การสร้างภาพเคลื่อนไหว (GIF) จากวัตถุ 3 มิติ ด้วยซอฟต์แวร์โดยอัตโนมัติ

Towards Generating GIFs from 3D Model: An Automated Software Framework

Samart Moodleah[1], Pornsuree Jamsri[2] and Supannada Chotiphan[3]

Faculty of Information Technology, King Mongkut's Institute of Technology Ladkrabang, Thailand[1, 2, 3]

E-mail: samart@it.kmitl.ac.th[1], pornsuree@it.kmitl.ac.th[2], supannada@it.kmitl.ac.th[3]

บทคัดย่อ

งานวิจัยนี้นำเสนอซอฟต์แวร์ในการสร้างภาพเคลื่อนไหว (GIF) อัตโนมัติจากวัตถุ 3 มิติ โดยการรวมภาพที่มีความแตกต่างที่ต่อเนื่องกันเป็นลำดับเวลา ซึ่งภาพต่าง ๆ เหล่านี้ถูกสร้างขึ้นโดยอาศัยหลัก “การสร้างภาพแบบสุดท้าย” (Render) ที่ใช้ข้อมูลจากวัตถุ 3 มิติต้นแบบ คือ Vertex, Face และ Normal โดยที่คุณภาพของภาพเคลื่อนไหวจะขึ้นอยู่กับหลายองค์ประกอบ เช่น จำนวนของภาพต่อเนื่องที่จะเป็นตัวกำหนดคุณภาพโดยตรงของความต่อเนื่องในการเคลื่อนไหวของภาพ เป็นต้น

Abstract

We present an automated software framework creating a GIF animation image of 3D model using a sequence of rendering images of its 3D contents: vertices, faces, colors, and normal. The quality of an animated GIF depends on many factors, e.g. a number of images that reflect the object’ smooth transition.

1. Introduction

3D CAD (computer-aided design) models are used in many areas; 3D printing industry uses them as the objects that later will be transformed into a collection of machine coordinates, x-y positions, which tell the printer’s nozzle where to move. Video games industry uses them as the assets, the architecture sector uses them to represent the real physical object in their design, and the medical industry uses them to represent human organs. A 3D model represents a physical body utilizing a collection of points in 3D dimensions, associated by different geometric elements such as line, triangle, polygon, etc. The generating of a 3D model can be done with several approaches, for example, by hand, mathematical modeling, scanning, reconstruction technique, etc.

Thousands of models are available online to be used free or paid and at an entertaining or professional level. Online communities provide these models in several formats (e.g. 3DS, BLEN, DAE, OBJ, STL, and PLY) that can be used in almost all available application platforms. The popular 3D repository websites such as Thingiverse [1], GrabCAD [2], or Sketchfab [3], provide a large collection of 3D models and display them as static images on their web page for users to preview before taking any further action. This analogy allows users to visually see the overview of an interested object: color, size, texture, shape, etc. However, in many cases, an image alone is just not enough to decide whether it is the right choice for what users are looking for.

Repository sites such as Thingiverse provide static thumbnail images that allow users to see several 3D models. Therefore, when an interested image is selected, a detailed page will be loaded and downloadable files prepared in several formats that can be open by any choice of application on a user's computer.

On the other hand, a Sketchfab [3-5] site provides a strategy to explore its contents using two steps: static image thumbnail and WebGL. First, users explore a site contents through thumbnail image, similar to a previous site strategy, and click any image there that interests which take them to the later step that demonstrates a 3D content using WebGL technology which allows users to see an object in greater detail: rotation, zooms, etc.

To display multiple 3D models on the internet browsers, web browsers use WebGL, which might cause some difficulties for users in many possible ways: high internet traffic, high client-browser computational power, etc. Moreover, downloading every interesting 3D model to verify on a local computer may not be convenient or practical, as this requires large bandwidth and storage space. Therefore, many online websites, see [6, 7] for examples, compromise on displaying 3D models on a web browser by using a GIF animation format that enable animation visualization with low complexity as a static image. Facebook and other social media software have now incorporated plugins that can show 3D models that require qITF tool to develop such rendering

In this paper, we present an automated software framework to create a 3D model GIF animation image using a sequence of rendering images of its 3D contents: vertices, faces, colors and normal. The framework make use of reliable opensource libraries such as vcglib [8], imagemagick [9], OpenCV [10], and OpenGL [11].

2. Literature Review

Presenting 3D content online through a website has been considered a lack of speed approach compared to other media such as text, image, video or sound [5].

The smallest element of a 3D Mesh topology is represented by three connected vertices, the plane containing the three vertices is known as the face, and a vector normal to the

```
ply
format ascii 1.0           { ascii/binary, format version number }
comment made by anonymous  { comments are keyword specified }
comment this file is a cube
element vertex 8           { define "vertex" element, 8 in file }
property float32 x         { vertex contains float "x" coordinate }
property float32 y         { y coordinate is also a vertex property }
property float32 z         { z coordinate, too }
element face 6             { there are 6 "face" elements in the file }
property list uint8 int32 vertex_index
                           { "vertex_indices" is a list of ints }
end_header                 { delimits the end of the header }
0 0 0                      { start of vertex list }
0 0 1
0 1 1
0 1 0
1 0 0
1 0 1
1 1 1
1 1 0
4 0 1 2 3                  { start of face list }
4 7 6 5 4
4 0 4 5 1
4 1 5 6 2
4 2 6 7 3
4 3 7 4 0
```

Figure 1. A PLY files format

plane represents the direction of the face surface. The early format that represent 3D data for web is VRML (Virtual Reality Modeling Language) and its subset X3D (where the X stands for Extensibility). The 3D data will be visualized by a specific visualization tool which might require users to install a plug-in in order to display the content properly. This is a considerable major drawback of this approach because not only does it rely on user technical knowledge, but also it requires a specific visualization tool for different web browsers. However, this disadvantage has been overcome by the introduction of WebGL standard in 2009.

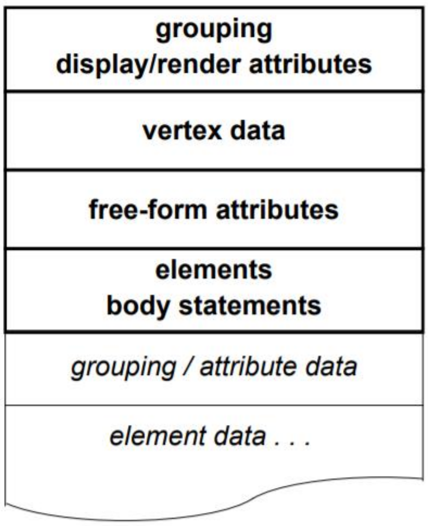


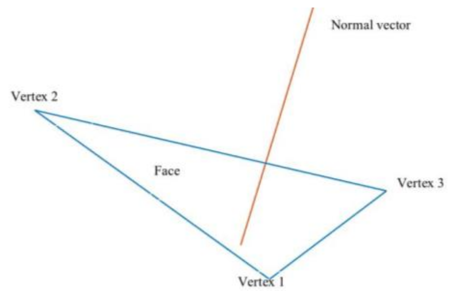
Figure 2. A structure of an Obj file format

WebGL fundamentally changes a strategy to display 3D content online by enabling web browsers to natively render 3D data using the features of 3D graphic hardware of end-user devices without extra plug-in installation. One advantage of WebGL is that it is available for multiple platforms as long as they support modern web browsers: Windows, MacOS, iOS, Android [12]. However, WebGL itself could create some difficulty in making an application because it requires deep understanding about graphic programming.

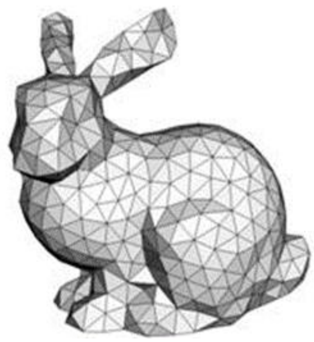
3D content can be stored in enormous formats which depend on what type of area that they will be used. For example, STL format has been standard for an industry that is used in prototyping areas and also popular in additive manufacturing areas, e.g. 3D printer [13]. The popularity of STL file format is its simplicity because it majorly contains only mesh information: vertices, faces, normal. Unfortunately, the major disadvantage is also because of its simplicity; there is no color or texture information. More recent formats such as STEP, PLY, OBJ, DXS, 3DS, and SLC have both pros and cons that in some way are gaining popularity and becoming a more important format in several areas of use. The greater details of those described formats can be seen in [13, 14].

In this paper, the Polygon File Format or the Stanford Triangle Format known as PLY and the OBJ, i.e. geometry definition file format, is used solely to demonstrate how a proposed framework operates.

A PLY/OBJ format was originally designed to store 3D data from a 3D scanner. The file format is simply designed to keep an object in the form of flat polygons collection. It also is capable of storing multiple object properties such as color, surface normal, texture coordinates, etc. This format can be stored in two versions: ASCII and Binary.



(a)



(b)

Figure 3. (a) Mesh topology and (b) mesh model

An ASCII version of PLY format and OBJ file structure are presented in Figure 1 and Figure 2 [15], respectively. Figure 3 shows mesh topology and mesh example. It consists of two major parts: header and element. The first part presents general information of an object such as number of vertices, number of faces, and other properties. Another part displays all elements, i.e. vertices coordinates, faces (vertex-indices), vertices color, texture coordinates, and other user defined properties. OBJ format is another widely used format to store 3D objects first developed by Wavefront Technologies for its advanced visualizer animation package, details of the format is on the Paulbourke website [16].

Graphic Interchange Format (GIF) is an image file format and it is considered as one of the most popular formats to represent images on the web, One important feature is the ability to encode simple animations by storing a number of images in a single file for later sequential display [17]. GIF is essentially an indexed image file format that stores multiple, slightly different, images with a certain time-delay. Therefore, it is suitable for displaying a simple animation or video clip.

Managing 3D data through a website is not an easy task. This is so, for example, because of the data's size of multiple objects to be rendered on the front-page or the computational power that end-user's device requires for rendering 3D content. To propose a reasonable strategy between static images and WebGL 3D rendering contents, GIF animation would be a proper and efficient way to display 3D animation-like image format that allow users to see a multi-view of 3D content while containing the complexity similar to displaying static images.

3. Methodology

Our novel approach lies in proposing the software framework that automatically generates GIF animation image from a 3D model, converting 3D space to 2D space. GIF animation image is composed of several images that slightly change in time so that when they are combined together they result in a motion-like animation.

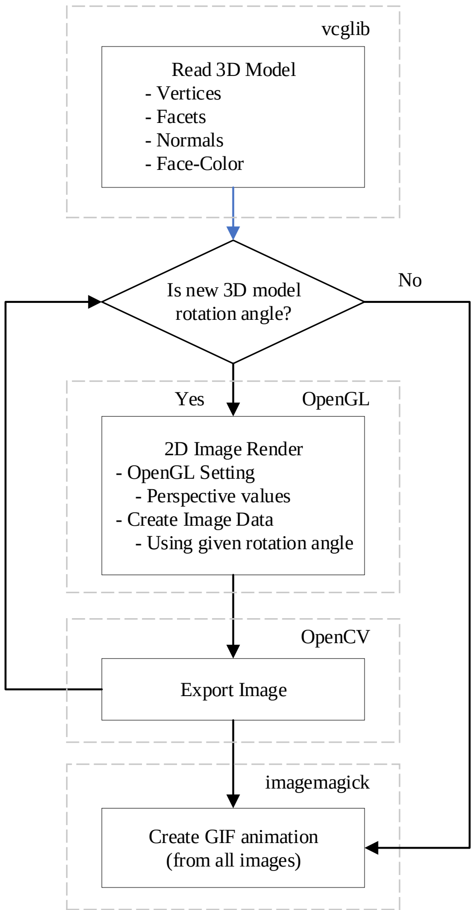


Figure 4. A proposed framework

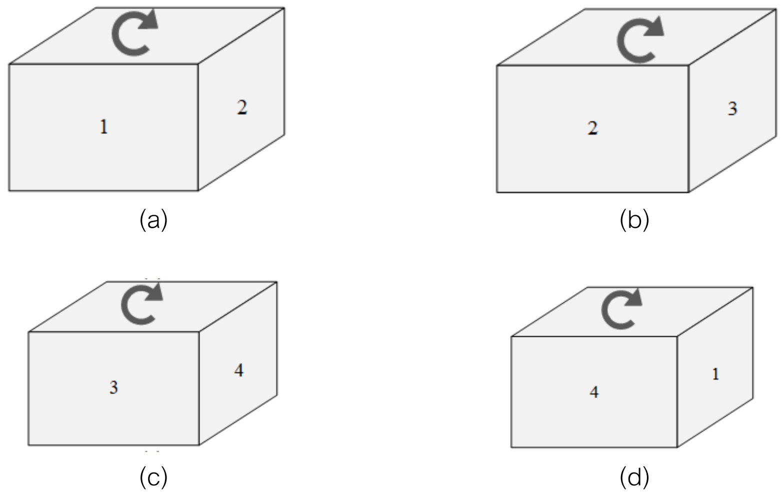


Figure 5. A demonstration of each face of an object rotation at different angles

A framework diagram is illustrated in Figure 4. It comprises four steps: read 3D model, image rendering, exporting image, and creating GIF animation. The details of each part and opensource to be used will be explained in this section.

The first step requires that the 3D model is read and kept in the vcglib data structure format. The result can offer two scenarios: with and without texture. In the first scenario, a texture mapping file will be stored in an extra image file that will be read altogether at the same time. Then, texture mapping will be calculated. The later scenario, 3D model, provides only vertices color which will be later approximated and distributed for all faces. Finally, the output of this step, either the first or second scenario, will be vertices, faces, and colors. The normal of each facet will be calculating using vcglib library.

Before the 3D data will be rendered in the next step, the angle of the model to be rotated is required to make an animation be defined, for example 20 degrees which results totally in 18 images to make it fully rotate 360 degrees.

In the second step, a collection of 3D data will be used to render an image using OpenGL library. Its processes begin with to determine an approximate size of an object (3D model), setting perspective values in order to project 3D data onto 2D space. Finally, a collection of 3D data will be one-to-one mapped onto 2D space.

In the next step, a 2D projection data will be export and save in image format (e.g. jpeg) using OpenCV library. This process will be repeated until a 3D model is fully rotated. At the end of this step, multiple images (with different angles) will be saved, see an example in Figure 5.

At the last step, all images will be combined together for one GIF animation image, automatically.

4. Result and Discussion

Our implementation comprises open-source libraries written in C or C++. We tested our automated software framework on Linux operating system (Ubuntu) computer with Intel Core i7-5500 2.40 GHz and 8 GB of RAM. For testing we choose seven Textured 3D Models generated through 2D images using OpenMVG [18] and OpenMVS [19] (3D reconstruction strategy).

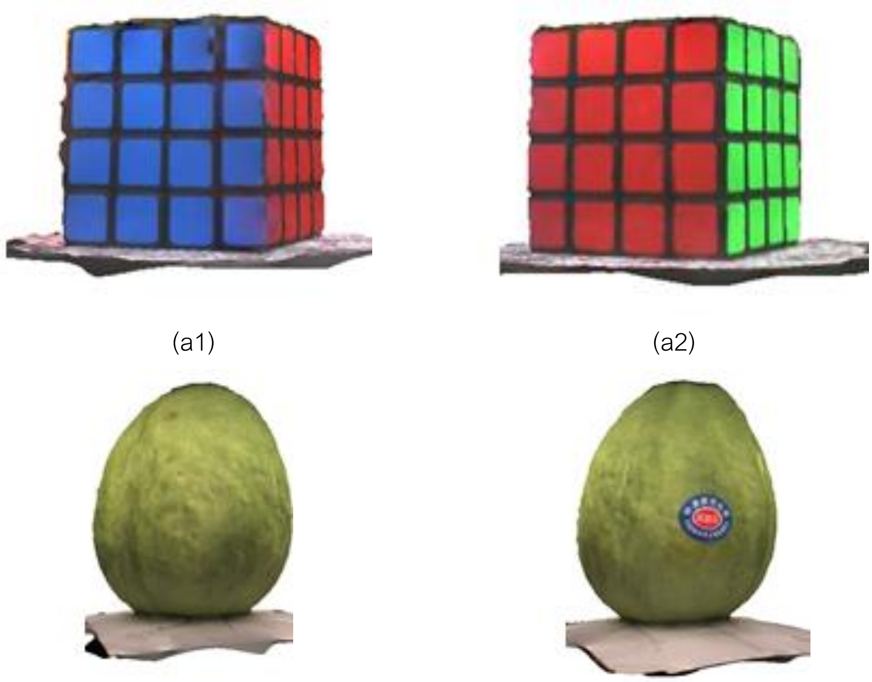
Depending on the resolution of images, the size of the 3D Model varies. An object will be rendered based on its vertices and faces information, then rotated around Y-axis (can be any axis among X-, Y-, or Z-axis), with a certain degree of each rotation step until it reaches 360 degrees. At the individual rotation step, an object image will be saved as a PNG file which later will be combined as a final animation GIF file. There are two criteria that we use in this performing test: an image size is 500x500 pixels and the variation of rotation degrees are 5, 15 and 45 degrees, On the other hand, 72, 24, and 8 images will be generated for final GIF animation. The examples of two quadrant images of an object are illustrated in Figure 6 and two frames generated from each GIF is show in Figure 7.

The quality of an animated GIF depends on many factors, e.g. a number of images that reflects the object smooth transition. Users are able to trade between quality and size of the output.

(a)

(b)

Figure 6. An example of images on different rotation degree



(a1) (a2)

(b1) (b2)

Figure 7. The examples of output images to make GIF: (a) Cube and (b) Pear

5. Conclusion

In this paper we proposed an automated framework to generate an animated GIF to view a 3D Model. We tested the algorithm on seven textured 3D Objects Models with different angles for each view and different file size. We found that our proposed framework can successfully generate 2D GIF file for viewing a 3D Model without using any specific geometric software.

References

[1] https://www.thingiverse.com.

[2] https://grabcad.com.

[3] https://sketchfab.com.

[4] S. Villagrasa, D. Fonseca, E. Redondo, and J. Duran, "Teaching Case of Gamification and Visual Technologies for Education," J. Cases Inf. Technol., vol. 16, no. 4, pp. 38-57, 2014.

[5] R. Scopigno, M. Callieri, M. Dellepiane, F. Ponchio, and M. Potenziani, "Delivering and using 3D models on the web: are we ready?," in Virtual Archaeology Review, 2017, vol. 8, no. 17, pp. 1-9: Universitat Politècnica de València.

[6] https://sirv.com/demos/360-spin-3d-model/.

[7] https://www.webrotate360.com/products/webrotate-360-product-viewer/create-animated-gif.aspx.

[8] http://vcg.isti.cnr.it/vcglib/.

[9] https://www.imagemagick.org.

[10] https://opencv.org/.

[11] https://www.opengl.org/.
[12] P. Cozzi and C. Riccio, OpenGL insights. AK Peters/CRC Press, 2012.
[13] J. D. Hiller and H. Lipson, "STL 2.0: a proposal for a universal multi-material additive manufacturing file format," in Proceedings of the Solid Freeform Fabrication Symposium, 2009, no. 1, pp. 266-278: Citeseer.
[14] K. McHenry and P. Bajcsy, "An overview of 3d data content, file formats and viewers," National Center for Supercomputing Applications, vol. 1205, p. 22, 2008.
[15] "Obj structure," https://www.cs.utah.edu/ ~boulos/ cs3505/obj_spec.pdf.
[16] Paulbourke, "OBJ data format."
[17] W. Burger, M. J. Burge, M. J. Burge, and M. J. Burge, Principles of digital image processing. Springer, 2009.
[18] https://github.com/openMVG/openMVG.
[19] https://github.com/cdcseacave/ openMVS.