Multiple Material Layer Visualization for Cultural Heritage Artifacts

A. Moutafidou[2], G. Adamopoulos[2], A. Drosou[2], D. Tzovaras[2] and I. Fudos[1]

[1]Department of Computer Science & Engineering,
University of Ioannina, Greece
[2] Information Technologies Institute,
Center for Research & Technology, Greece

Abstract

Material aging has a significant effect on the appearance of cultural heritage objects. These aging effects depend on material composition, object usage and weathering conditions but also on physical and chemical substance parameters. Some types of changes in the materials underneath the visible layers can also be detected and subsequently simulated. Furthermore, recent 3D printing technology enables exporting 3D objects with transparency information. We report on the development of software tools for visualization of material aging for artwork objects that can be used by curators and archaeologists to understand the nature of aging and prevent it with minimal preservation work.

1. Introduction

An artwork piece may consist of different layers of materials, such as a painting with multiple layers of colors, which decay differently and may affect the local morphology of the object. Archaeologists and curators bear the responsibility to study the aging process of each material and then apply that knowledge to restore artifacts to a prior state or to prevent further decay. In this context, it is imperative to have tools that can render multi-layered objects.

Material aging related visual effects are important for capturing realistic appearance in computer generated images. Simulating such phenomena results in images which have a much higher degree of realism [PNdJO14].

Subsequently, tools are needed for (i) the realistic rendering of artwork objects that have undergone simulated aging and (ii) the visualization of all different material layers to prevent further decay.

For (i) we have developed a renderer that utilizes all material model data. This consists of surface geometry, material color, reflectance distribution and other material properties. The material modeling is based on a hybrid methodology that combines material aging models and physical models that represent multiple material types. Accordingly every object is represented by a 3D model, which is composed using 2D and 3D multi-modal sensor data. All these models are then combined to provide a realistic rendering of the object.

To address (ii) we have developed a multiple layer renderer to analyze and visualize all the inner material layers of artwork objects. Finally, we provide a tool for exporting models with transparency for 3D printing multiple layer objects using recent state of the art technology. With this new feature a curator or an archaeologist may also print and examine the surface and the inner layers of an artwork object.

2. Related Work

Material aging depends on material composition, object usage, weathering conditions and a large number of other physical, biological, and chemical parameters. Different aging phenomena often play a key role in realistic rendering. Their absence results to non-realistic surfaces, looking too clean and smooth [MG08, EGKVKD11, Rus09].

While the simulation of fracture physics has been studied in computer graphics [LKA16], reproducing fracture patterns observed in real-world materials remains a difficult problem. In [PNdJO14] a high-poly mesh is dynamically modified to adaptively capture local details wherever it is required by the simulation. Crack patterns observed in materials arise due to small-scale interactions between elastic strain, plastic yielding, and material failure [DH96, DEJ*99]. Stress gradients can be very large near the crack tip where the stress field often approaches singularity. In [YFC04] the surface of wood is modeled by values assigned to tetrahedral mesh vertices.

Photo-realistic rendering techniques are capable of rendering images that attempt to capture the appearance of yet to be manufactured objects [Rus09]. The visualization of the effect of several impact factors on material aging such as physical, chemical, biological, environmental, and weathering requires novel techniques for

DOI: 10.2312/gch.20181353

simulating and then combining these results to create geometrically and visually realistic scenes [Kid12].

Several algorithms ranging from photorealistic rendering, such as global illumination, order-independent transparency to volume visualization, molecular, hair and solid geometry rendering require accurate multifragment processing at interactive speeds [VF13], [VF12]. In this paper, we have used such state of the art techniques to facilitate interactive realtime rendering of objects with multiple material layers.

3. A Visualization Tool for Cultural Heritage Artifacts

The work of curators and archaeologists is based on the observation of existing cultural heritage artifacts in their current state, combined with the knowledge gathered from experience on how different materials age. In this scenario, it is of extreme importance to have tools produce and visualize digital representations and models of visual surface appearance and material properties, to help the scientist understand how they evolve over time and under specific environmental conditions. Our work adapts and combines state of the art approaches from artificial material aging and multi-fragment rendering to develop a tool that helps visualize knowledge from scanners and sensors drawn on cultural heritage objects, using advanced photo-realistic rendering methods.

3.1. A Renderer for Material Aging Visualization

Rendered objects consist of two definite types of data, triangulated mesh which gives us a rough approximation of the geometry both in outer surface and inner layers, and texture maps which describe detailed properties of the surface and the material of each given layer. The Physically Based Rendering (PBR) technique, uses these types of data to realistically render real world objects in a virtual world with as little processing power as possible [LW].

Multiple sensors are used to provide the data required for each model. For the outer surface reconstruction of the models, photogrammetry and laser scanning techniques are used to obtain a middle or high resolution triangulated mesh representing the surface. Material models consist of surface geometry (including normals), material color, reflectance distribution and other light related material properties. Material modeling is based on a hybrid method that combines material aging models and physical models.

The multiple material layer visualization tool is used to visualize, understand and analyze multiple sensor data. Therefore, it is designed to investigate the integration of: (i) low-resolution photogrammetry or laser scanner produced mesh, (ii) high resolution micro-profilometry surface measures, (iii) RTI for capturing surface shape/color and enabling the interactive re-lighting of the object and (iv) x-ray, ultrasound and ultraviolet light for detecting multiple material layers.

Figure 1 (a) and (b) shows the improvement of the rendering result by using only the normal map texture.

3.2. Browsing Multiple Material Layers

We have developed a multi-fragment renderer that uses multiple rendering passes to visualize the details of the volumetric geometry of our models. The vertex and fragment shader programs used in our tool combine modern rendering techniques, like Physically Based Rendering, algorithms that achieve per pixel multifragment processing and our material layer highlighting technique. We give users the ability to observe both outside and inside surfaces of an artifact in an intuitive way. The user can browse the changes made to an object as time evolves, when provided with sufficient data. The aging data are either different versions of the entire object, or local deformations, based on real-world aging measurements or an emulation process.

To process all useful fragments sorted by depth we use three consecutive rendering passes before the final rasterization. The idea is to store the depth value of each fragment in a buffer during a single pass and in a separate pass we sort the fragments per depth and give appropriate values to the alpha parameter of the fragment color.

The common *alpha-blending* technique is not appropriate for our viewer because the different layers cannot be rendered distinctly and the observation of the attributes of each layer is not possible (see for example Figure 3 as compared to Figure 2). In the edges of Figure 3 we can observe front faces of surfaces that are produced by folds and wrinkles and therefore should not be visible. During our process the user gives a preference as to which of the layers she/he wants to be highlighted. By having information about sorted fragments per-depth it is feasible to prioritize a selected fragment as part of a specific layer and render it appropriately.

For each pixel all fragments are stored and sorted per-depth [VF13]. We render only the k nearest front facing fragments that belong to different layers in the correct order. By doing so we can guarantee that there will be no silhouette artifacts as far as the models are solids with: (i) non intersecting boundary surfaces (ii) each boundary surface is a closed watertight non manifold orientable surface. In case that the surfaces have holes our multiple layer renderer still produces no artifacts since we ignore all front facing fragments of a specific layer when we encounter a back facing fragment without a prior front facing fragment.

Our model has k different layers for the user to select from. The preference of the user regards the number of highlighted layers $d, d \leq k$ and a parameter $0 \leq v_\alpha \leq 1$ that specifies how much the non highlighted layers will contribute to the final result. When the value of $v_\alpha = 1$ only the highlighted layers are contributing to the final image. For each pixel the *rgb* color value $col.rgb$ of of the i_{th} fragment f_i is a weighted sum of the colors of the fragments, influenced by the user preference, as follows:

$$col.rgb = \sum_{i=1}^{k} col(f_i).\alpha * col(f_i).rgb, \; where \sum_{i=1}^{k} col(f_i).\alpha = 1$$

and the α value of fragment f_i is determined by the user preferences d and v_α as follows:

$$col(f_i).\alpha = \begin{cases} \frac{1}{d} v_\alpha, & \text{if } f_i \text{ is highlighted} \\ \frac{1}{k-d}(1 - v_\alpha), & \text{if } f_i \text{ is not highlighted} \end{cases}$$

The result is illustrated in Figures 2b and 2c where three differ-

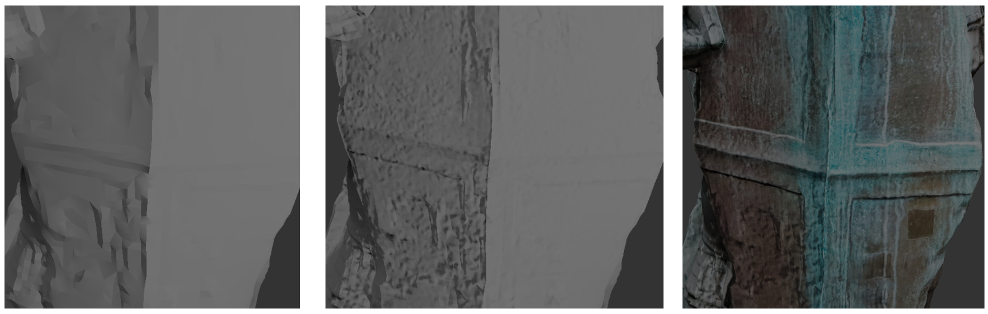

(a) *geometry* **(b)** *geometry and normal map* **(c)** *geometry and all texture maps*

Figure 1: *A detail of rendering original geometry, then enhancing using texture maps.*

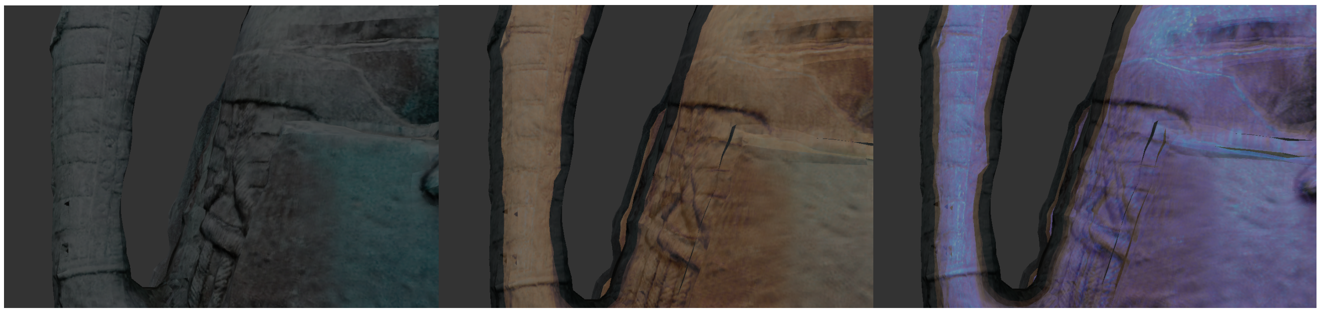

(a) *outer layer* **(b)** *in-between layer* **(c)** *inner layer*

Figure 2: *Rendering multiple material layers.*

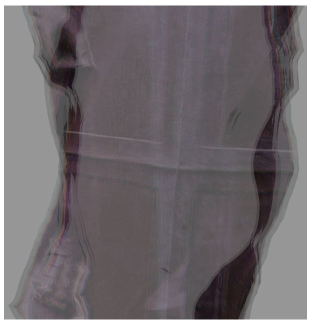


Figure 3: *Using simple alpha-blending is not appropriate for browsing the inner layers.*

ent layers are rendered (different color is used on each layer to help the user distinguish among the visible layers). In this example, the inner layers represent layers of color, coating, and cladding. In this example we render only the surface of each layer.

3.3. 3D Printing with Transparency

The inner layers of any 3D model are not visible when exported in standard format for 3D printing (e.g. stereo lithography stl files). The observation of the inner layers is important for the work of curators and other scientists that are involved in the process of preserving or restoring cultural heritage artifacts. For people that are not acquainted with 3D viewing systems on the computer it is important to be able to fabricate 3D objects with transparency where inner layers may be visible. Recent research in [ABU18] has proposed and developed a technique for 3D printing transparent or partially transparent objects. We have built on this technology to allow the user to select certain areas on the surface of an object that will be printed with transparency.

The concept is that the user selects an area for which she/he wants the 3D printed replica to be transparent and the tool has to modify the values of the diffuse texture in the selected area to $(255,255,255,0)$ for the 3D printer to realize that this area has to be printed with a clear transparent material.

In our interface we determine a volume using a selection box which will be made transparent when printed. Since we are going to produce an stl file for printing we convert all polyhedral surfaces to triangles. The faces (triangles) of the model that are inside this box will have transparent values on the corresponding textures during export.

We perform clipping of all triangles that lie inside the selection box. For efficiency, we use the 3D version of the Hodgeman-Sutherland polygon clipping algorithm. Furthermore, we use two criteria for excluding triangles similar to the Cohen Sutherland algorithm region tests for line segments. Triangles with all vertices on the same side of a plane defined by the faces of the box lie completely outside the box and there is no need to treat them. Triangles with all vertices inside the box lie completely within the box.

After running this algorithm we get convex polygons that lie completely within the box. We convert the polygons to triangles. Finally, we convert the texture coordinates values $TexCoords(x,y)$ of these vertices to pixel values $p(x,y)$ referring to the diffuse texture of the model, using the width and height $size(w,h)$ of the texture, $p(x,y) = TexCoords(x,y) * size(w,h)$.

This task can also be implemented on the GPU pipeline by determining all vertices that lie inside the boxes and then reproducing the texture from the color of the vertices. However, this may result in deterioration of the quality of the diffuse texture.

We edit the value of all pixels of the triangle of the texture to $(255,255,255,0)$.

The outer layer of the object has to differ from the inner ones in such a way that is apparent that the object has multiple layers. This is why we make the selection box smaller for each inner layer of the model. As a result we create a custom window to browse the inner layers on a 3D printed transparent-ready object (see Figure 4).

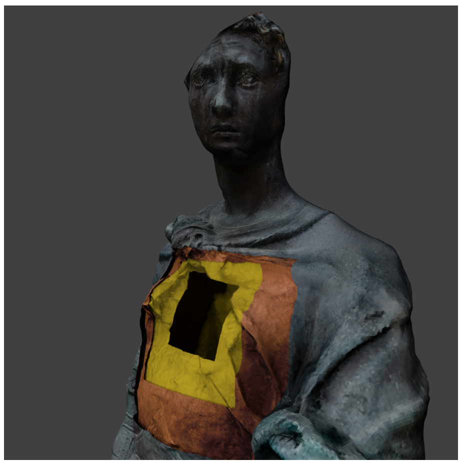


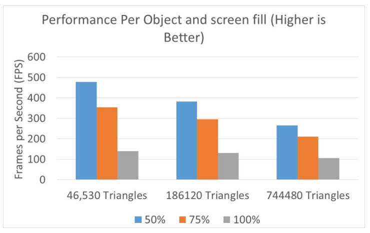
Figure 4: *3D printer ready model with multiple material layers and transparency.*

4. Experimental Evaluation

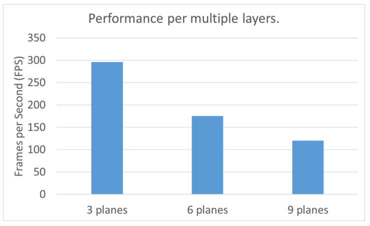
In this section we present quantitative and qualitative experimental evaluation of our visualization tool.

We have conducted experiments to evaluate the performance of the visualization tool during the rendering process. We measure FPS (frames per second) in several cases and we compare them with each other to understand what is the overhead for each part, as well as with the basic $Alpha Blending$ technique which is implemented in OpenGL.

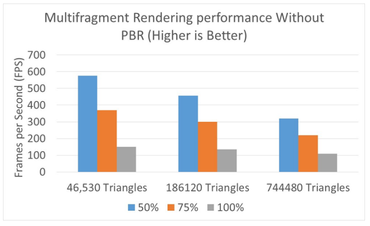
Our method is affected by many parameters such as number of triangles of the mesh, the screen resolution and the percentage of the screen that contains fragments. Because our method is fragment based the number of fragments per pixel affects the performance more than the triangle count of a mesh. Figure 5a) illustrates the FPS during rendering of three different objects for the same screen resolution (1024x1024) and for different screen fragment population percentages(50% 75% 100%).



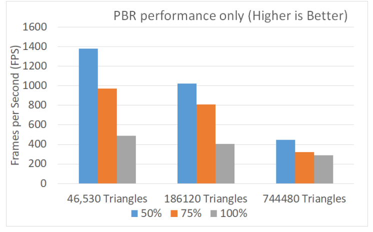
(a) *Rendering performance per object and screen population.*



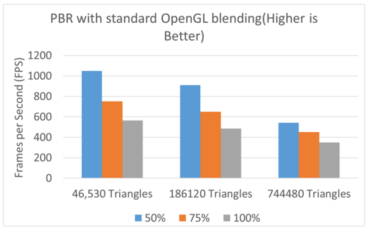
(b) *Performance for multiple layer visualization.*



(c) *Multiple layer performance without PBR.*



(d) *PBR performance without multiple layers.*



(e) *Simple alpha-blending performance with PBR.*

Figure 5: *Performance results*

The objects we used are comprised of three layers and they have the same geometry (three material plates) but represented at different level of detail (LOD). The low polygon mesh has 46.530 triangles, the mid polygon mesh has 186.120 triangles and the high polygon mesh consists of 744.480 triangles. We can see that the number of triangles, despite been largely varied does not cause significant drop in FPS. Even if we double or triple the number of layers we observe a linear drop in FPS (Figure 5b). Screen population with fragments has a significant impact on performance. This is expected since we need to perform more computations to store and process all fragment per pixel. We have evaluated the performance of each rendering method separately to understand the influence it has on the overall performance of the tool. As expected the multi-fragment rendering algorithm entails a large overhead because of the buffers used and the processing and sorting of all fragments. Figure5c shows that the overhead of *PBR* is almost negligible when we perform multi-fragment rendering. In Figure 5d we see that for large number of triangles we have low performance rates, while for small number of triangles we see high performance values. Taking into respect that the low polycount can portray almost the same level of detail as the high polycount with the *PBR* pipeline we realize why *PBR* is used widely in the games and movies industry. Finally, Figure 5e shows the increased performance of simple alpha-blending as compared to multifragment rendering which however is not appropriate for our purposes (see Figure 3).

Figure 1 illustrates the visual results of the *PBR* pipeline. Figure 1a depicts the geometry of the model without any additional information. In 1b we can observe the detail on the surface of the model such as the corrosion, which is captured by photogrammetry in the normal map. In Figure 1c the final result of the *PBR* method is shown with all surface material sensor data taken into account.

Finally in Figure 2a we can see the outer layer of the object zoomed in the lower part of the right arm and in Figure 2b we see the layer that lies in the middle, while in the last Figure 2c we can observe all three layers of the model in real-time. The inner layers of the model are colored differently for clarity purposes. However we can see that the results are rendered without errors from foldings, they are straightforward and the rendering is performed in real-time.

5. Conclusions

We have presented a method for visualizing the morphology of scanned 3D objects using multiple sensor data. Furthermore, we have introduced a multiple material layer renderer based on several sets of sensor data and state of the art multifragment rendering utilizing modern GPUs. Finally, we have provided an interactive utility for the user to create transparent windows inside the object before exporting it for translucency enabled 3D printing.

6. Acknowledgment

This work has been partially supported by the European Commission through project Scan4Reco funded by the European Union H2020 Programme under Grant Agreement No. 665091. The opinions expressed in this paper are those of the authors and do not necessarily reflect the views of the European Commission.

References

[ABU18] Alan Brunton Can Ates Arikan T. M. T., Urban P.: 3d printing spatially varying color and translucency. *Siggraph 2018, Vancouver, to appear* (2018). 3

[DEJ*99] Dorsey J., Edelman A., Jensen H. W., Legakis J., Pedersen H. K.: Modeling and rendering of weathered stone. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1999), SIGGRAPH '99, ACM Press/Addison-Wesley Publishing Co., pp. 225–234. 1

[DH96] Dorsey J., Hanrahan P.: Modeling and rendering of metallic patinas. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1996), SIGGRAPH '96, ACM, pp. 387–396. 1

[EGKVKD11] El-Gaoudy H., Kourkoumelis N., Varella E., Kovala-Demertzi D.: The effect of thermal aging and color pigments on the Egyptian linen properties evaluated by physicochemical methods. 1

[Kid12] Kider Jr. J. T.: *Simulation of Three-dimensional Model, Shape, and Appearance Aging by Physical, Chemical, Biological, Environmental, and Weathering Effects*. PhD thesis, Philadelphia, PA, USA, 2012. AAI3542821. 2

[LKA16] Lee S., Kim J.-W., Ahn E.: A visual simulation method for weathering progress of stone artifacts. *Multimedia Tools Appl. 75*, 23 (Dec. 2016), 15247–15259. 1

[LW] Lafortune E. P., Willems Y. D.: A theoretical framework for physically based rendering. *Computer Graphics Forum 13*, 2, 97–107. 2

[MG08] Mérillou S., Ghazanfarpour D.: Technical section: A survey of aging and weathering phenomena in computer graphics. *Comput. Graph. 32*, 2 (Apr. 2008), 159–174. 1

[PNdJO14] Pfaff T., Narain R., de Joya J. M., O'Brien J. F.: Adaptive tearing and cracking of thin sheets. *ACM Trans. Graph. 33*, 4 (July 2014), 110:1–110:9. 1

[Rus09] Rushmeier H.: Computer graphics techniques for capturing and rendering the appearance of aging materials, 01 2009. 1

[VF12] Vasilakis A., Fudos I.: S-buffer: Sparsity-aware multi-fragment rendering. In *Eurographics (Short Papers)* (2012), pp. 101–104. 2

[VF13] Vasilakis A.-A., Fudos I.: Depth-fighting aware methods for multifragment rendering. *IEEE Trans. Vis. Comput. Graph. 19*, 6 (2013), 967–977. 2

[YFC04] Yin X., Fujimoto T., Chiba N.: Cg representation of wood aging with distortion, cracking and erosion. 216–223. 1