

Migration of Conventional Model Transfer Format on Additive Manufacturing

Deng-Hui ZHANG ^{1,a*}, Zheng-Xu ZHAO ², Yi-Qi ZHOU ³, Yang GUO ²

¹Key Laboratory of High Efficiency and Clean Mechanical Manufacture of MOE, Shandong University, 250100 Jinan, China

²School of Information Science and Technology, Shijiazhuang Tiedao University, 050043 Shijiazhuang, China

³School of Mechanical Engineering, Shandong University, 250100 Jinan, China

^azhang.denghui@foxmail.com

*Corresponding author

Keywords: STL; data reduction; data migration; additive manufacturing.

Abstract. The STL (Standard Transform Language) file format has been the industry standard for Additive Manufacturing (AM) communities for last two decades. It falls short with the rapid development of AM related hardware and materials technologies. Kinds of alternatives to STL have been proposed. However, Massive 3D models are still described using STL file format. It is necessary to migrate STL files to an alternative format. In this paper, a redundancy vertex removal algorithm is first proposed to achieve a constant time in data reduction. A simple and friendly-to-operations AM file format is then proposed to exchange data between CAD models and the AM process, its performance of data migration is evaluated and 3D printing experiment is carried out to validate the feasibility of the data migration.

Introduction

With the development of manufacturing technologies, AM (also called 3D printing) technology is gaining lots of traction lately[1]. AM is quickly evolving from primarily single-material, homogenous parts to multi-material geometries in full colour with functionally graded materials and microstructures.

The unique feature of AM is its fabrication capability without involving tooling, fixturing, and other peripheral activities associated with conventional manufacturing. Its novelty stems from the layer-wise deposition principle offers a range of possibilities including multi-material structures, and mesoscopic devices[2].

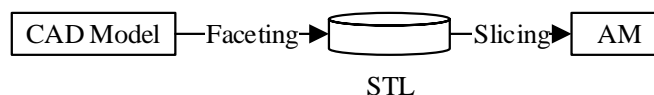


Fig. 1. The CAD model to AM data flow

A schematic illustration of the data flow from a CAD model to the AM process is shown in Fig. 1. STL file format is the de-facto standard in AM communities. It is used to exchange production information between CAD models (surface or solid) and the AM system. STL plays an important role in manufacturing of parts. However, it falls short with the rapid development of AM related hardware and materials technologies, which motivates the development of alternative AM data exchange formats. Many 3D file formats such as Surface Triangles Hinted (STH) format, Cubital Facet List (CFL), RPI[3], Additive Manufacturing File (AMF)[4] and Standard for the Exchange of Product model data (STEP)[5] have been proposed by AM communities to replace or extend the STL file format[6]. However, none of them have met with enthusiasm from the AM community. A

* Corresponding author: zhang.denghui@foxmail.com

general accepted solution for model migration in AM communities does not exists[7]. Regards to the current situation of AM communities, although an international standard for data transfer which is suitable and gaining the traction of AM communities will become available in the future, it is still necessary to migrate old STL file to the new AM format.

In this paper, an AM application system which overcomes the shortcomings of STL was developed. Its objective is to establish a framework to reuse existing model data with the advent of AM technology. The performance of data migration is emphasized, which is critical when transferring huge data through a low bandwidth. The effectiveness of data reduction using an elaborate hash function is evaluated. A AM data exchange format which is simple and friendly to program languages is proposed to substitute for the STL file format. The file format is easily extensible to support topology and materials information. At last, a 3D printing experiment is carried out to validate the feasibility of the data migration.

STL file migration

The STL file format was established by 3D Systems in 1987. It consists of a number of triangular facets. Each triangular facet is represented by three unique vertices and a normal direction pointing towards the exterior of the model. There are two data formats, ASCII and binary in the STL standard. The file stored by an ASCII format is more readable at the cost of large size, while there is a smaller file size in the file stored by a binary format.

Table. 1 Advantages and disadvantages of STL file format.

Advantages	Disadvantages
Simple	Redundancy
Wide range of input	Approximation
Simple slicing algorithm	Poor scalability
Splitting STL models	Lacks of auxiliary information

The STL format has been used by AM communities for almost three decades. It was developed to serve as an exchange format for the stereolithography process. Although the stereolithography process is the first commercial AM technology, it was not designed to be an industry standard. The advantages and disadvantages of STL are presented in Table. 1. The STL file structure is clear, but it lacks of topology information. Vertices among adjacent facets have to be stored repeatedly for each facet. Based on the right-hand rule each facet normal can be calculated from order and location of three vertices. That is, facet normals should have been ignored in the STL file. So a data reduction process has to be taken before STL file migration.

Data reduction

According to the Euler equation of the spatial geometry $V-E+F=X(P)$, where V is the number of vertices on the polyhedron P , F is the number of facets and E is the number of edges. A STL file is composed by closed geometry models, so it can be got that $X(P)=2$. The edge in each triangular facet is shared twice, so $E=3F/2$. The number of vertices stored in the STL file is $MV=3F$. It can be found that the relationship between MV and V has the form

$$MV = 6(V - 2) \quad (1)$$

That is, each vertex is stored approximately 6 times. The most direct way to remove redundancy is to delete vertex by vertex, which runs in $O(N^2)$ time. It is inefficient since a STL file often contains millions of vertices. Other data reduction techniques include binary search tree algorithm(BST) and hash table algorithm[8]. When inserting a new vertex, self-balancing BST performs transformations on the tree to keep its height small. This yields average time complexity $O(\log_2 N)$. However, it has been proven that it is not always feasible to maintain the height of a tree at its minimum value or

give a lower bound constant to keep the height small. The precision problem of floating point also makes BST ineffective in the comparison operation.

The hash table algorithm lookups a node by hashing method. By constructing a suitable hash function, comparison and search operations can proceed in constant time. In an ideal world, different keys are mapped to different addresses according to a mapping method, but a plurality of keys are usually mapped to the same addresses. Hash table construction process has to be divided into two steps: The first step is to map a key to its table address. The hash function is called frequently in reduction process, so it has to be designed carefully to retrieve a good balance between process time and memory usage. It should not be too complicated to be a time-consuming operation. One of mapping methods is the modulus operation, which selects a prime number P as the array size. The address of key K can be got through dividing K by table size P , that is $H(K) = K \bmod P$. Since a key must be an integer, vertex coordinates in floating point form has to be converted to integer correspondingly. Based on the feature of accuracy of floating point and three coordinates components V_x , V_y , V_z of a vertex in the STL file, a hash function which maps coordinates components to an integer has the form

$$K = V_x * 1000000 + V_y * 1000 + V_z * 10 \quad (2)$$

The second step is to resolve address conflicts. The usual way is to build a list for nodes of the same hash address. It uses minimal memory, but it increases the process complexity. Another method is the linear detection method. If the size of a hash table can be predicted, its wasted memory problem can be avoided.

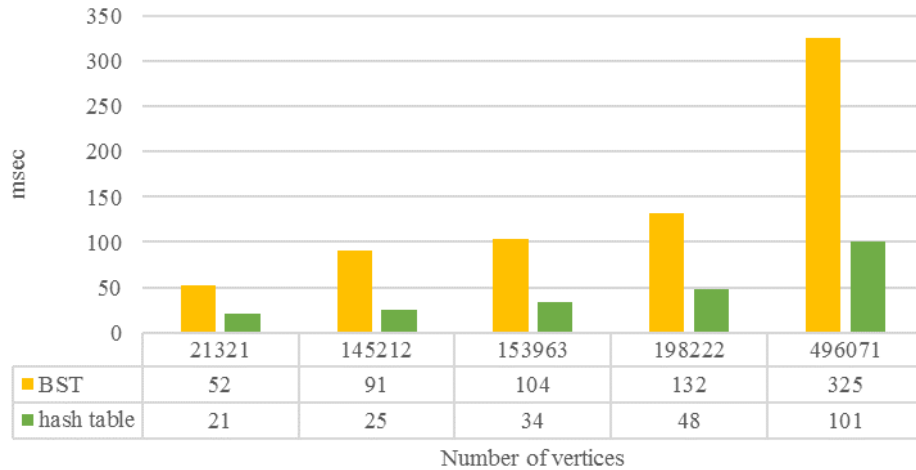


Fig. 2. Data reduction time of hash table and BST for different STL files

For the STL file, before reading vertex coordinates, the number of facets F and the number of vertices $F/2$ have been acquired. A sparse table can be constructed according to the number of vertices. It can detect conflicts by just one search. To achieve better balance between running time and memory space, the smallest prime number P which is bigger than $F/2$ is selected in Messene Prime. Considering the complex calculation process of Merlin Prime, a pre-computed Messene Prime table is adopted. The time efficiency comparison of hash table algorithm and self-balancing BST algorithm is shown in Fig. 2. 6 STL files are used for performance evaluation. The computer used in the experiment is a PC with 4G memory and Intel Core i3 CPU. Yellow bars represent BST reduction time and Green bars represent hash table reduction time. It could be found that the time cost is significantly reduced by using the proposed hash table algorithm, which is 3 times shorter than the standard BST algorithm.

YMF file format. The STL file format lacks of topology and materials information. kinds of alternatives have been proposed. Many of them are represented by XML (Extensible Markup Language) format[4], [9]. XML is described in self-descriptive plain text, which is easy to exchange across different platforms and networks. In order to describe additional metadata, data stored in XML format occupies larger storage space than other formats. The XSLT and parent-child

relationships of nodes bring great complexity in codecs of XML file[10]. A couple of light-weight data exchange languages have been proposed to overcome the shortcomings of XML. JSON can be recognized by browsers' JavaScript interpreter and is widely used in web development. YAML is even simpler to produce and modify than JSON. It uses indent to indicate structure. Arrays are indicated by dashes. Keys are separated from values by a colon. That is almost the entire contents of its syntax. Its delimiters make is friendly to operations, while XML requires a sophisticated parser.

<i>YMF File Format</i>	
mesh:	
- vertices:	
- x:	0
- y:	1.32
- z:	3.72
- x:	0
- y:	1.27
- z:	2.45
-	...
- triangles:	
- v1:	0
- v2:	1
- v3:	3
- v1:	1
- v2:	0
- v3:	4
-	...

Fig. 3. The proposed YMF file format

A new file format called YMF (Yaml-based additive Manufacture File) is proposed to replace STL. As shown in Fig. 3, YMF is a compact YAML-based additive manufacture data exchange format. It uses a face-vertex polygon mesh layout. The top level tag mesh contains two sub-tags: vertices and triangles. The vertices tag comprises all vertex coordinates used in this object. The sub-tag x, y and z give the position of each vertex in 3D space. After vertex information, the triangles tag com is used to define triangles by the right-hand rule from indices of the vertices tag. Indices of three vertices are specified using v1, v2 and v3 tags. The data redundancy of STL is eliminated by the fact that common vertices of triangles references the same vertex index.

The YMF file also represents data types such as lists, maps in high-level languages. The feature makes it easy to transform between data models and program languages and friendly to file operations such as read, modify. The proposed YMF file format is easily extensible without modifying its old parser. It only contains compact geometry information currently. Material and adjacency tags can be added to support materials and topology information in future work.

3D Printing

To evaluate the printability effectiveness of theoretical considerations, An AM system to migrate STL files to YMF models was developed with Python language on Windows platform. Fig. 4 shows the printing result of compact YMF models.



Fig. 4. YMF models manufactured by the printer cluster

In order to improve its performance, a C language extension module is used in data reduction process. YMF models are manufactured by A Markbot3D printer cluster based on Fused Deposition Modeling (FDM) technology with the plastic PC-ABS material. The system works well for all the tested models.

Conclusions

The data transfer problem between the CAD model and rapid prototyping systems will remain for a while yet. When developing more suited file format with regard to the future AM technology, conventional 3D data should not be abandoned. In this paper, an AM application system was developed to reuse existing model data with the advent of AM technology. a redundancy vertex rapid removal algorithm is first proposed. Based on the characteristics of STL files, a hash function is designed to achieve constant time in data reduction process. Different from XML-based alternatives to STL, a YAML-based additive manufacture file format YMF is proposed. It is easy extensible and friendly to operations in program languages such as Python, Perl and Ruby.

Compared to other AM methods such as selective laser sintering (SLS) and stereolithography (SLA), the FDM is the most widely used AM process, but FDM is a fairly slow process, which is infeasible for bulk products manufacturing. To improve printing efficiency, the next work is to print models parallelly with the 3D printing cluster.

References

- [1] The third industrial revolution, The Economist, (2012).
- [2] Y. Guo, Research into the Methods for Modeling and Designing Non-natural Mechanical Properties of Material Structure Using Complex Networks, Shandong University, (2015).
- [3] V. Kumar and D. Dutta, An assessment of data formats for layered manufacturing, Adv. Eng. Softw., **28**, 151–164, (1997).
- [4] J. D. Hiller and H. Lipson, STL 2.0: a proposal for a universal multi-material additive manufacturing file format, Solid Free. Fabr. Proc., 266–78, (2009).
- [5] M. J. Pratt and B. D. Anderson, A shape modelling applications programming interface for the STEP standard, Comput.-Aided Des., **33**, 531–543, (2001).
- [6] I. Stroud and P. C. Xirouchakis, STL and extensions, Adv. Eng. Softw., **31**, 83–95, (2000).
- [7] Z. Zhao and J. Fan, An Investigation into Small World Phenomenon in Engineering Informatics, *Computing in the Global Information Technology, 2008. ICCGI'08. The Third International Multi-Conference on*, IEEE, 125–130, (2008).
- [8] T. H. Cormen, Introduction to algorithms, MIT press, (2009).
- [9] X. Y. Kou and S. T. Tan, An XML Implementation for Data Exchange of Heterogeneous Object Models, *Advanced Design and Manufacturing Based on STEP*, Springer London, 419–438, (2009).
- [10] Maeda, Performance evaluation of object serialization libraries in XML, JSON and binary formats, *Digital Information and Communication Technology and it's Applications (DICTAP), 2012 Second International Conference on*, IEEE, 177–182, (2012)