Adaptive Vertex Quantization for Mesh Compression

Z. M. Qiu[1], Yoke San Wong[2] and Jerry Y. H. Fuh[3]

[1]National University of Singapore, mpeqzm@nus.edu.sg
[2]National University of Singapore, mpewys@nus.edu.sg
[3]National University of Singapore, mpeyhfuh@nus.edu.sg

ABSTRACT

With the increasing adoption of 3D CAD and pervasive 3D visualization in PLM environments, efficient 3D compression is highly desired to reduce the size of mesh models for fast transmission and efficient storage purposes. Among existing vertex compression schemes, vertex quantization uses integers to represent floating point vertex coordinates to achieve smaller model size. However, the current uniform quantization approach can be further refined since vertices bear various visualization significance to viewers and justify various compression strength. Inspired by earlier research on geometric simplification, this paper presents the development of an adaptive quantization scheme that differentiates vertices based on their visual significance and performs quantization at different bit length. The initial experimental results indicate that the proposed approach enhances the performance of previous uniform quantization approach while maintaining good visualization quality. The application of this approach in role-based visualization is also discussed in the concluding part of the paper.

Keywords: 3D CAD, mesh compression, adaptive quantization, visualization.

1. INTRODUCTION

Computer Aided Design (CAD) is the standard technology for mechanical designers to create 2D drawings or 3D solid models. While most CAD users (75%) still work in 2D environment, 3D CAD adoption is steadily expanding due to a plethora of features 3D CAD offers, such as complete modeling and design repurposing for downstream engineering activities. As a result, manufacturing enterprises are expected to handle more 3D models in their Product Data Management (PDM) databases.

3D CAD often takes place in a collaborative environment with sharing of the product data along the entire value chain share for better product design, manufacture, marketing and maintenance. As the most accurate form of product representation, CAD models are often repurposed to support downstream engineering or business activities. As a key aspect of CAD repurposing, effective 3D visualization for collaborative environment is highly desired and hence various lightweight 3D formats [1-4] have been developed to facilitate effective 3D visualization in distributed environments. Compared with native CAD formats, these lightweight 3D formats can reduce the size of models for faster transfer and visualization and thus are adopted in major Product Lifecycle Management (PLM) solutions, such as TeamCenter, ENOVIA, SmarTeam, etc.

Most aforementioned lightweight 3D technologies (a special case is 3DXML whose compression capability is based on re-tessellation of parametric surfaces) employ general compression schemes (like GNUZIP) to make models compact. However, their compression performance can be further enhanced by incorporating geometric compression technology.

In general, a 3D visualization model consists of vertices and triangles. Triangles can be geometrically compressed if a triangle can be represented with fewer vertices (less than 3). Efficient navigation schemes such as span-tree or triangle strip achieve the above purpose. Meanwhile, vertex compression relies on effective compression on vertex coordinates, which are mostly represented by floating point numbers.

This paper aims to enhance existing vertex compression techniques by proposing adaptive vertex quantization. Normal vertex quantization uses integers at fixed length to represent all coordinates and thus ignores the fact that some vertices have less impact on final visualization of 3D CAD models and hence can be represented with a lower resolution value in the form of number with a shorter bit length. An adaptive quantization mechanism exploits this attribute to improve the compression performance.

A normal quantization and the inspiration behind the proposed work are first discussed. Next, the key aspects of the adaptive approach are elaborated. Experiments and an application case are then given. Finally, future research work on enhancement of the proposed approach is suggested.

2. PRELIMINARIES

In geometric compression domain, vertex quantization is an approximate procedure to represent vertex coordinates, of usually 32- or 64-bit floating point numbers, as integers to reduce space consumption. Vertex quantization can be combined with other compression schemes, such as predictor-based vertex encoding and span-tree-based mesh connectivity encoding, to achieve a higher degree of mesh compression. Literature [12] gives a survey on current geometric compression technologies.

The procedure for quantizing mesh model M involves the following steps:

(1) Establish bounding box *B* of *M*.

(2) Determine n , the bit length for vertex coordinate.

(3) Establish a 3D grid which partitions *B* into $2n$ sections along X, Y or Z axis.

(4) Align each vertex of *M* with the closest grid intersecting point.

After the above procedure, a vertex can be approximately positioned by its new relative coordinates - an integer tuple (i, j, k) in the grid system. The corresponding absolute coordinate (x, y, z) can be restored by:

$$\begin{aligned} x &= x_0 + \frac{i \cdot a}{n} \\ y &= y_0 + \frac{j \cdot b}{n} \\ z &= z_0 + \frac{k \cdot c}{n} \end{aligned} \tag{2.1}$$

Here (x_0, y_0, z_0) referring to the minimum coordinates on X, Y and Z axes, and (a, b, c) referring to dimensions of *B*.

Based on uniform partition, the above quantization approach is able to reduce storage space for vertex coordinates several times smaller. Although effective, this approach has room for improvement because in reality the number of vertices can be appropriately adjusted according to resolution requirements in the final views of models. And this was first exploited by Rossignac who developed vertex-merging [7], which is a simplification method for rendering complex 3D scenes. In this method, a 3D grid is imposed onto a mesh model to cluster mesh vertices into grid cells. A 2D example is illustrated in Fig. 1. The over-tessellated 2D mesh, shown in Fig. 1(a), is imposed on a 2D grid. Vertices in each cell are merged into one vertex based on their visual significance. The survived vertices form the simplified mesh, as shown in Fig. 1(b). It should be noted that the simplification process may change the contour of the mesh.

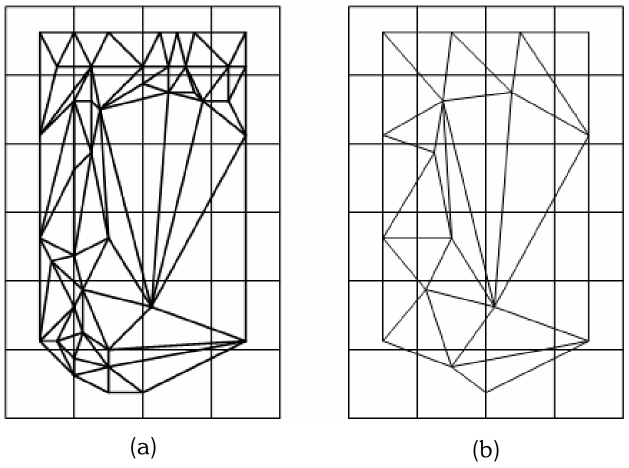


Fig. 1: Vertex merging on 2D mesh.

In a cell, vertices are ranked in importance against a criterion for visual significance. After ranking, the corresponding cell is replaced with the top-ranked vertex in the cell. This replacement process is applied to all cells and the entire model is significantly simplified after losing a large number of vertices. This method may require a post-merging stage to repair destroyed topologies. Later, the ranking scheme was refined by Low [8]. In the above works, the major ranking criterion is the model silhouette change caused by removal of a vertex.
A vertex causing small silhouette change bears low significance. Two specific factors for the ranking are: (1) the inverse of the maximum angle between all pairs of incident edges on the vertex and (2) the length of the longest edges among all others incident on the vertex. The ranking function is as follows:

$$R(v) = a \cdot \cos\frac{\theta}{2} + (1-a) \cdot \frac{d(v)}{d_0} \tag{2.2}$$

Here the coefficient $a \in [0,1]$. If $a = 0$ or $a = 1$, the above ranking function includes one factor from the above two. The number $d(v)$ denotes the length of the longest edges incident on vertex v and d_0 denotes the longest edge in the mesh.
The above suggests a possible enhancement to existing uniform quantization approach, whereby vertices could be differentiated based on their visual significance and thereafter quantized at different bit length.

3. ADAPTIVE QUANTIZATION

Adaptive quantization uses more bits for important vertices. The ranking procedure is performed for all model vertices for subsequent bit assignment process. Thereafter, quantized coordinates are written in proper sequence to a file for storage and transfer.

3.1 Vertex Ranking

The ranking method discussed in section 2 is designed for geometric simplification and thus needs modification for CAD model visualization. The role of the vertex ranking procedure in vertex merging has limited impact on the overall simplification process since (1) the ranking takes place in a cell which has local effect on the final model, and (2) the simplification aims at tolerable silhouette change on a 2D display device.
Meanwhile, the priority on CAD visualization is accurate design intent delivery. Hence, minute features, which may be under-ranked due to low geometric significance, are vital to convey design concept and frequently manipulated (rotation, zoom-in, measurement, etc) by human viewers. Therefore, the ranking scheme should be re-developed for CAD visualization.
In CAD applications, models are usually displayed as wireframe, hidden line removal (HLR) surface model, shaded solid model, and shaded solid model with edge, etc. Among these modes, shaded solid model with edge (SE) mode is the most common one since SE helps human viewers to recognize 3D shapes and features. The resolution of surface edges significantly affects the perception of CAD models since they define the skeleton of model topology. In mesh models, edges are represented as trimming lines. Intuitively, vertices on trimming lines are more important in conveying design features and thus need more bits.
In addition to identifying the incident trimming lines of a vertex, it is also useful to examine the distance between the vertex and its neighboring vertices. With a long distance, any minute position change on the vertex, introduced by quantization is less likely to distort the design features behind the vertex. If all incident edges of a vertex have similar lengths, a small change on the vertex's position will not affect the perceived shape related to the vertex. Figure 1 illustrates this observation. Hence, the ratio between the shortest or longest incident edges of a vertex can indicate the local shape's sensitivity to the vertex fluctuation.
The following ranking function combines the above two factors and evaluates a vertex v as:

$$R(v) = a \cdot S(v) + (1-a) \cdot \frac{d_1(v)}{d_0(v)} \tag{3.1}$$

Here $S(v) = 1$ if *v* resides on a trimming line or 0 if not. Distances $d_1(v)$ and $d_0(v)$ are associated with the shortest and longest edges incident to *v*, respectively. The coefficient *a* is larger than 0.5 because it is necessary to ensure vertices incident to trimming lines are assigned high ranking values.

3.2 Bit Assignment

After the vertex ranking procedure, each vertex is assigned with a number of bits based on its ranking value. There are several ways to perform the assignment task:

(1) Proportional assignment. The bit length *n* for vertex *v* is proportional to its ranking value. Specifically, $n(v) = \left\lceil \frac{R(v) \cdot n_0}{R_0} \right\rceil$ where n_0 denotes the maximum length and R_0 the maximum ranking value among all vertices.

This approach inevitably imposes difficulties on the reverse-quantization process at the viewing client since the system needs to memorize the bit length for all vertices.

(2) Staged assignment. Vertices with similar ranking values are grouped together and assigned with a bit length.

(3) Budget assignment. A desirable feature for vertex compression is quality control, which can be facilitated by setting a bit budget for an overall mesh model. Staged assignment can support this mode and thus is suitable for development in practical environments.

The entire procedure for budgeted stage assignment consists of the following steps:

(1) Rank vertices based on an evaluation function

(2) Group vertices in similar ranking values. Although there are various sophisticated clustering algorithms available, an effective 4-group scheme is as follows:

   (a) Create group A with all vertices incident to any trimming lines

   (b) Create group B with the remaining vertices

   (c) In group A, find two adjacent vertices v_i and v_{i+1} so that the number $R(v_i) - R(v_{i+1})$ is maximized

   (d) Split group A into two groups $(v_1,...,v_i)$ and $(v_{i+1},...,v_n)$ where v_n is the vertex with the lowest ranking value

   (e) Split group B in similar manner

   This scheme is very easy to implement in practice.

(3) Assign each group with a suitable bit length to keep total bit length under bit budget N . The value of N might be determined by the size of mesh models and the specific requirements from applications concerned, e.g., the measuring accuracy and visualization quality. The bit length for group i , n_i , is determined as $\left\lfloor \frac{N \cdot r_i \cdot m_i}{\sum \cdot r_i \cdot m_i} \right\rfloor$ where r_i and m_i refer to the average ranking value for group i and vertex number in group i , respectively.

(4) Record down the assignment pattern for vertex restoration at viewing client side via the format: (category number bit length vertex number bit length vertex number bit length....)

(5) At viewing client side, the quantized vertex coordinates can be restored with the assignment pattern, which takes up small amount of storage space.

3.3 Entire Procedure

At the viewing client side, the entire procedure to visualize a vertex-quantized mesh model is illustrated as follows:

(1) Receive the main data block of the model

(2) Retrieve 3D-grid-related parameters, i.e. $(x_0, y_0, z_0, a, b, c, n)$

(3) Retrieve bit assignment pattern data block, i.e. $(4, n_1, m_1, n_2, m_2, n_3, m_3, n_4, m_4)$

(4) Create an empty vertex list with length $m = m_1 + m_2 + m_3 + m_4$

(5) Retrieve a n_1 -bit integer from the vertex data block and convert it into a floating point number via the formula $x = x_0 + \frac{ia}{n}$ to form the x coordinate for the current vertex; repeat the step two times to compute y and z coordinates and put the completed vertex coordinates into the vertex list

(6) Repeat the above step m_1 times to restore all vertices in group 1.

(7) By applying parameters (n_i, m_i) where $i = 2, 3, 4$, coordinates of the vertices in groups 2, 3 and 4 are restored and saved in the vertex list.

(8) The connectivity information is retrieved and combined with the vertex list for rendering the entire mesh model

It should be noted that the entire procedure only requires a single pass of the raw vertex data block and thus can run very fast.

The 4-group schema is selected in this work since it is straightforward to implement. It is however reasonable to explore other bit assignment methods. Benchmarking is needed to determine which assignment method is optimal for specific 3D CAD models. Hence, it is future work to develop more advanced assignment methods than the current 4-group one.

4. RESULTS

The adaptive quantization approach is evaluated in terms of (1) compression ratio (2) visualization fidelity (3) compression speed, and (4) model loading speed. The result of the above four comparisons is given in Tab. 1.

The model illustrated in Fig. 1 is a sheet metal object and hence all its vertices are located in trimming lines. This extreme example is chosen to demonstrate the approach in a worst case. For models with a considerable number of vertices inside surfaces, the compression ratio is higher due to more aggressive quantization on these internal vertices. With uniform quantization, all vertices are assigned with 16 bits and thus a compression ratio of 50% is achieved, since the original single-precision floating point representation requires 32 bits for each coordinate. Under adaptive scheme, all vertices are allocated into two groups, according to the four-stage grouping method given in section 3.2. By applying the ranking function as according to Eq. 3.1, these vertices are further distributed into two groups (one with the upper third of top-ranked vertices and the other with the rest of the vertices). The overall compression ratio is 41%, as illustrated in Tab. 1. The compression and loading processes for both schemes are of little difference in speed.

Scheme	Model profiles	Compression ratio	Compression time	Loading time
Uniform	triangles: 520 overall vertices: 321 vertices on edges: 321 original bit length: 32	bit length: $16(100\%)$ ratio: 50%	0.1ms	0.1 ms
Adaptive		bit length: $16(\frac{1}{3})-12(\frac{2}{3})-0-0$ ratio: $41\%=\frac{1}{3}\times\frac{16}{32}+\frac{2}{3}\times\frac{12}{32}$	0.2 ms	0.15 ms

Tab. 1: Compression result.

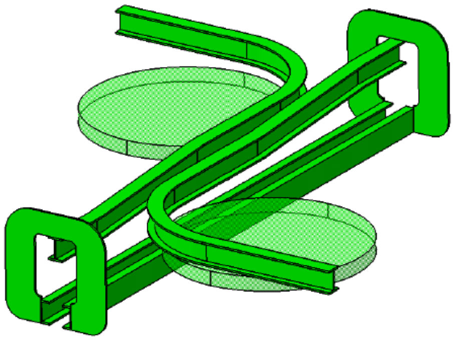
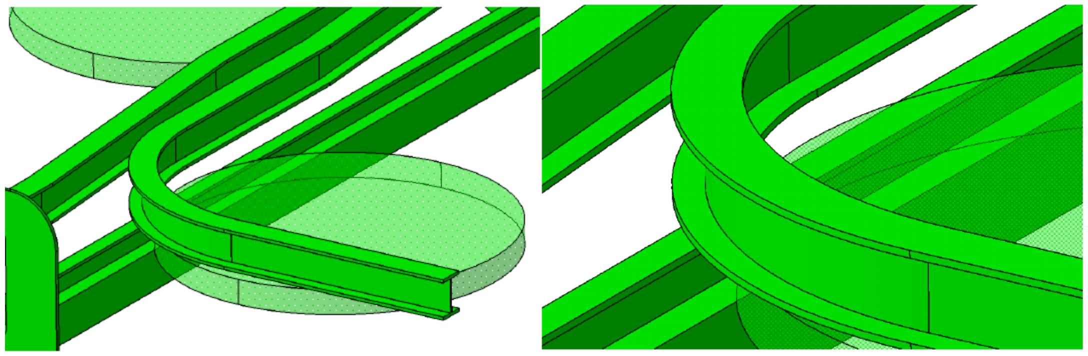


Fig. 2: The fit entire view of the uncompressed model.

The full view of the model concerned is shown in Fig. 2. Two zoom-in views for the adaptively quantized model are illustrated in Fig. 3(a) and Fig. 3(b), respectively. It can be seen that the quantization does not compromise the visualization of the model.
The experiment has been conducted using a normal IBM ThinkPad T41 notebook with 512MB memory. The compression and loading programmes are written in C++. Models are rendered as VRML objects in CATIA V5.



(a) (b)
Fig. 3: The zoom-in views for the adaptively compressed model.

5. APPLICATION

An interesting application of adaptive quantization is role-based visualization (RBV) [9–11], a technology to reduce 3D CAD details according to the role of a viewer. For example, a designer is modifying P1, a component of assembly A = (P1, P2, P3). The designer sets P1 to be provided as a full CAD model, P2 an accurate mesh model for fine measurement and dimensioning, and P3 a mesh model merely for visualization. The vertex coordinates in P2 should be represented with reasonable precision to support measurement and dimensioning, while P3 could afford reasonable loss of coordinate precision. Hence, quantization-based compression can be applied to P2 and P3 with different bit length arrangements.

As an example, both P2 and P3 have 360 vertices. Instead of representing their vertex coordinates with 32-bit floating point numbers, quantization is applied to them at 16 and 8 bits, respectively. Hence, P2 needs 17280 (360 × 3 × 16) bits while P3 needs 8640 bits. Compared with 34560 bits required for full floating point representation, a significant amount of storage space is saved.

Compression of mesh models can be carried out at different times and locations. In conventional CAD design scenarios, CAD models are tessellated at local design clients which are therefore a natural location to perform the compression. In collaborative PLM environments, compression of models can be performed at the underlying PDM server end since it is the place to host data for user access. The server has a plethora of idle CPU time which can be utilized for model compression in an economical manner since the server is assumed to have more computation power than local design clients.

Dynamic data provision scheme for role-based visualization [11, 13] can serve as an effective way to present requested models to users. Under such a scheme, the PDM server dynamically generates the entire assembly with different component compression ratio based on the requester's privileges. Hence, the user can modify, review or measure components with appropriate vertex precision.

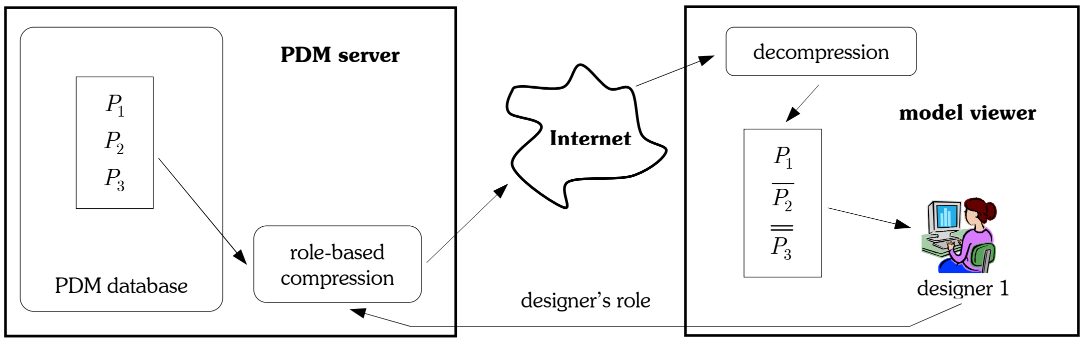


Fig. 4: Dynamic data provision for role-based geometric compression.

The mechanism for dynamic provision is illustrated in Fig. 4. A PDM server hosts a PDM database and several software components. At the viewer end, a designer, designer 1, is assigned with a role to access the PDM database with defined privileges. In a collaborative context, the designer has different level of access to the PDM database according to his role in the collaboration. An assembly, which consists of three components, is stored in the PDM database. Component P_1 is controlled by designer 1. Hence, designer 1 can access the full CAD model of P_1; meanwhile, he can access the other two models in detail-reduced forms for visualization.

When designer 1 requests the assembly from the PDM database, the role-based compression module at the PDM server delivers component P_1 in its original form to designer 1 while compresses P_2 and P_3 into $\overline{P}_2$ (quantized mesh in long bit length) and $\overline{\overline{P_3}}$ (quantized mesh in short bit length), respectively. Therefore, designer 1 can modify P_1, perform accurate measuring and dimensioning at P_2, and view P_3 with good visualization resolution. With such a central data provision mechanism, enterprise 3D data can be protected from undesired access from various roles in product collaboration chain.

The bit length for P_2 and P_3 is determined by the project manager or PDM administrator based on specific requirements in collaboration. The requirements are usually specified before the collaboration begins.

In this example, only vertex quantization is applied to reduce the size of 3D models for efficient visualization. However, geometric simplification is another way to achieve the same purpose, as introduced in literature [13]. With geometric simplification, a 3D model can be visualized faster with minor fidelity loss. The downside of a simplified CAD model is poor visualization details when the model is viewed in close distance. If the fidelity loss is significant, the viewer may undergo difficulties to capture the design intents of the model. In general, geometric simplification is used for either over-tessellated 3D models (such as scanned objects) or remote models that do not need fine renderings. For mechanical CAD models, which are usually properly tessellated, geometric compression is almost the only option to shrink model size for fast transmission.

With adaptive quantization, mesh vertices are ordered to indicate their individual bit lengths. The vertex order, however, also indicates the compressed connectivity in span-tree based compression schemes, such as EdgeBreaker [14]. Therefore, the connectivity information should refer each vertex as the vertex's index in the vertex list instead of the vertex's coordinates. For instance, the connectivity data set (such as a triangle strip) is represented as (…index(v_i),index(v_{i+1}),index(v_{i+2}),…) where vertices v_i , v_{i+1} and v_{i+2} form a triangle in the current triangle strip. It should be noted that the index-based representation may suggest larger model size when the triangle count is large (according to analysis in literature [14]). However, for a mechanical CAD part, the triangle count is often less than 500. The complexity of mechanical CAD models lies in the overall assembly structures, not individual design parts.

6. SUMMARY

This paper explores adaptive quantization for mesh vertex compression. Compared with the previous uniform quantization approach, adaptive quantization enables more aggressive compression with bit budget control. The proposed approach identifies vertices that are of less importance to final viewers and compress them more intensively. The entire procedure is fast (rendering $O(n)$ complexity with n vertices) and easy to realize in practical CAD/PDM environment. Initial experiments show that adaptive quantization can improve compression ratio without loss of required visualization quality. It also delivers good runtime performance on the viewing client.

Adaptive quantization can be conveniently realized with existing mesh connectivity compression methods. It also bears good potential to support interesting applications, such as role-based visualization. The approach however is subject to further refinements in the following ways.

Firstly, with the current form of the approach, the vertices inside trimmed surfaces are aggressively compressed and hence experience more position fluctuation. Although these internal vertices are not explicitly displayed in the final views, they affect the shading process: the normal vectors associated with these internal vertices are used by the lighting mechanism in the underlying rendering programme. Hence, it is preferable to make appropriate adjustments on the normal vectors of all aggressively-quantized vertices so as to minimise the change on the surface shading result.

Secondly, the four-state grouping mechanism can be replaced with more sophisticated approaches. If effective methods are found to support compact representation of complex bit assignment process, the adaptive quantization method could deliver more aggressive compression capability.

The third enhancement could be optimal computation of the bit budget N for various application scenarios. It is desirable to investigate the quantitative relationship between the reduced coordinate accuracy and the visualizing/measuring/analyzing requirements so as to guide the application of the proposed method in real applications.

As an effort to further investigate the proposed study, the future work is to apply adaptive quantization on a more complex assembly model under role-based visualization scenario.

7. REFERENCES

[1] CADigest: your online CAD digest, http://www.cadigest.com/, CADigest

[2] 3DXML: a universal, lightweight XML-based format, http://www.3ds.com/3dxml, Dassault Systemes

[3] JT Open: 3D visualization for engineering, http://www.jtopen.com/, UGS

[4] Universal 3D Format, http://www.intel.com/technology/systems/u3d/, Intel

[5] OpenHSF: lightweight 3D visualization and streaming, http://www.hoops3d.com/, HOOPS 3D

[6] Isenburg, M.; Triangle Strip Compression, Computer Graphics Forum, 20(2), 2001, 91–101.

[7] Rossignac, J.; Borrel, P.: Multi-Resolution 3D Approximations for Rendering Complex Scenes, in: Modeling in Computer Graphics, B. Falcidieno and T. L. Kunii, Springer-Verlag, 1993, 455–465.

[8] Low, K.-L.; Tan, T.-S.: Model Simplification Using Vertex-Clustering, in: 1997 Symposium on Interactive 3D Graphics, Providence RI USA, 1997, 75–81.

[9] Cera, C. D.; Braude, I.; Kim, T.; Han, J.; Regli, W. C.: Hierarchical role-based viewing for multilevel information security in collaborative CAD, Transactions of the ASME, Journal of Computing and Information Science in Engineering, 6(1), 2006, 2–10.

[10] Cera, C. D.; Kim, T.; Han, J.; Regli, W. C.: Role-based viewing envelops for information protection in collaborative modeling, Computer Aided Design, 36(12), 2004, 873–886.

[11] Qiu, Z. M.; Fuh, J. Y. H.; Wong, Y. S.; Secure CAD model retrieval and data consistency: issues in role based visualization, Computer Aided Design & Applications, 3(1-4), 2006, 139–145.

[12] Alliez, P.; Gotsman, C.: Recent advances in compression of 3D meshes, Advances in multiresolution for geometric modelling, N. A. Dodgson and M. S. Floater and M. A. Sabin, Springer-Verlag, 3-26, 2005.

[13] Qiu, Z. M.; Kok, K. F.; Wong, Y. S.; Fuh, J. Y. H.; Role-based 3D visualization for asynchronous collaboration, Computers in Industry, 2007, in press

[14] Rossignac, J.: Edgebreaker: connectivity compression for triangle meshes, IEEE Transactions on Visualization and Computer Graphics, 5(2), 1999, 47-61.