

Evaluation of an Appearance- Preserving Mesh Simplifica- tion Scheme for Configura AB

Rasmus Hedin <rashe877@student.liu.se>
Erik S. V. Jansson <erija578@student.liu.se>

Contents

Contents

1	Introduction	1
1.1	Motivation	1
1.2	Aim	2
1.3	Research Questions	2
1.4	Delimitations	2
1.5	Background	2
2	Theory	3
2.1	Mesh Simplification	3
2.2	Metrics for Appearance Preservation	3
2.3	Measuring Algorithm Performance	3
	Bibliography	4



1 Introduction

For many years the field of computer graphics has been an important part in many industries, but especially in the entertainment industry (for instance video games and motion pictures). These industries generate a lot of money, and are quickly growing in size. A recent survey by Kroon and Nilsson [3] from *Dataspelsbranschen* have shown that the video games industry in Sweden generated €1325 M in revenue in 2016, a steep increase from the €392 M in 2012. Also, most movies nowadays use to some extent 3-D computer graphics in scenes where the cost would be too large to reproduce in reality, be too risky for actors, or simply be impossible.

The rendering of meshes (a collection of polygons describing a surface) is one of the main activities in computer graphics (usually, a collection of meshes; a so called scene description). In many cases, these meshes are very detailed, and require a large amount of polygons to fully describe a surface. This is problematic, since the rendering time of a scene depends on the number of polygons it has. Therefore, it's important to reduce the number of polygons in a mesh as much as possible. This is especially true in video games, where the scene needs to be rendered in real-time. However, if the number of polygons are reduced too much, it will degrade the visual quality of a mesh, giving a progressively flatter surface than intended and removing small surface details. This destroys the intended *geometrical appearance* of the mesh.

1.1 Motivation

While the geometrical appearance of a mesh is important, it's not the only factor which gives the final appearance of a mesh when rendering. According to Cohen *et al.* [1], both the surface curvature and color are equally as important contributors. We'll use *textured appearance* as the common name for these since surface properties are usually specified by using a texture map.

In computer graphics, the process to reduce the number of polygons in a mesh based on some metric is called a *mesh simplification algorithm*, as seen in Talton's survey [4] on the subject. Historically, these have been mostly concerned with minimizing the geometrical deviation of a mesh when applying it. Somewhat recently, methods for minimizing the texture deviation when simplifying a mesh have also appeared. They attempt to reduce the texture deviation and stretching caused when removing polygons from a mesh, as described in Hoppe *et al.* [2].

By simultaneously taking into account the geometrical and texture deviation of a mesh, one can *preserve the appearance of a mesh* when applying mesh simplification. If polygons can be removed without affecting appearance significantly, we can reduce render times for "free".

1.2 Aim

To survey the field for state-of-the-art mesh simplification algorithms that preserve the visual appearance of a mesh, and integrate these into *Configura's* (see Section 1.5) graphics pipeline. This will enable *Configura* to generate better *Level of Detail* (LoD) meshes for speeding up their rendering time. Currently, *Configura* only uses geometrical deviation for simplification.

Thereafter, we plan to evaluate each of these solutions by measuring their performance and ability to preserve the meshes' original appearance. In the end, the goal is to find the mesh simplification algorithm that suits *Configura's* requirements best based on our findings.

1.3 Research Questions

After implementing and measuring the performance of these mesh simplification algorithms, answers to the questions below should have been obtained. These will be used to decide on a suitable alternative for *Configura* and also other systems with a similar set of requirements.

1. What alternative *mesh simplification schemes* exist that *preserves the appearance* of a mesh?
2. Which of these alternatives give the best *performance* and *appearance preservation*? When:
 - a) Measuring the algorithm's *computation time* while targeting an *appearance threshold*?
 - b) Measuring the algorithm's *memory usage* while targeting some *appearance threshold*?
 - c) Measuring the real-time *rendering time* of the LoD-hierarchy of the *simplified mesh*? (by using the meshes generated according to the target *appearance threshold* above)
3. Which of the alternatives gives the best *appearance preservation* for a target *polygon count*?

1.4 Delimitations

Since there are many mesh simplification algorithms in previous work, a proper literature review would have to be done to find possible candidate solutions. Since our thesis' goals are mostly concerned with implementing and evaluating each solution, we've decided to base our choices on existing surveys and literature reviews to skip doing some of these ourselves.

Also, since implementing and doing measurements on all algorithms would take too long, we've decided to only pick an interesting subset of the algorithms presented in the surveys. More precisely, we've chosen to pick three different mesh simplification algorithms, one that doesn't take texture deviation into account and two that do. In addition, we'll also compare them to *Configura's* own existing mesh simplification scheme; for a total of four algorithms.

1.5 Background

This thesis was requested by *Configura AB*, a company in Linköping which provides space planning software. Their main product, *CET designer*, let's companies plan, create and render 3-D office spaces (among other things). These scenes can have a large amount of polygons that need to be rendered in real-time for customers to evaluate their creations in *CET designer*.

To allow larger scenes to be rendered with higher frame-rates (e.g. needed when exploring environments in *Virtual Reality* (VR), to prevent motion sickness), it would be beneficial to reduce the amount of polygons as much as possible. The meshes in these scenes usually have textures applied to them, and it's therefore important to keep their quality as high as possible.

While *Configura* already has a mesh simplification in their pipeline, it only accounts for surface simplifications, and doesn't take into account the texture appearance that might be degraded when applying mesh simplification. Hence, the given task was to integrate a new mesh simplification scheme that takes into account texture quality when simplifying a mesh.



2 Theory

2.1 Mesh Simplification

Quadric-Based Error Metrics

Appearance-Preserving Simplification

Texture Mapped Progressive Meshes

2.2 Metrics for Appearance Preservation

2.3 Measuring Algorithm Performance



Bibliography

- [1] Jonathan Cohen, Marc Olano, and Dinesh Manocha. “Appearance-preserving simplification”. In: *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*. ACM. 1998, pp. 115–122.
- [2] Hugues Hoppe. “Progressive meshes”. In: *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. ACM. 1996, pp. 99–108.
- [3] Jacob Kroon and Anna Nilsson. “Game Developer Index 2017”. In: *Based on 2017 Annual Reports. Dataspelbranchen* (2017).
- [4] Jerry O Talton. “A short survey of mesh simplification algorithms”. In: *University of Illinois at Urbana-Champaign* (2004).