# Mini Review on Efficient Data Structure For 3D Modelling of Polygonal Mesh

*Nurhamiezrah MIDIN* & Abdullah BADE*

Mathematics, Graphics and Visualization Research Group (M-GRAVS), Faculty of Science and Natural Resources, Universiti Malaysia Sabah, Jalan UMS, 88400 Kota Kinabalu, Sabah,

*Corresponding author: mynz_07@yahoo.com,

Tel: +6088-320000, Fax: +6088-435324

*Received:* 29 June 2015
*Revised:* 29 July 2015
*Accepted:* 30 July 2015
*Online:* 5 August 2015

*Keywords:*
Polygonal mesh; data structure; remeshing; random access; streaming representation

**Abstract**

3D graphics has become an increasingly important part in geometric modelling visualization. 3D model is mainly represented by polygonal mesh. Highly complex meshes result in expensive rendering cost, exceeding the memory storage, difficulty to transmit data and unable to be edited. The development of data structure to store meshes information and handling those problems have begun since 30 years ago. This paper aims to highlight the major approaches of various types of algorithm used to address specific problems in storing mesh data over a decade. Trend has shown that remeshing, random access and streaming representation are the methods that been used widely recently. We believe this paper will help other researchers to be familiar with polygonal mesh and their connotation.

## Introduction

Polygonal meshes are often used to represent 3D models. There are various types of mesh such as triangles, quadrilateral and tetrahedral. Polygonal mesh refers to a closed shape. The shape is constructed by edges formed by connected vertices (Neperud, 2005). The simplest polygon is triangle. Figure 1 is an example of 3D model that created by using triangular meshes. The triangles become the surface of the model. Basically, triangles and quadrilateral are used to represent surface of 3D objects without concerning the volumetric of the objects. The surface is constructed by one or more polygons with shared edges. This paper will only focus on representation of 3D models' surfaces. Meshes are classified to few characteristic such as dynamic or static mesh, multiresolution or single resolution mesh and manifold or non-manifold mesh. Connectivity and geometry of meshes are crucial to be deal with.

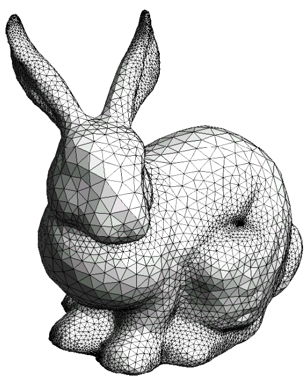


**Figure 1**: Triangle Mesh Model (cs.brown.edu)

TRANSACTIONS ON SCIENCE AND TECHNOLOGY

*Static and Dynamic Mesh*

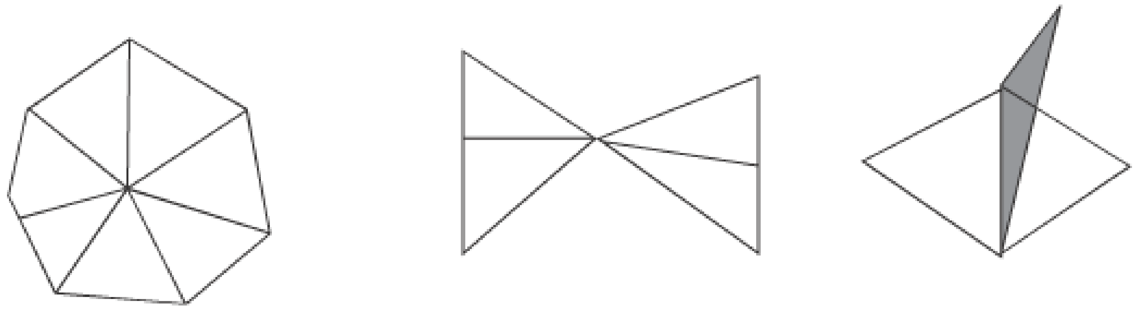
Static mesh refers to a mesh that cannot be modified. Their vertices cannot be animated in visualization. The mesh only can be transformed to another location, size or orientation. It can only be translated, rotated and scaled since they are cached in video memory (SM, 2014). They are efficient to be rendered but more complex than other types of geometry. In simplification of model, static simplification creates different level of detail of the object in several discrete versions. This process occur offline without regards to real-time rendering constraint. Dynamic mesh has opposite characteristic with static mesh. Dynamic mesh allows modification of mesh that includes in changing the geometry and topology of an object. This type of mesh can produce a more realistic 3D object. However, efficient algorithms are required to handle the complexity and limitation of dynamic mesh. The flexibility of this mesh helps in discovering the potential of algorithms to be combined in order to create more reliable data structure in handling complex mode (Serna *et al.*, 2011).

*Level-of-Details and Multiresolution*

Level-of-details model allows multi-representation of an object at different level of details depends on their visual important and the requirement of the application. Models are visualized in specific level of details after selective refinement. This is an operation where a level of representation is extracted (De Floriani *et al.*, 2005). Multiresolution analysis is an outline to represent data set in different levels of resolution. The initial data set decomposed into a sequence of details based on the requirement (Roy *et al.*, 2010). Multiresolution mesh is commonly used to construct compact data structure because of its ability to present models in various resolutions. The main advantage of both LOD and multiresolution is their ability to represent models and data sets in different levels and resolution.

*Manifold and Non-manifold*

If each edge of a polygonal mesh incidents to one or two faces, it is a manifold mesh. The features on the boundary are connected in form of ring and produce a single surface (Luebke, 2001). For triangular mesh, each edge is shared by two triangles. All the triangles have three neighbouring triangles as each triangle has exactly three edges. The coordination of the triangle faces are in cyclic arrangement of the incident vertices. Non-manifold mesh has the opposite characteristic. It has self-intersecting, holes, separate object, inner faces and overlapping geometry. Figure 2 shows the examples of manifold and non-manifold triangle meshes.



a) b) c)

**Figure 2**: a) manifold, b) non-manifold, c) non-manifold (www.autodesk.com)

*Topology and Geometry*

Mesh elements are described by their topology or commonly known as connectivity and their geometry. Mesh topology refers to the incidence relationship between mesh elements. Mesh geometry is for vertex geometric characteristic such as its position (Luebke, 2001). Modification of mesh topology and geometry are allowed only for dynamic mesh with specific algorithm. Some algorithms do not change either topology or geometry of mesh. Some algorithms make changes to mesh topology or geometry in order to simplify, compress or remesh a model.

**Background and History**

Active research on polygonal mesh has begun since three decades. From the last decade, issues regarding graphic visualization have been discovered along its high demand in courses like medical visualization, architecture and advertisement. Virtual visualization has made works easier in terms of modification such as in architecture and reduces implementation cost such as in medical and battle training. In other case, 3D visualization has opened a different way to present ideas, models or entities such as in advertising and product promotions. Painting and printing are not the only method to illustrate them. However, high demand of this representation has led to many problems in visualization such as computational cost and visual fidelity. Next subtopic will discuss about the issues raised in 3D visualization. Polygonal mesh will be the major topic in overcoming the problems regarding 3D visualization.

*Issues*

Surface meshing has been explored massively and many issues are discovered along with the rapid development of technology. This topic will discuss concisely about issues raised in polygonal mesh.

*Visual fidelity and time efficiency*

High visual quality of a model representation can be achieved by composing an object using high number of vertices. However, it is costly in performance because of increasing in computation. Therefore, adjustment always occurs between processing time and visual fidelity of graphic models.

Progressive mesh is one of the methods to produce high fidelity of visualization. Meshes are processed incrementally until they are fully compressed, simplified or modified (Maglo & Hudelot, 2013). Quadric error metrics are also used as an algorithm to process complex model for better visual fidelity. Modified meshes will be compared to the original model to calculate the differences between them. Smaller values of errors indicate higher fidelity of the models. This technique is known as quadric error metrics (Li *et al.*, 2010). Some algorithms are efficient in computation time such as vertex clustering technique where vertices of the meshes are clustered into a number of groups before being processed from one cluster to another (Rus & Vasa, 2010).

*Geometry accuracy*

In particular application such as medical or biological data, geometry accuracy plays an important role to visualize the exact properties and geometry of patient or subject. This is important to reduce the risk of failure in operating the real patient. Another example is in architecture. Geometry accuracy is crucial to plan and construct high quality monuments or buildings. The stability and resistance of a building can be tested virtually before it is built. Therefore, the accuracy of the model is also essential. Inaccurate geometry representation in medical or architecture visualization may result in patients' decease or crashes in building.

*Flexibility*

Most current data structures are only formulated to deal with specific input model. One data structure for one mesh problem. Mesh models are created with different types and structures that made them incompatible with certain data structure. Input models may have non-manifold structure, holes and isles, gaps and overlap, self-intersecting, inconsistent orientation or complex edges [Attene13]. A flexible data structure is where it is able to manage any type of input mesh by repairing the model to a form that is convenient for further process. Some of algorithms that are useful to handle those types of mesh input include consistent normal orientation (repair the orientation of meshes), surface based holes filling (cover the holes in meshes), mesh conversion to manifold (modify non-manifold meshes to manifold), gap closing between meshes and topology simplification.

*Space efficiency*

Space efficiency is the ability to manage huge size of mesh data in internal memory or creating external memory to hold the data. There are some techniques to achieve space efficiency such as compression and simplification of data. This will reduce the size of the data or simplify mesh geometry so it will fit in the memory. Another approach is by using out-of-core representation where external memory is used as additional data storage. Another way to operate huge mesh data set is by using compact data structure such as SQuad (Luffel *et al.*, 2014).

*Easiness of use*

In order to visualize and manage many models, a simple to code data structure may be a demand as to reduce time taken to formulate algorithm for different models. An automatic algorithm may be a high demand or available mesh library will be an advantage such as OpenMesh (Botsch *et al.*, 2002).

**Scheme**

There are few frequently used schemes that favored by researchers to be used or invented by joining two or more schemes to produce a more reliable data structure.

*Simplification*

Zheng (2012) proposed General Mesh Simplification (GMS) which can simplify any type of mesh that set in Euclidean spaces of 3D approximately. The idea is to take all the vertices of a mesh and unite them on the barycentre of the mesh. This is called a decimation operation type that is usually used to simplify progressive mesh. Boubekeur and Alexa (2009) introduced fast mesh simplification algorithm named TopStoc. This TopStoc uses stochastic vertex selection where the vertices are selected randomly and then re-indexed to form a more simple arrangement of triangles. This algorithm is compatible only for triangular mesh. It requires minimum data but still preserves geometrical and topological features. Li *et al.* (2010) presented mesh simplification algorithm by introducing absolute curvature, a method to calculate the value of curve into evaluation of quadric error metric. Edge collapse algorithm is used to simplify the arrangement of triangles. The results are improvement of simplification efficiency and computational complexity. Geometric features are preserved even after drastic simplification.

*Edge collapse*

$$(v_i, v_j) \rightarrow \tilde{v} \tag{1}$$

$v$ denotes the vertices of meshes. $v_i$ and $v_j$ are vertices that will be processed or moved to collapse an edge(s). Vertices $v_i$ and $v_j$ will be moved to the new position $\tilde{v}$ and all incident edges are connected to $v_i$ and $v_i$ will be deleted. The process is illustrated in Figure 3.

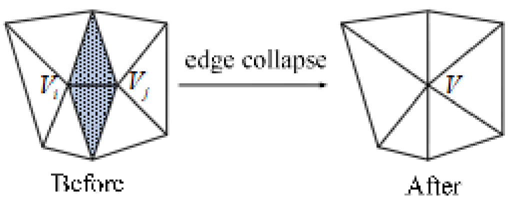



**Figure 3**: Edge collapse (Li *et al.*, 2010)

Quadric error metric (QEM) is defined as below:

$$\Delta(v) = \sum_{pc\,planes\,(v)} d_p^2\,(v) = \sum_{pc\,planes\,(v)} v^T (K_p) v = v^T \{\sum_{pc\,planes\,(v)} (K_p)\} v \quad (2)$$

$(v)$ denotes triangle plane set of vertex $v$. Quadric error of vertex $v$ is sum of squared distance between vertex v and triangular planes $(d_p^2\,(v))$ where $v = (x, y, z, 1)^T$. $x, y$ and $z$ are referred to the coordinates of vertices in xyz-plane. $p = [a, b, c, d]^T$ represents the plane defined by equation $ax + by + cz + d = 0$ in which $a^2 + b^2 + c^2 = 1$. $a$, $b$ and $c$ are constant number which decide the location of the vertices. T is the matric transpose.

$$K_p = pp^T = \begin{pmatrix} a^2 & ab & ac & ad \\ ab & b^2 & bc & bd \\ ac & bc & c^2 & cd \\ ad & bd & cd & d^2 \end{pmatrix} \quad (3)$$

Let $Q'(v)$ denotes as the QEM of vertex v matrics, $Q'(v) = K_p$ where $K_p$ is the matric of the vertex, the initial estimated error of each vertex is zero. When edge collapse is formed, the cost of edge collapse is

$$\Delta(\bar{v}) = (\bar{v})^T (Q_i + Q_j)\bar{v} \quad (4)$$

and QEM of new vertex $\bar{v}$ can be represented by $Q_i + Q_j$.

*Absolute curvature*

$$K = (2\pi - \sum_{i=1}^{k} \emptyset_i \,.\frac{3}{A}) \quad (5)$$

$K$ is Gaussian curvature which represents curvedness of model at the vertices. $A = \sum_{i=1}^{k} f_i$ represents sum of the triangle areas ($f$ as the triangle's area) which related to vertex $v$ and $\emptyset$ represents apex angle associated with vertex $v$. Neighbouring curvedness represented by mean curvature. Let $e_i$ represent the edge with endpoint vertex v, the mean curvature $H$ is defined by

$$H = (\sum m(e_i)).\frac{3}{A} \quad (6)$$

where $m(e_i)$ denotes the angle of normal of two adjacent triangles. Assuming two main curvature of vertex is $k_1$ and $k_2$,

$$k_1 = H + \sqrt{H^2 - K} \tag{7}$$

$$k_2 = H - \sqrt{H^2 - K} \tag{8}$$

if $H^2 - K > 0$, let $H^2 - K = 0$. Absolute curvature of vertex is

$$k_{abs} = |k_1| + |k_2| \tag{9}$$

The algorithm starts with calculating the absolute curvature of each vertex in mesh model. For each vertex, the cost of every vertex is calculated and the least cost put in the priority queue. The least cost then going through consistency test and will be deleted after pass the test. All affected area will be updated. The simplification process repeated until the requirement achieved.

Figure 4 shows the simplification comparison between QSlim algorithm and Li *et al.* (2010) proposed algorithm. The proposed algorithm produced simpler organization of triangles as the meshes has lesser intersection as shown in the figure.

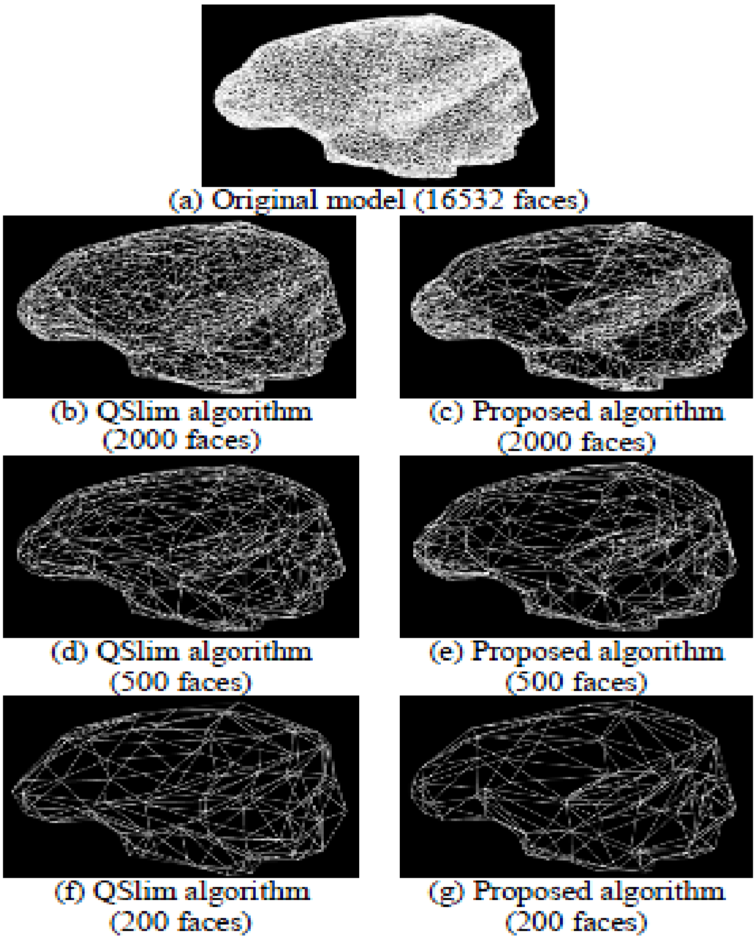



**Figure 4**: Head model simplification comparison (Li *et al*., 2010)

*Compression*

Polygonal mesh is used persistently in graphic visualization. The sizes of meshes for models are increasing and there is no sign that this trend will change. Compression is needed to cope with huge size of polygonal mesh for transmission and storage. Peng *et al.* (2005) and Alliez & Gotsman (2005) produced complete reviews of mesh compression. Since then, many new methods are proposed. There are 4 types of mesh compression that is single-rate compression, progressive mesh compression, random accessible mesh compression and mesh sequence compression. For single-rate compression, compression occurs in one shot. It is different from progressive compression where compression occurs incrementally. Diaz *et al.* (2005) produced an algorithm for compressing triangulated two-manifolds based on spanning tree. Spanning tree refers to protocol that ensures no loop occurs in the process of compression. Kalberer *et al.* (2005) proposed FreeLence that compress mesh data using valence coding to deal with triangle manifold mesh. Valence is referred to the number of edges incident on a vertex (Alliez and Gotsman, 2005). If a vertex is shared by 4 edges, the valence of the vertex is 4. Mamou *et al.* (2009) offered TFAN (Triangle Fan-based compression) that treats meshes with arbitrary topologies. Vasa and Brunnett (2013) revealed that by exploiting mesh connectivity with knowledge of vertex valence will form more accurate prediction in tangential direction using parallelogram prediction. It can be easily implemented in existing compression algorithms at different level of sophistication. Amjoun and Strasser (2008) proposed a new scheme by combining Predictive and Discrete Cosine Transform (PDCT) and establish local coordinate frame (LCF) where vertex is well predicted in the clusters. All these proposed algorithms mentioned is specifically for triangle mesh. Progressive mesh compression will be explained in streamable representation. Random accessible mesh compression will be defined in another subtopic. In mesh sequence compression, principal components analysis (PCA) and spatio-temporal prediction are broadly used in recent years. PCA is a way to reduce dimensionality of multivariate datasets (Jolliffe, 2002). Spatio-temporal prediction is used to predict the geometry filling a space and time required to compress a model (Erwig *et al.*, 1999). Amjoun and Strasser (2009) used local principal components analysis (LPCA) algorithm which clusters vertices based on local similarity between trajectories in coordinate system. This scheme provides improvement in compression ratio. Payan and Antonini (2005) presented clustered principal component analysis (CPCA). This algorithm use data-driven approach where it can identify mesh parts in animation. Vasa and Skala (2009) produced COBRA, an extension of dynamic mesh compression technique based on PCA. It is good for 3D moving representation. Rus and Vasa (2010) have analyzed the influence of vertex clustering on PCA-based dynamic compression by using Coddyac as a basic compression algorithm and combine it with cluster algorithm to demonstrate the performance of this approach. Cheng *et al.* (2010) proposed a novel scheme for 3D compression based on mesh segmentation using multiple principle plane analysis that results in good compression performance and reconstruction quality. Muller *et al.* (2005)], Wang *et al.* (2015), Stefanoski and Ostermann (2008), Bici and Akar (2011) and Ahn *et al.* (2013) used spatio-temporal prediction scheme as a based in their compression algorithm.

*Remeshing*

The idea of remeshing is simply about modifying mesh geometry and connectivity to produce better quality of mesh model (Francois and Cuilliere, 2000). It replaces an arbitrarily structured mesh by structured mesh. Alliez *et al.* (2008) made a survey about the development of remeshing technique over the past few years before 2008. They had classified the technique based on the goal, structure, compatibility, quality, feature and error-driven remeshing. Aghdai *et al.* (2012) introduced a new type of meshes named 567 meshes. These meshes are a closed triangle with each vertex has a valence of 5, 6 or 7. Valence of vertices will affect mesh processing algorithm. They have shown that any arbitrary closed triangle mesh with any genus will always can be remeshed to a 567 mesh. Their algorithm works in two phases, conversion of arbitrary meshes to 567 mesh and mesh refinement. Some remeshing algorithms are based on improving the geometry and some are removing irregularities or modifying the mesh to regular mesh to improve connectivity. Connectivity and geometry are not totally independent on each other. Using centroidal Voronoi tessellations (CVT) in remeshing algorithms, where the generating points of the tessellations are the centre of Voronoi regions (Du and

Wang, 2003) usually generate meshes with vertex valence of 6. Vidal *et al.*(2015) recently proposed 567-remeshing algorithm that locally retriangulates the mesh. Vertex valence, vertex budget and mesh fidelity are considered to improve remeshing performance in terms of regularization and connectivity compression. The main contributions are new local strategies for removing vertices valence lower than 5 and greater than 7, controlling mesh fidelity and preserve edges' feature during remeshing process.

*Random Access*

Random access allows mesh to be compressed and decompressed for selected parts only (Yoon and Linstrom, 2007). This helps to reduce the complexity and computation time. Choe *et al.* (2009) used cluster-based random accessible compression for segmentation of input mesh, using Angle Analyzer encoder to compress the mesh. Chen *et al.* (2009) also used segmentation algorithm and compress mesh with Edgebreaker algorithm. Yoon and Linstrom (2007) added random-accessible support to streaming mesh compression. Kim *et al.* (2010) extended Yoon and Linstrom's algorithm to compress mesh using bounding volume hierarchies composed of axis-aligned bounding boxes. Courbet and Hudelot (2009) proposed hierarchical representation based on vertex sequences. They use same line predictor as in Choe *et al.* (2009). Kim *et al.* (2006) used mesh refinement framework to produce multiresolution random accessible mesh compression algorithm. Maglo *et al.* (2013) encoder, POMAR generates discrete level of detail with half-edge collapse during compression. This scheme produces a smooth transition between the levels of details between the coarsest mesh to the finest one. Luffel *et al.* (2014) used Squad representation, a compact data structure by Gurung *et al.* (2011) to support on-the-fly streaming construction and processing. Gurung *et al.* (2011) proposed compact data structures such as Squad, Laced Ring (LR) and Zipper (Gurung *et al.*, 2013).

*Out-of-core representation*

Out-of-core representation approaches are including mesh cutting, vertex clustering, using external memory and streaming data (Ahn *et al.*, 2006). Mesh cutting is used to split large mesh into pieces so it can fit in the main memory. The pieces of mesh are processed separately while maintaining the boundary of each piece. However, this approach lowers the visual fidelity. Vertex clustering method is also proposed so the mesh data can fit in the main memory but the output complexity is still confined by the size. Pakhira (2010) has proposed an out-of-core Visual Assestment of Tendency (VAT) algorithm for very large data sets. Using external memory also result in time and space cost. Recent out-of-core representation uses streaming representation that has been formalized by Isenburg and Lindstrom (2005). Kot *et al.* (2005) created effective out-of-core run-time system which extending the memory utilization to the out-of-core level. It simplifies and streamlines time consuming application. Disk, network and memory hierarchy are utilized to achieve high utilization of computation resources.

*Streamable representation*

Isenburg and Linstrom (2005) defined streaming mesh as sequential insertion of indexed vertices and triangles based on the information about when vertices are introduced and finalized. Streaming mesh is compact if it is both vertex and triangle compact. Vertex-compact is a state where previous or next triangle in the stream refers to all vertices. Triangle-compact is where each triangle refers to previous or next vertex in the stream. Streaming mesh is a representation without providing information such as manifoldness, valence, and any other topological attributes of triangles. Allegre *et al*. (2007) used streaming algorithm for reconstructing closed surfaces of 3D models from large set of points based on geometric convection technique associates with 3D Delaunay triangulation. Bolitho *et al.* (2007) proposed a multilevel streaming for out-of-core surface reconstruction using Poisson-based reconstruction scheme and multilevel streaming representation. Luffel *et al.* (2014) used streaming representation to handle large meshes without storing the whole mesh data in memory. The data structure named Grouper. Construction of Grouper consists of streaming writer and streaming reader. Streaming writer reconstruct adjacency information for triangles and streaming reader sequentially reads Grouper stream and returns streaming mesh to the application.

## Contribution and Application

Those mentioned schemes are useful to improve the quality and performance of meshes. Some may be useful in compressing mesh so it can fit in main memory. Some could help in reducing computational cost. Others may help in producing high fidelity of mesh for better visualization. Polygonal meshes are very useful in data visualization for certain area such as medical application, architecture, geography, heritages and so on. It is also widely used in digital contents creation, games, e-learning etc.

## Conclusion

Compression and simplification are the basic scheme used to overcome highly complex polygonal mesh. Later, it is crucial to use remeshing process due to the differences in input mesh before mesh can be proceed to another process. Random access scheme, out-of-core representation and streaming mesh seem to be an emerging trend as they are addressing the most issues regarding polygonal mesh such as computational cost, time efficiency, data management and storage. We believe that further researches will focus on these approaches as they are starting to take place in these most recent years.

## References

Aghdaii, N., Younesy, H. & Zhang, H. 2012. 5–6–7 Meshes: Remeshing and analysis. *Computers & Graphics*. **36**(8): 1072-1083.

Ahn, M., Guskov, I. & Lee, S. 2006. Out-of-core remeshing of large polygonal meshes. *IEEE Transaction on Visualization and Computer Graphics*. **12**(5): 1221-1228.

Ahn, J. K., Koh, Y. J. & Kim, C. S. 2013. Efficient Fine-Granular Scalable Coding of 3D Mesh Sequences. *IEEE Transactions on Multimedia*. **15**(3): 485-497.

Allegre, R., Chaine, R. & Akkouche, S. 2007. A Streaming Algorithm for Surface Reconstruction. *SGP '07 Proceedings of the Fifth Eurographics Symposium on Geometry Processing*. 4 July, 2017, Switzerland.

Alliez, P. & Gotsman, C. 2005. Recent advances in compression of 3D meshes. *In*: Neil, A. D., Michael, S. F. & Malcolm, A. S. (eds.) *Advances in Multiresolution for Geometric Modelling*. Berlin: Springer Berlin Heidelberg.

Alliez, P., Ucelli, G., Gotsman, C. & Attene, M. 2008. Recent advances in remeshing of surfaces. *In*: Floriani, L. D. & Spagnuolo, M. (eds.) *Shape analysis and structuring*. Berlin: Springer Berlin Heidelberg.

Amjoun, R. & Strasser, W. 2008. Predictive-DCT coding for 3D mesh sequences compression. *Journal of Virtual Reality and Broadcasting*. **5**(6): 13pp.

Amjoun, R. & Strasser, W. 2009. Single-rate near lossless compression of animated geometry. *Computer-Aided Design*, **41**(10): 711-718.

Attene, M., Campen, M. & Kobbelt, L. 2013. Polygon mesh repairing: An application perspective. *ACM Computing Surveys (CSUR)*. **45**(2): A15.

Bici, M. O. & Akar, G. B. 2011. Improved prediction methods for scalable predictive animated mesh compression. *Journal of Visual Communication and Image Representation*, **22**(7): 577-589.

Bolitho, M., Kazhdan, M., Burns, R. & Hoppe, H. 2007. Multilevel streaming for out-of-core surface reconstruction. *Symposium on Geometry Processing*. 4-6 July, Barcelona, Spain.

Botsch, M., Steinberg, S., Bischoff, S. & Kobbelt, L. 2002. Openmesh - a generic and efficient polygon mesh data structure. *OpenSG Symposium*. January 202, Darmstadt, Germany.

Boubekeur, T. & Alexa, M. 2009. Mesh simplification by stochastic sampling and topological clustering. *Computers & Graphics*. **33**(3): 241-249.

Building 3D Primitive. 2014. Introduction to Computer Graphics (cs.edu.my). Accesssed on January 2015.

Chen, X., Golovinskiy, A. & Funkhouser, T. 2009. A benchmark for 3D mesh segmentation. *ACM Transactions on Graphics (TOG)*. **28**(3): 12pp.

Cheng, S. C., Kuo, C. T. & Wu, D. C. 2010. A novel 3D mesh compression using mesh segmentation with multiple principal plane analysis. *Pattern Recognition*. **43**(1): 267-279.

Choe, S., Kim, J., Lee, H. & Lee, S. 2009. Random accessible mesh compression using mesh chartification. *IEEE Transactions on Visualization and Computer Graphics*. **15**(1): 160-173.

Courbet, C. & Hudelot, C. 2009. Random accessible hierarchical mesh compression for interactive visualization. *Symposium on Geometry Processing*. 15 July 2009, Switzerland.

De Floriani, L., Kobbelt, L. & Puppo, E. 2005. A survey on data structures for level-of-detail models. *In*: Dodgson, N. A., Floater, M. S. & Sabin, M. A. (eds.) *Advances in multiresolution for geometric modeling*. Berlin: Springer Berlin Heidelberg.

Diaz-Gutierrez, P., Gopi, M. & Pajarola, R. 2005. Hierarchyless Simplification, Stripification and Compression of Triangulated Two-Manifolds. *Computer Graphics Forum*, **24**(3): 457–467.

Du, Q. & Wang, D. 2003. Tetrahedral mesh generation and optimization based on centroidal Voronoi tessellations. *International journal for numerical methods in engineering*. **56**(9): 1355-1373.

Erwig, M., Gu, R. H., Schneider, M. & Vazirgiannis, M. 1999. Spatio-temporal data types: An approach to modeling and querying moving objects in databases. *GeoInformatica*. **3**(3): 269-296.

Francois, V. & Cuilliere, J. C. 2000. 3D automatic remeshing applied to model modification. *Computer-Aided Design*. **32**(7): 433-444.

Gurung, T., Laney, D., Lindstrom, P. & Rossignac, J. 2011. SQuad: Compact representation for triangle meshes. *Computer Graphics Forum*. **30**(2): 355-364.

Gurung, T., Luffel, M., Lindstrom, P. & Rossignac, J. 2011. LR: compact connectivity representation for triangle meshes. *ACM Transactions on Graphics (TOG)*. **30**(4): A67.

Gurung, T., Luffel, M., Lindstrom, P. & Rossignac, J. 2013. Zipper: A compact connectivity data structure for triangle meshes. *Computer-Aided Design*. **45**(2): 262-269.

Isenburg, M. & Lindstrom, P. 2005. Streaming meshes. *IEEE Visualization 2005*. 23-28 October, 2005, Minneapolis, United States.

Jolliffe, I. 2002. *Principal component analysis*. John Wiley & Sons, Ltd, Hoboken, New Jersey.

Kalberer, F., Polthier, K., Reitebuch, U. & Wardetzky, M. 2005, September. FreeLence-Coding with Free Valences. *Computer Graphics Forum*. **24**(3): 469-478.

Kim, J., Choe, S. & Lee, S. 2006. Multiresolution random accessible mesh compression. *Computer Graphics Forum*. **25**(3): 323-331.

Kim, T. J., Moon, B., Kim, D. & Yoon, S. E. 2010. RACBVHs: Random-accessible compressed bounding volume hierarchies. *IEEE Transactions on Visualization and Computer Graphics*. **16**(2): 273-286.

Kot, A., Chernikov, A. N. & Chrisochoides, N. P. 2005. Out-of-core parallel Delaunay mesh generation. *17th IMACS World Congress Scientific Computation, Applied Mathematics and Simulation*. 25-29 April, 2005, Paris, France.

Li, L., He, M. & Wang, P. 2010. Mesh simplification algorithm based on absolute curvature-weighted quadric error metrics. *5th IEEE Conference on Industrial Electronics and Applications (ICIEA)*. 15-17 June, 2010, Taichung, Taiwan.

Luebke, D. P. 2001. A developer's survey of polygonal simplification algorithms. *IEEE Transactions on Computer Graphics and Applications.* **21**(3): 24-35.

Luffel, M., Gurung, T., Lindstrom, P. & Rossignac, J. 2014. Grouper: A Compact, Streamable Triangle Mesh Data Structure. *IEEE Transactions on Visualization and Computer Graphics.* **20**(1): 84-98.

Maglo, A., Grimstead, I. & Hudelot, C. 2013. POMAR: Compression of progressive oriented meshes accessible randomly. *Computers & Graphics*. **37**(6): 743-752.

Maglo, A., Lavoue, G., Dupont, F. & Hudelot, C. 2015. 3D Mesh Compression: Survey, Comparisons, and Emerging Trends. *ACM Computing Surveys (CSUR)*. **47**(3): A44.

Mamou, K., Zaharia, T. & Preteux, F. 2009. TFAN: A low complexity 3D mesh compression algorithm. *Computer Animation and Virtual Worlds*. **20**(2-3): 343-354.

Muller, K., Smolic, A., Kautzner, M., Eisert, P. & Wiegand, T. 2005. Predictive compression of dynamic 3D meshes. *IEEE International Conference on Image Processing (ICIP 2005).* 11-14 September, Genova, Italy.

Neperud, B. 2005. Visualizing mesh data structures and algorithms. *Proceedings of MICS*. 8-9 April, 2005, Eau Claire, Winconsin.

Pakhira, M. K. 2010. Out-of-core assessment of clustering tendency for large data sets. *2nd IEEE International Conference on Advance Computing Conference (IACC).* 19-20 February, 2010, Patiala, India.

Payan, F. & Antonini, M. 2005. Wavelet-based compression of 3d mesh sequences. ACIDCA-ICMI'2005. 5-7 November, 2005, Tozeur, Tunisia.

Peng, J., Kim, C. S. & Kuo, C. C. J. 2005. Technologies for 3D mesh compression: A survey. *Journal of Visual Communication and Image Representation*. **16**(6): 688-733.

Repair Defects in Meshes. 2010. Modelling (autodesk.com). Accesssed on March 2015.

Roy, M., Foufou, S. & Truchetet, F. 2010. Recent advances in multiresolution analysis of 3D meshes and their applications. IS&T/SPIE Electronic Imaging, International Society for Optics and Photonics. 17-21 January, 2010, California, USA.

Rus, J. & Vasa, L. 2010. Analysing the influence of vertex clustering on pca-based dynamic mesh compression. *In*: Lopez, P., Jose, F., Fisher, Robert B., Moeslund, Thomas B. (eds.) *Articulated Motion and Deformable Objects*. Berlin: Springer Berlin Heidelberg.

Serna, S. P., Stork, A. & Fellner, D. W. 2011. Considerations toward a dynamic mesh data structure. SIGRAD Conference. 17-18 November, 2011, Stockholm, Sweden.

Stefanoski, N. & Ostermann, J. 2008. Spatially and temporally scalable compression of animated 3D meshes with MPEG-4/FAMC. *IEEE International Conference on Image Processing (ICIP 2008).* 12-15 October, 2008, San Diego, USA.

Static Mesh (SM). 2014. Unreal Engine 4 Documentation (docs.unrealengine.com). Accesssed on March 2015.

Vasa, L. & Brunnett, G. 2013. Exploiting connectivity to improve the tangential part of geometry prediction. *IEEE Transactions on Visualization and Computer Graphics.* **19**(9): 1467-1475.

Vasa, L. & Skala, V. 2009, September. Cobra: Compression of the basis for PCA represented animations. *Computer Graphics Forum.* **28(**6): 1529-1540.

Vidal, V., Lavoue, G. & Dupont, F. 2015. Low budget and high fidelity relaxed 567-remeshing. *Computers & Graphics.* **47**: 16-23.

Wang, C., Liu, Y., Guo, X., Zhong, Z., Le, B. & Deng, Z. 2015. Spectral Animation Compression. *Journal of Computer Science and Technology*. **30**(3): 540-552.

Yoon, S. E. & Lindstrom, P. 2007. Random-accessible compressed triangle meshes. *IEEE Transactions on Visualization and Computer Graphics.* **13**(6): 1536-1543.

Zheng, Z. 2012. Simplification Algorithm for General Meshes. *Fifth IEEE International Joint Conference on Computational Sciences and Optimization (CSO).* 23-26 June, 2012. Harbin, China.

TRANSACTIONS ON SCIENCE AND TECHNOLOGY