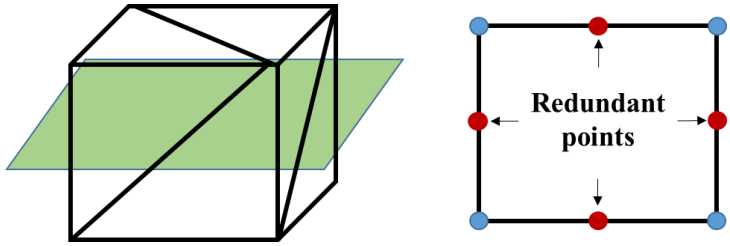

Fig. 2: Slicing operation on the cube for one layer.

Main Idea:
An algorithm has been developed to convert coplanar STL triangles into the higher order polygons by deleting common edges. A program has been written in MATLAB software for extracting coordinates of triangles. After extracting coordinates of triangles, facet normal has been calculated. To understand the approach in detail, consider a triangle having vertices (x1, y1, z1), (x2, y2, z2) and (x3, y3, z3) as shown in the Fig. 3. The facet normal of the triangle was calculated by vector product. From given coordinates, two vectors $\vec{u}$ and $\vec{v}$ were formed as shown in the Fig. 3. First vector $\vec{u}$ was calculated from (x1, y1, z1) and (x2, y2, z2) and second vector $\vec{v}$ was calculated from (x1, y1, z1) and (x3, y3, z3) given by Eqn. (1). and Eqn. (2). respectively.

$$\vec{u} = (x_2 - x_1)\hat{i} + (y_2 - y_1)\hat{j} + (z_2 - z_1)\hat{k} \tag{1}$$

$$\vec{v} = (x_3 - x_1)\hat{i} + (y_3 - y_1)\hat{j} + (z_3 - z_1)\hat{k} \tag{2}$$

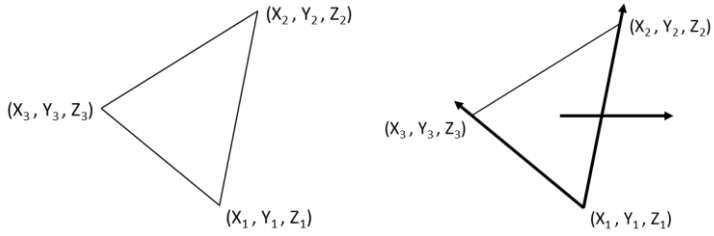

Fig. 3: (a) STL triangle coordinates (b) Cross product on STL triangle.

After formation of two vectors in the same plane, the cross product was performed given by the Eqn. (3). u_1, u_2, u_3 and v_1, v_2, v_3 were three components of $\vec{u}$ and $\vec{v}$ vectors in three directions.

$$\vec{u} * \vec{v} = \begin{pmatrix} \hat{i} & \hat{j} & \hat{k} \\ u_1 & u_2 & u_3 \\ v_1 & v_2 & v_3 \end{pmatrix} \tag{3}$$

The final obtained equation for calculating facet normal is given by:

$$\vec{u} * \vec{v} = ((u_2 * v_3) - (u_3 * v_2))\hat{i} - ((u_1 * v_3) - (u_3 * v_1))\hat{j} + ((u_1 - v_2) * (u_2 - v_1))\hat{k} \tag{4}$$

After vector product, the developed algorithm picked the triangles based on their normal direction calculated by the Eqn. (4). and merged them into higher order polygon. The final obtained file contains all the necessary data with retaining its original internal and external boundaries.

The proposed algorithm has been applied to STL triangles of cube model having unit normal N = [1, 0, 0] as shown in Fig. 4(a). Among sixteen STL triangles, two STL triangles ABC and DAC have same unit normal N = [1, 0, 0] as shown in Fig. 4(b). Firstly, triangle ABC is considered. In this triangle, firstly, edge AB is selected, and repetition of the same edge in reverse order is checked in triangle DAC. As there is no repetition of edge AB in reverse order is found, therefore, this edge AB is added to the matrix to form a new temporary polygon as shown in Fig. 4(c). After that, the same operation is carried out on edge BC and edge CA. Also for edge BC, there is no repetition found. Therefore, it is added in the matrix to form a temporary polygon as shown in Fig. 4(d). For edge CA, repetition is found with edge AC in triangle DAC. After that, the edge is removed from current triangle ABC and found triangle DAC. Remaining edges of triangle DAC are added in matrix to form a new temporary polygon as shown in Fig. 4(e). As a result, a closed temporary polygon is formed. Now there is no other triangle left, thus, currently formed temporary polygon is final higher order polygon.

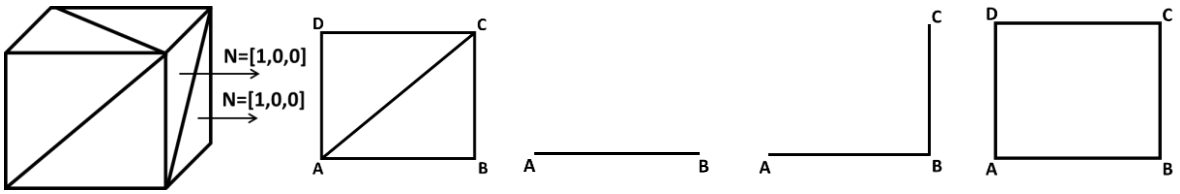

Fig. 4: (a) STL model cube (b) One face of cube model (c-f) Proposed algorithm.

The higher order polygon replaces the coplanar STL triangles. In text format of STL file, the same face is represented by six points which include two repeated points and two facet normals whereas in generated higher order polygon, the same face is represented by four points and one facet normal only as shown in Fig. 5(a). and Fig. 5(b). When slicing operation is carried out on coplanar STL triangles at the layer, four points generated as shown in Fig. 6(a) and 6(b). However, when the same operation is carried out on higher order polygon, instead of four points, only two points are generated. Moreover, there is no geometrical data loss with higher order polygon approach as shown in Fig. 6(c). and Fig. 6(d). In this way, the developed approach offers the advantages during the slicing operation. When slicing operation is carried out, fewer data points will be used to extract the contour information as compared to original STL model. It indicates that developed file format will be different from the STL format which will contain the triangles as well as higher order polygons depending on the geometry and the accuracy of original STL file. Therefore, in this developed file format, higher order polygons are written by their vertices and facet normals same as the structure of STL file format. A utility has

been developed in MATLAB software which can read both file format i.e. developed file format and STL format.

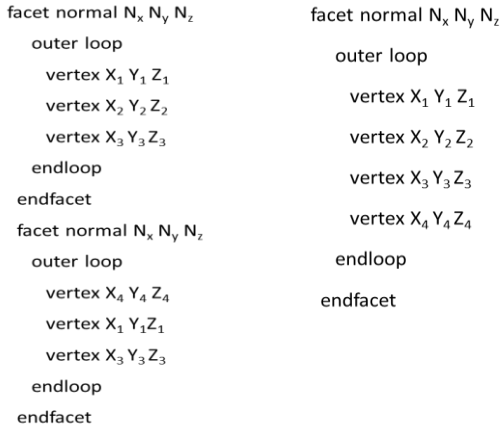

Fig. 5: (a) STL file format (b) Proposed format.

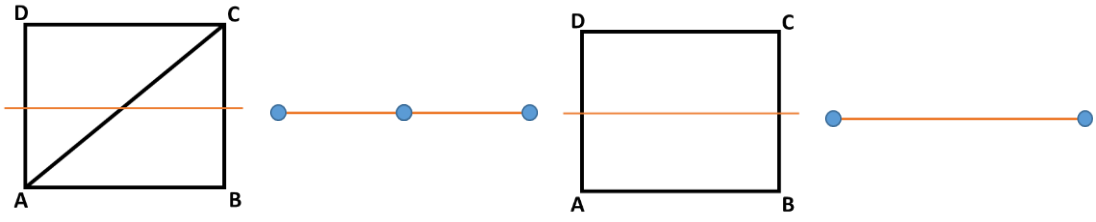

Fig. 6: (a) Slicing operation on STL triangles (b) Sliced layer of STL triangles (c) Slicing operation on higher order polygon (d) Sliced layer of higher order polygon.

Case Study:

The proposed approach is verified by considering a case study. A model with complex geometry is considered to show the effectiveness of proposed approach. The original STL of the considered model consists of 1680 triangles, as shown in Fig. 7(a). After applying the developed algorithm, only 703 polygons can represent the same geometry without any data loss, refer Fig. 7(d). This can be validated by slicing the geometry obtained from STL and proposed approach. The slicing operation was performed at particular z-height on the STL and proposed model, refer Fig. 7(b-c) and Fig. 7(e-f). A massive reduction in the sliced data points was observed for proposed model. The sliced points were reduced up to 50%. It is observed that, the number of slicing points has been reduced for each layer. The reduction in number of slicing points causes a reduction in file size of the model by 70.78%. In addition, the slicing time was also calculated for both models which also shows the improvement through proposed approach. It is observed that, slicing time is reduced by 39.95%.

Conclusion:

The redundancy in STL file affects the slicing operation. The present article was focused on reduction in redundant data and large file size of STL model. The redundancy problem was eliminated by converting all coplanar triangles in STL into higher order polygons. Moreover, a new file format was also proposed which stores the information of polygons in the form of vertices and facet normals without affecting the original geometry of model. Two case studies were considered to validate the proposed approach. Results were reported by comparing the file size and slicing time for STL and

proposed a model. The obtained results showed the significant reduction in file size and slicing time for proposed model. The outcome of the current study may be useful for slicing operation on the models consist of large number of STL triangles in which huge slicing time is required. In future, the proposed file format can be embedded in the modeling software by replacing STL file format.

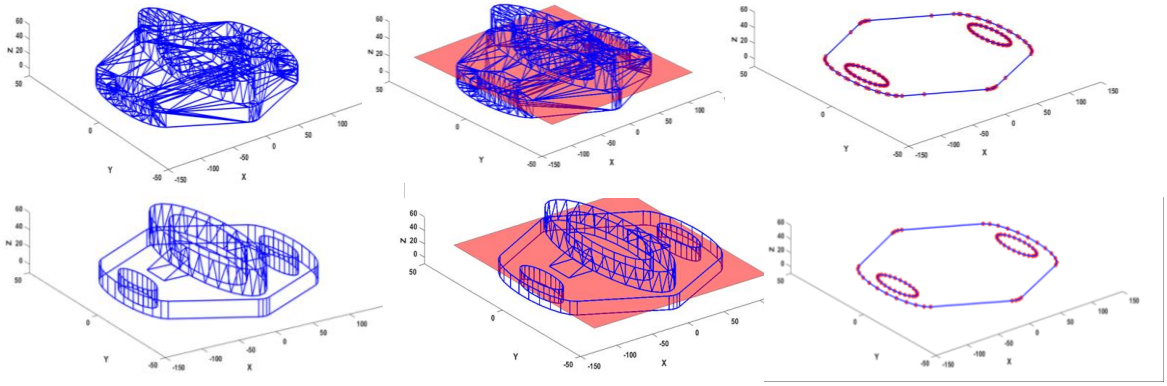

Fig. 7: (a) STL model of the geometry (b) Slicing operation on STL model (c) One sliced layer after slicing STL model (d) Proposed model of the geometry (e) Slicing operation on the proposed model (f) One sliced layer after slicing proposed model.

References:

[1] Ciobota N.: Standard Tessellation Language in Rapid Prototyping, Sci. Bull. VALAHIA Univ., 7(7), 2012, 81–85, http://fsim.valahia.ro/sbmm.html/docs/2012/mechanics/4_Ciobota_2012.pdf.

[2] Fadel G.-M.; Kirschman C.: Accuracy Issues in CAD to RP Translations, Rapid Prototyping Journal, 2(2), 2006, 4–17. https://doi.org/10.1108/13552549610128189

[3] Koc B.; Ma Y.; Lee Y.: Smoothing STL Files by Max-Fit Biarc Curves for Rapid Prototyping, Rapid Prototyping Journal., 6(3), 2000, 186–205. https://doi.org/10.1108/13552540010337065

[4] Liu, F.; Zhou H.; and Li D.: Repair of STL Errors, International Journal of Production Research 47(1), 2009, 105-118. https://doi.org/10.1080/00207540701424539

[5] Shaikh S.; Kumar N.; Jain P.-K.; Tandon P.: Hilbert Curve Based Toolpath for FDM Process, CAD/CAM, Robotics and Factories for the Future. Springer India, 2016, 751–759. https://doi.org/10.1007/978-81-322-2740-3_72

[6] Taufik M.; Jain P.-K.: A Study of Build Edge Profile for Prediction of Surface Roughness in Fused Deposition Modeling, Journal of Manufacturing Science Engineering, 138(6), 2016, 1–11. https://doi.org/10.1115/1.4032193