

Light-Reflection Analysis Method for 3D Surface Damage Identification

Michał Turek and Dariusz Pałka

Abstract The article introduces a new high resolution 3D mesh comparison method that can be used for 3D object surface analysis. A need of 3D mesh analysis is evident in many procedures in engineering, medical or strictly graphical applications. A typical 3D object scanning process produces a high resolution 3D triangle mesh describing the surface of the object. Surface analysis based on this kind of material is usually complex or inaccurate, because each 3D mesh vertex must be identified, positioned and analyzed. A solution proposed in the paper is focused on using native 3D mesh rendering processes for mesh analysis, especially in the surface damage identification field. 3D graphical acceleration hardware and Pixel/Vertex Shaders technology will be used to prepare sets of 2D images—generated with natively 3D accelerated but specially modified light reflection rendering technique. Images then will be analyzed by comparing their 2D reflections with correct object patterns to find any damage-caused differences. The method has proven very quick to calculate and easy to apply; the test applications were programmed over a standard PC 3D accelerated graphical modules. It can also be flexibly applied, which allows for analyzing only a part of the 3D object surface if needed. Additionally, it can produce very accurate results without any precision lowering mathematical-model assumptions, commonly met and usually necessary to apply in typical 3D triangle mesh analysis.

Keywords Surface damage identification · 3D mesh analysis · Light reflection rendering

M. Turek (✉) · D. Pałka

AGH University of Science and Technology, 30 Mickiewicza Av., 30-059 Krakow, Poland
e-mail: mitu@agh.edu.pl

D. Pałka

e-mail: dpalka@agh.edu.pl

© Springer International Publishing AG 2017

J. Świątek et al. (eds.), *Information Systems Architecture and Technology: Proceedings of 37th International Conference on Information Systems Architecture and Technology—ISAT 2016—Part III*, Advances in Intelligent Systems and Computing 523, DOI 10.1007/978-3-319-46589-0_7

1 Introduction

Surface analysis is a widely known problem in computer graphics applications. Nowadays highly developed 3D scanning methods can produce advanced multi-point 3D mesh structures, which should be analyzed and compared with patterns. A vast number of triangles and vertices which need to be processed in 3D scanning can cause huge computation complexity problems. Direct 3D triangle mesh calculations are very complex here, so other methods of analysis need to be developed. And if there is a ready comparison pattern also encoded into 3D triangle mesh, much more calculation is needed to perform this process [1, 2]. The concept of work described in the paper assumes the use of a “native graphical 3D object presentation” as a mid-result passed between two phases of surface analysis:

- Mathematically ideal 3D rendering based on native “light reflection like” processing (3D mesh is rendered into 2D image with light-reflections processing that highlights all surface imperfections)
- Domain-dependent 2D image processing (either a cross-section of an image or full 2D image analysis) is used to calculate surface imperfection metric values and judge their fitness.

The approach will enable us to perform a quick object damage analysis with easy precision shifting (zoom-like). It is focused on vast object surface anomaly patterns recognition, not per-vertex calculations. Comparison patterns can be expressed with any method—using 2D pixel-map imaging (direct image comparison) or functional relations (2D image cross-sections as a mid-result will be generated in this particular approach). The following basic assumptions are established:

- Medical applications for the method should be considered a priority, especially CT data analysis (for example, endoprosthesis).
- An entry point for the analysis process will be 3D triangle mesh representing the examined 3D object (created using commonly known methods [3]).
- Mesh resolution or triangle positioning cannot affect the final result as long as it represents a 3D object.

The great advantage of the proposed approach is the use of GPU-accelerated native rendering procedures to blend-in any mesh-specific data (too many triangles, different triangles passing in the mesh describing the same object, etc.)

2 3D Rendering Method

There was a need to develop a 3D rendering engine, which can be used to produce a light-reflective mesh image. The first experiments were performed using a standard OpenGL 3.0 smoothing features (for generating blended per-triangle light-reflection images). However, these attempts gave poor results, and there was no way to

control and customize the rendering process, having every pixel value (of the generated image) calculated correctly. Therefore, a full light-reflection generation process have been developed, based on a 3D Vertex Shader and Pixel Shader collaboration. Vectors called mesh vertex-normals are a base for any calculations here. They are assigned to each vertex in a mesh. To calculate a vertex normal, it is necessary, first, to have a vector perpendicular for each triangle adjacent to that vertex, then, to calculate average values of those perpendicular vectors for their x, y and z components, and finally, to combine them into a new vector, that is a mesh vertex-normal.

Mesh vertex-normals are commonly generated as an addition to a 3D model expressed with a 3D triangle mesh set and will be required by shader transformations. These vectors will be transferred together with vertex position vectors into shader programs as a non-uniform values (via a vertex profile). Inside a shader they will be used to compute the values of so-called vertex light reflections. Vertex light reflection is a vector product of a light direction vector and a vertex normal vector. After the vertex light-reflection computation, it is passed to Pixel Shader via a shader profile. Before it gets to the pixel processing engine, it is interpolated between light-reflection values of two other triangle vertices. Finally, a correct pixel-reflection value is passed to pixel Shader and then further to output 2D image. Any surface (caused by a vertex position shift) will now be exposed. The procedure of such rendering will be extremely fast, because it is natively accelerated by GPU hardware. A test implementation was made in nVidia CG (C for Graphics) shaders (over shader profile 2.0) [4, 5]. The communication between Pixel Shader (in a CG API called—Fragment Shader) and Vertex Shader has been established using extended texture coordinates data passed via a vertex profile [6]. This is a common approach in such cases. Typical implementation of per-pixel shading procedure is based on cooperation between vertex shading and pixel shading programs. Light-reflection vector (calculated in Vertex Shader with material and light features consideration) is passed to Pixel Shader via shader environment profile. During this passing, vector interpolation is calculated each time for particular 2D pixel. Since a “light source” in the proposed method is fixed and single, there is a chance to simplify light-reflection calculations. The goal is to detect surface damage, shown as a normal vector values disturbance spread in the vast number of vertices. A typical per-pixel surface shading technique (with a multi-light reflection color emulation involved) is not needed here. Instead, a vital result needed is just a shaded damage graphical raster. Therefore, a vertex shader procedure can be altered to generate interpolated light reflections based on vertex normals only. Now, a final shaded pixel color for rendered 3D object will express a 3D surface disturbance (possible damage), and not a surface color adjusted by lighting configuration.

Going into details, a vertex normal vector is being normalized first to have all components fixed in $[0..1]$ range. Next, since it is a 3D vector, its x, y and z components are mapped into a new vector inside outbound shader profile (a vertex texture coordinate field was used in this profile). In vertex shader/pixel shader collaboration an additional vector (texture coordinate) will be interpolated for each pixel and passed into a pixel shader. Now, a pixel shader comes to action. It

performs a final pixel colorization using a former vertex normal vector (interpolated) instead of typical pixel shading colorization input values (natural light reflection for a particular surface material color described by diffuse, ambient and specular values). As a result, a precise graphical view of picture deformation is created, with adjustable precision level (zoom) and geometric damage highlights expressed in pixel colors. The solution is also quick, putting lots of calculations in a shader layer and making explicit vertex-set calculation unnecessary.

3 An Example of Application

The usefulness of the method presented above can be shown by the example from the domain of medical images. The example focuses on detecting surface deformations of the mandibular condyle. The analysis of damages of this surface is an important factor in the diagnosis of temporomandibular joint syndrome (in short TMJ syndrome) [7]. This surface can be analyzed on the basis of cross-sections of a 3D bone model obtained from CT pictures. However, the interpretation of these cross-sections, as is the case with the interpretation of other medical images made by humans (even if they are highly-qualified physicians), can be quite difficult, as it was pointed out in [8]. This is a direct result of both vast number of images (cross-sections) to be interpreted and the level of their complexity.

So, this example uses an alternative approach to the problem of detecting damages on the surface of the mandibular condyle. As stated above, the first stage of the presented method is the preparation of a 3D triangle mesh representing a 3D object (in this case a mandibular condyle). It is obtained by the following steps:

- **CT Data Acquisition.** The data used for the analysis of degenerative changes on the bone surface of the mandibular joint are obtained from a CT scanner with voxel dimension $0.4 \times 0.4 \times 0.4$ mm.
- **Surface 3D Construction.** Marching cubes algorithm [3] is used to create a polygonal representation of an ISO surface of a 3D scalar field from a CT image set. As a result, a 3D model made up of triangle facets is obtained.

A diagram presenting the process described above is presented in Fig. 1.

Figure 2 shows the surface of a 3D model created in the “Surface 3D Construction” step on the basis of the CT data using marching cubes algorithm. Next, the presented method of damage identification can be used for semi-automatic detection of the surface of the mandibular condyle deformation.

The first way in which the presented method can be used is to semi automatically detect the mandibular condyle deformation, which can indicate TMJ syndrome.

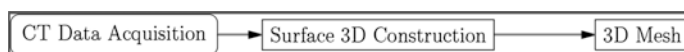
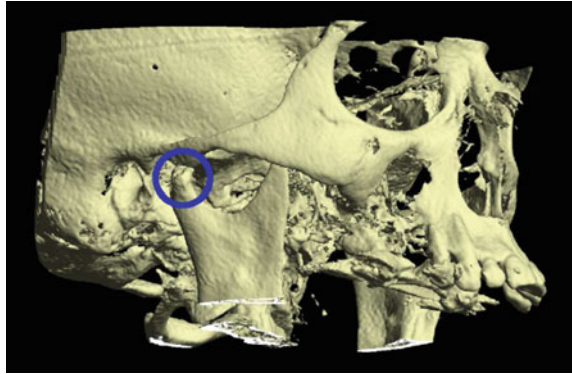


Fig. 1 Key points in the process of the analysis of surface damages

Fig. 2 The 3D model created using the marching cubes algorithm. The *marker* shows the mandibular condyle



In this case, the process consists of the following steps:

- The threshold of the permissible surface deformation level has to be set. This should be done by the domain expert (for example, a physician) by selecting the referential case of the mandibular condyle deformation. Then, every examined case with the surface deformation above this threshold will be treated as a potential cause of TMJ syndrome. After selecting the referential case, the zoom is used to set the referential distance from the surface to the observer for which the vertex-normal vector field is constant (which is represented by the same color of rendered pixels) due to average normal vectors observed from a given distance.
- For each examined case (the surface of the mandibular condyle), the comparison with referential case is performed. The color representation of the vertex-normal vector field is rendered from the referential distance. If the rendered surface color representation is the same as in the referential case (i.e. the vector field seen from the referential distance is constant), the mandibular condyle surface is below the threshold and is not considered the cause of TMJ syndrome. If, however, the rendered color representation of the vertex-normal vector field differs from the referential one, the surface deformations are above the threshold and can be potential causes of TMJ syndrome.

The difference in the color representations of the vertex-normal vector fields for referential and examined cases can be automatically detected using a standard 2D image comparison methods.

Figure 3 presents the mandibular condyle surface rendered with the use of a standard OpenGL procedure and the color representation of the vertex-normal vector field.

Figure 4 presents the color representation of the vertex-normal vector field for three sample distances from the surface to the observer. The distance value unit is the number of pixels from the camera to the nearest surface element.

The second way of using the presented method for detecting deformations of the object surface is the comparison of two CT images of the same object registered in

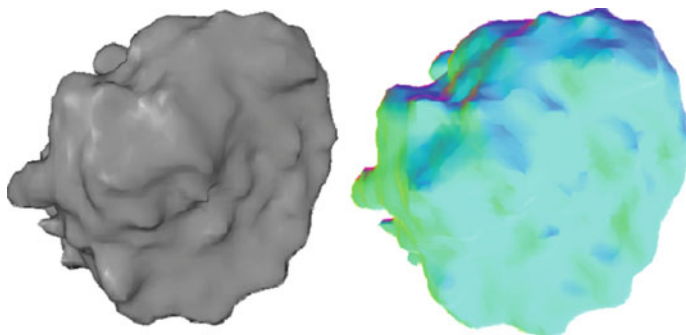


Fig. 3 The mandibular condyle surface. The image on the *left* is rendered with the use of a standard OpenGL procedure, while the image on the *right* is color representation of the vertex-normal vector field

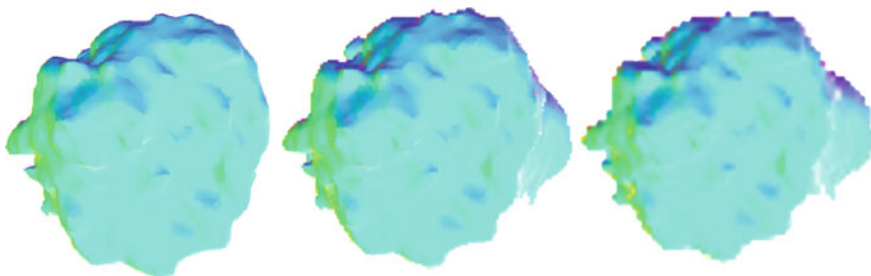


Fig. 4 The color representation of the vertex-normal vector field rendered for three distances—from *left*: distance 50, distance 250, distance 550

two different points in time (for example, the ones registered before and after the treatment). In such case, the comparative analysis of the surface can be provided. The process is as follows:

- The viewpoints for both images should be fitted (relative to the examined surface)
- The color representation of the vertex-normal vector field is rendered using the method presented in this paper
- The difference between the vertex-normal vector fields obtained is calculated. This can be done by a simple subtraction of two images with the color representation of the normal vector fields.

Figure 5 presents the examples of the color representation of the normal vector field of the mandibular condyle surface: original and deformed. The viewpoints in the both images are fitted.

Figure 6 demonstrates the comparison of the original and the deformed mandibular condyle. This figure is obtained as a result of subtracting color representations for normal vector fields. The pixel resultant color in the RGB model is

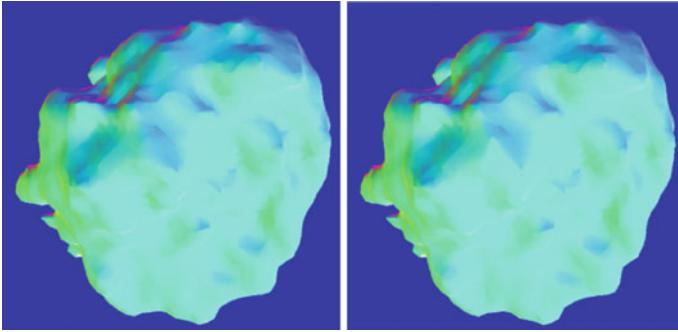


Fig. 5 The color representation of the vertex-normal vector field for the mandibular condyle: *left image*—original, *right image*—deformed

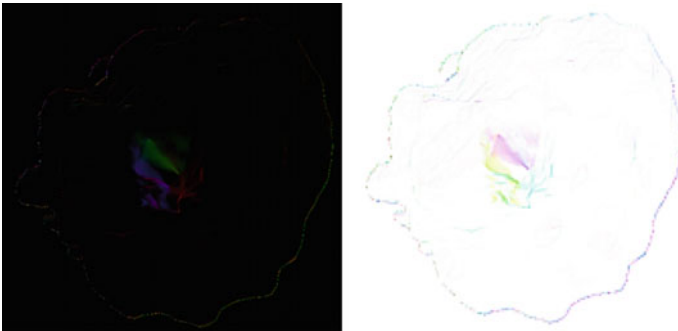


Fig. 6 The result of the subtraction of the vertex-normal vector field for the original and the deformed mandibular condyle *left image*—subtraction, *right image* inverted colors of subtraction

calculated as an absolute value of the difference between red, green and blue components of pixel colors from the left image of Fig. 5 and from the right image of Fig. 5. For better visualization, the color results of subtraction have been normalized.

As can be seen in Fig. 6, the presented method allows for quick visual comparisons of two surfaces (for example, an original and a damaged one). This approach can be useful, for example, for a quick evaluation of treatment results for TMJ syndrome.

4 Conclusions and Future Work

In recent years surface processing and analysis have been playing a more and more important role in a wide range of computer graphics applications. This is caused by both the popularization of 3D scanners and 3D printers and by the growing

popularity of medical 3D scanner devices (such as a CT and NMRI scanners). As a result, the demand for different types of analyses of multipoint 3D mesh structures, including surface damage identifications (especially in medical applications), increases.

In this paper the solution based on light reflections used to identify damages of a surface was presented. The proposed solution has one major advantage: it can be easily implemented with the use of typical and cheap computer hardware (e.g. a graphically-accelerated PC workstation, etc.). Vertex Shader-based 3D mesh processing and the proposed mathematically ideal rendering method offer a really fast and reliable surface damage comparison tool. It can be used in a real-time mode, processing complex 3D mesh objects taken directly from a 3D scanner or other similar devices.

On the other hand, it is important to assume that any precise 3D object surface analysis process would require multiple rendering passes—with zoom, angle or exposure shifts between them. In this case a 2D data will be harvested and passed to the external module for further analysis many times. Therefore, it is important to provide a process that will be able to generate results with extremely high speed—processing even hundreds of mesh “frames” per second (and subsequently producing results in real time). Thanks to 3D accelerated graphical platforms with vertex shading technology, it can be done natively. 2D images obtained will probably be compared with patterns in some domain-specific ways. For instance, comparative cross-sections from these images could be extracted. A comparison process itself can also be iterative, where each new image is rendered after a slight angle, zoom or exposure shift. This technique will be similar to a classical 3D rotary laser surface scanning [9], where a laser-head takes a 3D object reading with fixed angular resolution. Factory-ready procedures for such analyses can be developed in further work. For now we can be sure that a quick 3D to 2D rendering method suitable for such jobs is available—with extensive options opened.

References

1. Hedin, F., Klasén, K.: Terrestrial laser scanning: an investigation of 3D CAD model accuracy. Master’s thesis, Royal Institute of Technology (KTH) (2003)
2. Smith, C.: On vertex-vertex systems and their use in geometric and biological modelling. Ph.D. thesis, Calgary, Alta., Canada (2006)
3. Lorensen, W.E., Cline, H.E.: Marching cubes: a high resolution 3D surface construction algorithm. In: Stone, M.C. (ed.) SIGGRAPH. pp. 163–169. ACM (1987). doi:10.1145/37401.37422. <http://dblp.uni-trier.de/db/conf/siggraph/siggraph1987.html#LorensenC87>. <http://www.bibsonomy.org/bibtex/2d653a88c0bcd04018dbf0d67c1aa2e37/dblp>. Light-reaction analysis method for 3D surface damage identification
4. NVidia Vertex Shaders. http://www.nvidia.com/object/feature_vertexshader.html (2015)
5. Nvidia: Cg Toolkit Release Notes. NVIDIA Corporation (May 2012)
6. Crow, F.C.: Shadow algorithms for computer graphics. In: George, J. (ed.) SIGGRAPH, pp. 242–248. ACM (1977). <http://dblp.uni-trier.de/db/conf/siggraph/siggraph1977.html#Crow77>

7. Kaplan, A., Assael, L.: Temporomandibular disorders: diagnosis and treatment. W.B. Saunders (1991). <https://books.google.pl/books?id=VvNpAAAAMAAJ>
8. Tadeusiewicz, R., Ogiela, M.R.: Medical image understanding technology—artificial intelligence and soft-computing for image understanding. In: Studies in Fuzziness and Soft Computing, vol. 156. Springer (2004). doi:10.1007/978-3-540-40997-7
9. ShapeGrabber. <http://www.shapegrabber.com/sol-products-3d-auto-inspection-Ai310.shtml> (2015)