Prof. Dr.-Ing. Ralf Steinmetz
Multimedia communications Lab

Dipl. Inf. Robert Konrad
Polona Caserman, M.Sc.

TECHNISCHE UNIVERSITÄT DARMSTADT

„Game Technology“
Winter Semester 2017/2018

Solution 10

For bonus points upload your solutions until **Tuesday, January 16th, 13:29**

General Information

- The exercises may be solved by teams of up to three people.
- The solutions have to be uploaded to the Git repositories assigned to the individual teams.
- **The submission date (for practical and theoretical tasks) is noted on top of each exercise sheet.**
- If you have questions about the exercises write a mail to game-technology@kom.tu-darmstadt.de or use the forum at https://www.fachschaft.informatik.tu-darmstadt.de/forum/viewforum.php?f=557

P10 Practical Task: Coarse Texture Streaming (5 Points)

Implement coarse texture streaming – load in higher resolution textures for close objects, kick out higher resolutions for far away objects. Try to keep the framerate high and steady.

https://github.com/KTXSoftware/Exercise10.git contains additional code to help you out. You can either copy the code changes manually or just pull them into your own repository using
git pull https://github.com/KTXSoftware/Exercise10.git

You can find the solution code for the practical tasks at
https://github.com/TUDGameTechnology/Solution10.git.

T10 Theoretical Tasks: Compression (5 Points)

T10.1 Hardware

What makes it so important that texture compression algorithms are directly supported by the hardware?

Reading pixels from textures is the most fundamental and speed critical operation of GPUs. Compressed textures are only really useful when they don't slow this operation down.

T10.2 Artifacts

ETC is a lossy texture compression algorithm. Describe what characteristics an image should have to make those losses clearly visible.

Sharp contrasts across block boundaries – for examples black lines across a white background.
Alternatively, too many colors in one block.

T10.3 Tilemaps

Outline an algorithm to display tilemaps correctly in a 3D environment.

MegaTextures (see script) can solve this problem quite efficiently by saving the completely rendered tilemap in an image.
Alternatively, a custom texture sampling procedure could be implemented in a fragment shader.
Many more alternatives are possible.