FAKULTÄT
FÜR !NFORMATIK
Faculty of Informatics

Splitting of Meshes in Image-Space

BACHELORARBEIT

zur Erlangung des akademischen Grades

Bachelor of Science

im Rahmen des Studiums

Medieninformatik und Visual Computing

eingereicht von

Barbara Schwankl
Matrikelnummer 0852176

an der Fakultät für Informatik
der Technischen Universität Wien

Betreuung: Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Eduard Gröller
Mitwirkung: Projektass.(FWF) Ing. Peter Mindek, MSc

Wien, 05.04.2013

(Unterschrift Verfasserin) (Unterschrift Betreuer)

FAKULTÄT
FÜR !NFORMATIK
Faculty of Informatics

Splitting of Meshes in Image-Space

BACHELOR'S THESIS

submitted in partial fulfillment of the requirements for the degree of

Bachelor of Science

in

Media Informatics and Visual Computing

by

Barbara Schwankl
Registration Number 0852176

to the Faculty of Informatics
at the Vienna University of Technology

Advisor: Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Eduard Gröller
Assistance: Projektass.(FWF) Ing. Peter Mindek, MSc

Vienna, 05.04.2013

(Signature of Author) (Signature of Advisor)

Erklärung zur Verfassung der Arbeit

Barbara Schwankl
Unionstr. 133, 4020 Linz

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

(Ort, Datum) (Unterschrift Verfasserin)

Inhaltsverzeichnis

KAPITEL 1

Introduction

The main purpose of 3D computergraphics is to represent image data. To make these data examinable, user interaction becomes an important part. Hence, the user should be able to translate, rotate and scale the 3D object. With the complexity of the object, the desire to examine just parts of it rises. The examination of the objects' inner structures can also be of interest. That leads to the first problem that needs to be solved: How can we reveal the inner sturctures? By simply removing the outer structures, the context would also get lost that could be important for scientific findings. Hence, the idea is to reveal the point of interest whilst keeping the context. Therefor, several scientific approaches exist. One idea is to simply raise the opacity of the outer structures. For example to reveal a brain, the approach would be to simply raise the opacity of the scull and the skin respectively for making the parts of secondary interest semi-transparent. Another approach is to cut the object so that no information would get lost and still the inside would be fully visible. So called explosion views unfold the inside by breaking the outer struktures into multiple parts and shifting those apart whilst keeping the point of interest in the center.

This paper will examine several approaches to reveal inner structures of complex 3D models retaining the context to be able to examine the object regarding its surroundings.
TOEDIT In section BLA you will learn how to BLA and BLA BLA

KAPITEL 2

Analysis of existing approaches

KAPITEL 3

Methodology

3.1 Volumeshop

3.2 Creating a plug-in with Volumeshop

3.3 Concepts and languages

KAPITEL 4

Suggested implementation

In this section a step by step approach to reveal inner structures of a mesh will be discussed. First a plane needs to be defined that represents the position of the cut. To retain interactivity, several parameters in the Volumeshop interface can be set to translate, rotate and scale the plane. The color and opacity of the plane should also be adaptable to not conceal parts of the mesh. For the splitting we define an offset that indicates how far the two halfs of the mesh shall be apart from the plane. The larger the gap the better can be gained an insight into the model. The splitting itself is no real translation of the two halfs, but the model is rendered twice at different positions in space parallel to the plane. The final step is shading of the models' surface as well as the back facing vertices. The latter will be shaded with another color, but with the same amount of different lights to create the illusion of depth.

4.1 Definition of the plane

A plane in this context is a square of infinite size. It is used as a visual help to define where the mesh will be cut.
To be able to intervene with the Volumeshop, it is necessary to provide properties to set the translation, rotation, and scale vectors as well as the color. Each property requires a name and a type.
This would be a property named „Plane Translation Vector“ of the type Vector (in this case vec3):

```
GetPlugin().GetProperty("Plane Translation Vector").require(
   Variant::TypeVector(Vector(0.0f, 0.0f, 0.0f)));
```

To apply changes made in the interface immediately, an observer for the property has to be added:

```
GetPlugin().GetProperty("Plane Translation Vector").addObserver
   (&m_modVariantObserver);
```

To draw the plane, the Viewing Transformation Matrix has to be loaded. This matrix is for tranforming the coordinates from world space into viewing space.[BOOK: Hearn, Baker]
Then the plane is being adjusted by its affine transformations. All handed parameters are of the type 'float' and taken from the user input.
The commands in OpenGL:

```
glTranslatef(vecPlaneTranslation.GetX(), vecPlaneTranslation.
   GetY(), vecPlaneTranslation.GetZ());
glRotatef(vecPlaneRotationAngle, vecPlaneRotationVector.GetX(),
    vecPlaneRotationVector.GetY(), vecPlaneRotationVector.GetZ
   ());
glScalef(vecPlaneScaling.GetX(), vecPlaneScaling.GetY(),
   vecPlaneScaling.GetZ()); //'z' acts as a dummy as the plane
   is in 2D
```

The color of the plane has to be handed over normalized from the interface to the renderer. In Volumeshop, this can be easily archieved by calling the function 'GetNormalized<Color>'.

```
glColor4f(vecPlaneColor.GetNormalizedRed(), vecPlaneColor.
   GetNormalizedGreen(), vecPlaneColor.GetNormalizedBlue(),
   vecPlaneColor.GetNormalizedAlpha());
```

OpenGL supports several basic graphics primitives by default. For a plane a quad is needed that looks like the following [BOOK hill, p70]:

```
glBegin(GL_QUADS);
      glNormal3f(0, 0, 1);
      glVertex3f(-1, -1, 0);
      glVertex3f( 1, -1, 0);
      glVertex3f( 1, 1, 0);
      glVertex3f(-1, 1, 0);
glEnd();
```

As stated above our plane will be indefinite, but for the purpose as a visual helper it is displayed from -1 to 1 as a default. Note that it is also scaleable for bigger meshes.

The result should look like [FIGURE].

4.2 Splitting the mesh

The splitting offset is again implemented as a interface parameter. If the parameter is set to '0.0f' the mesh does not seem to be split. The larger the value of the offset the bigger the distance between the two parts of the mesh.

```
GetPlugin().GetProperty("Offset").require(Variant::TypeFloat
   (0.5f));
```

Note that of course an observer needs to be added as well.
The normal of the plane has be defined with '(0.0, 0.0, 1.0)', but regarding that the normal vector is still located in the object space of the plane, it needs to be transformed into the same space as the mesh, which would be the modelview space. This is being done by rotating the normal by the same amount as the plane is rotated in the interface.
The following code reveals the rotation matrix regarding the interface inputs.

```
glLoadIdentity();
glRotatef(vecPlaneRotationAngle, vecPlaneRotationVector.GetX(),
    vecPlaneRotationVector.GetY(), vecPlaneRotationVector.GetZ
   ());
```

Of course, the rotation matix could also be calculated using the generally known formulas[BOOK hill p 217]:

x-roll:

$$R\_x(\beta) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\beta) & -\sin(\beta) & 0 \\ 0 & \sin(\beta) & \cos(\beta) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{4.1}$$

y-roll:

$$R_y(\beta) = \begin{pmatrix} \cos(\beta) & 0 & \sin(\beta) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\beta) & 0 & \cos(\beta) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{4.2}$$

z-roll:

$$R_z(\beta) = \begin{pmatrix} \cos(\beta) & -\sin(\beta) & 0 & 0 \\ \sin(\beta) & \cos(\beta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{4.3}$$

However, it should be considered that 3D rotation matrices do not commute [BOOK hill p220], so the order of multiplication matters.

4.3 Shading the mesh

KAPITEL 5

Critical reflection

KAPITEL 6

Summary

KAPITEL 7

Bibliography

7.1 Literature Search

Information on online libraries and literature search, e.g., interesting magazines, journals, conferences, and organizations may be found at `http://www.big.tuwien.ac.at/teaching/info.html`.

7.2 BibTeX

BibTeX should be used for referencing.

The LaTeXsource document of this pdf document provides you with different samples for references to journals [3], conference papers [6], books [2], book chapters [7], electronic standards [5], dissertations [8], masters' theses [4], and web sites [1]. The respective BibTeX entries may be found in the file `references.bib`. For administration of the BibTeX references we recommend `http://www.citeulike.org` or JabRef for offline administration, respectively.

Literaturverzeichnis

[1] Business Informatics Group. http://www.big.tuwien.ac.at. Accessed: 2010-11-09.

[2] M. Hitz, G. Kappel, E. Kapsammer, and W. Retschitzegger. *UML @ Work, Objektorientierte Modellierung mit UML 2*. dpunkt.verlag, 3. edition, 2005 (in German).

[3] Christian Huemer, Philipp Liegl, Rainer Schuster, and Marco Zapletal. B2B Services: Worksheet-Driven Development of Modeling Artifacts and Code. *Computer Journal*, 52(2):28–67, 2009.

[4] P. Langer. Konflikterkennung in der Modellversionierung. Master's thesis, Vienna University of Technology, 2009.

[5] OASIS. *Business Process Execution Language 2.0 (WS-BPEL 2.0)*, 2007.

[6] A. Schauerhuber, M. Wimmer, W. Schwinger, E. Kapsammer, and W. Retschitzegger. Aspect-Oriented Modeling of Ubiquitous Web Applications: The aspectWebML Approach. In *Proceedings of the 14th Annual IEEE International Conference and Workshops on the Engineering of Computer-Based Systems (ECBS '07), March 26-29, Tucson, Arizona, USA*, pages 569–576. IEEE CS Press, 2007.

[7] W. Schwinger and N. Koch. Modeling Web Applications. In G. Kappel, B. Pröll, S. Reich, and W. Retschitzegger, editors, *Web Engineering*, pages 39–64. John Wiley & Sons, Ltd, 2006.

[8] M. Wimmer. *From Mining to Mapping and Roundtrip Transformations - A Systematic Approach to Model-based Tool Integration*. PhD thesis, Vienna University of Technology, 2008.