Prof. Dr. Leif Kobbelt
Patrick Schmidt, Joe Jakobi

Visual Computing Institute | RWTH AACHEN UNIVERSITY

Basic Techniques in Computer Graphics

Assignment 8

Date Published: December 13th 2022, Date Due: January 20th 2022

- All assignments (programming and theory) have to be completed in teams of 3–4 students. Teams with fewer than 3 or more than 4 students will receive no points.
- Hand in **one solution per team per assignment**.
- Every team must work independently. Teams with identical solutions will receive no points.
- Solutions are due on January 20th 2022 via Moodle. Late submissions will receive zero points. No exceptions!
- Instructions for **programming assignments**:
  - Make sure you are part of a Moodle group with 3-4 members. See "Group Management" in the Moodle course room.
  - Download the solution template (a zip archive) through the Moodle course room.
  - Unzip the archive and populate the `assignmentXX/MEMBERS.txt` file. The names and student ids listed in this file **must match** your moodle group **exactly**.
  - Complete the solution.
  - Prepare a new zip archive containing your solution. It must contain exactly the files that you changed. **Only change the files you are explicitly asked to change in the task description.** The directory layout must be the same as in the archive you downloaded. (At the very least it must contain the `assignmentXX/MEMBERS.txt`.)
  - One team member uploads the zip archive through Moodle before the deadline, using the group submission feature.
  - Your solution must compile and run correctly **on our lab computers** by only inserting your **assignment.cc** and **shader files** into the Project. If it does not compile on our machines, you will receive no points. If in doubt you can test compilation in the virtual machine provided on our website.
- Instructions for **text assignments**:
  - Prepare your solution as a single pdf file per group. Submissions on paper will not be accepted.
  - If you write your solution by hand, write neatly! Anything we cannot decipher will receive zero points. No exceptions!
  - Add the names and student ID numbers of all team members to every pdf.
  - Unless explicitly asked otherwise, always justify your answer.
  - Be concise!
  - Submit your solution via Moodle, together with your coding submission.

Prof. Dr. Leif Kobbelt
Patrick Schmidt, Joe Jakobi

Exercise 1 Triangle Texturing [10 Points]

You are given two triangles T_1, T_2. T_1 is defined by A $= \begin{pmatrix} 1 \\ 2 \\ -1 \end{pmatrix}$, B $= \begin{pmatrix} 5 \\ 0 \\ 2 \end{pmatrix}$ and C $= \begin{pmatrix} 3 \\ 7 \\ 4 \end{pmatrix}$ in Camera Space. T_2 is defined by A' $= \begin{pmatrix} 0.2 \\ 0.2 \end{pmatrix}$, B' $= \begin{pmatrix} 0.8 \\ 0.2 \end{pmatrix}$, C' $= \begin{pmatrix} 0.2 \\ 0.8 \end{pmatrix}$ in UV space. Our goal is to compute the pixel coordinates for our texture image for the point P. To this end, you will solve the following tasks.

(a) 3D-Barycentric Coordinates [4 Points]

Assume P to be given in Camera Space as P $= \begin{pmatrix} 3.8 \\ 3 \\ 2.5 \end{pmatrix}$. First, compute the barycentric coordinates for our point P relative to T_1. You may use Cramer's rule, which should be familiar from Sheet 06. Provide your calculations for α, β and γ.

(b) UV Coordinates [2 Points]

After calculating the barycentric coordinates, calculate the corresponding point in UV space. Calculate the point P' in UV space, given T_2.

(c) Distortion [4 Points]

In the lecture you have learned about the order of operations when computing the UV Coordinate of a point. One option is to project the triangle onto the screen and then perform barycentric interpolation. Or, we can first perform barycentric interpolation and then project the triangle onto the screen. Does the ordering matter? If there are any differences, where do they stem from? Justify your answer!

Exercise 2 Anti-Aliasing [12 Points]

(a) [2 Points per column]

Aliasing in the context of rendering can have the following causes:

- Texture alias: Introduced by point-sampling textures which leads to jagged edges in the case of magnification and moirée pattern in the case of minification.
- Geometry alias: Caused by the binary decision for each pixel whether a pixel is rasterized or not. Noticeable on geometry silhouettes.
- Shader alias: Produced by the fragment shader. An example would be a fragment shader that outputs a black and white checkerboard based on the world coordinates.

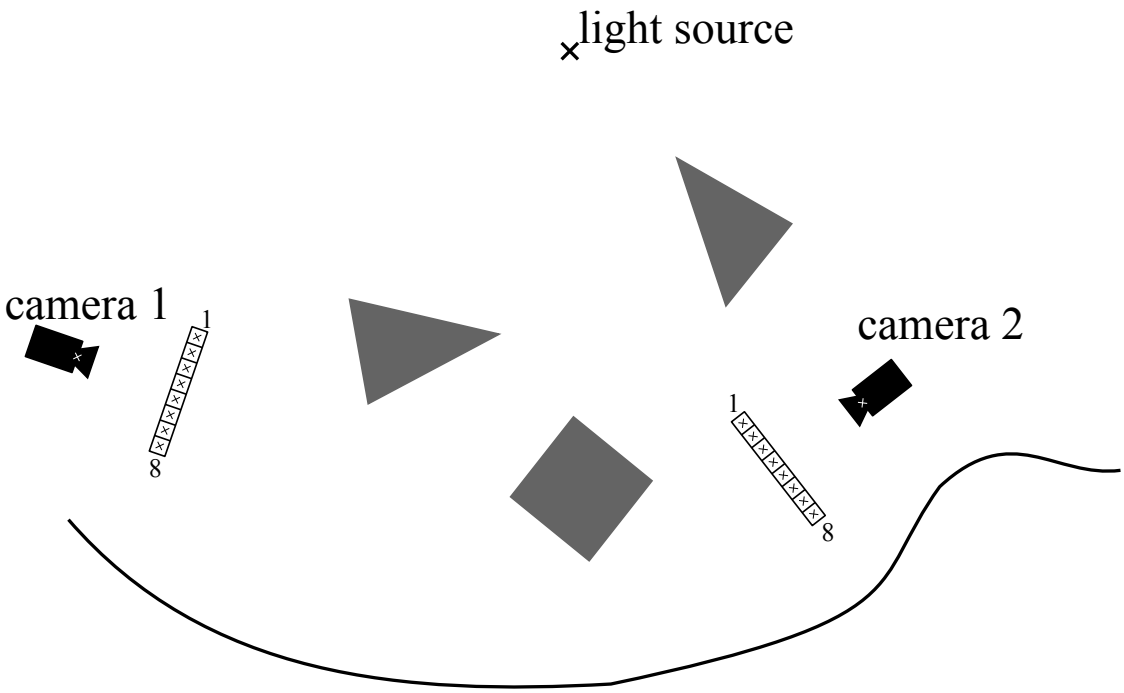
There exist several methods to reduce the effect of those artifacts such as Mipmapping/linear interpolation, Multisample Anti-Aliasing (MSAA), Post-processing Anti-Aliasing (for example FXAA) and Supersample Anti-Aliasing (SSAA). Indicate which method helps against which type of alias by filling the table below with either check marks ✓ (the method reduces the type of alias) or crosses × (the method does not help with the type of alias).

	Mipmapping	MSAA	FXAA	SSAA
texture				
geometric				
shader				

(b) [4 Point]

Shortly explain, **in your own words**, what *Multisampling* and *Supersampling* techniques do. Discuss the difference between both techniques.

Exercise 3 Shadow Volumes [12 Points]



Consider the scene given above, containing a light source, three occluders (grey objects) and two cameras, where the center of projection is given by the white cross inside the camera.

(a) [3 Points]

Draw the shadow volumes generated by the light source an the occluders (make sure your lines are reasonably straight, e. g. with a ruler). Clearly mark which sides of the shadow volumes are *front-facing* towards camera 1 and which sides are *back-facing*.

(b) [3 Points]

Explain the relationship between ray casting and the Stencil Buffer-based shadow volume algorithms.

(c) [3 Points]

Use ray casting through the centers of the pixels (small black crosses) to compute for every pixel (1-8) the Stencil Buffer entry assigned to it by the normal shadow volume algorithm. Specify those entries for both cameras.

(d) [3 Points]

What do you notice looking at the Stencil Buffer entries for camera 2? Explain why this happens and how to fix the problem.

Exercise 4 Shadow Maps [8 Points]

Another method to render shadows, besides using shadow volumes, is the Shadow Maps algorithm presented in the lecture. Its basic idea is to render the scene from the perspective of the light source and store the depth values observed by the light source into a special texture, the so-called shadow map. When rendering the scene from the view of the camera, the shadow map is used to determine which points seen from the camera are also seen from the light source and are therefore lit.

(a) [2 Points]

Let M_c be the view matrix of the camera and let M_l be the view matrix of the light source (with respect to which the shadow map has been rendered). Given an arbitrary point $p \in \mathbb{R}^3$ in the local coordinate system of the camera, specify the formula that transforms it into its representation p' in the coordinate system of the light source.

(b) [2 Points]

Let $f_{\text{SM}} : \mathbb{R}^3 \to \mathbb{R}$ be the function that takes a point q in the coordinate system of the light source and returns the depth value stored in the shadow map at the position onto which q projects on the image plane of the light source. Specify the condition (in terms of M_c, M_l, and f_{SM}) that answers whether a point p (represented in the local coordinate system of the camera) lies in the shadow according to the Shadow Maps technique.

(c) [4 Point]

What are the two types of aliasing that can occur when using shadow maps and which one can be fixed by using Perspective Shadow Maps? Explain your answer.