Prof. G. Zachmann
A. Srinivas

University of Bremen
School of Computer Science
CGVR Group
November 19, 2014

Winter Semester 2014/15

Assignment on Virtual Reality and Physically-Based Simulation - Sheet 2

Due Date 2. 12. 2014

Exercise 1 (Depth Cues, *4 Credits*)

As you know, there are a number of depth cues that our human eye and brain uses to estimate the 3D shape of objects, their distance from the eye, and depth relationships between objects. The most important ones are: occlusion, perspective foreshortening, shading, shadowing (self-shadowing and mutual shadowing), atmospheric perspective (= blue shift and contrast reduction proportional to depth), texture gradient, familiar size of objects learned by experience, disparity, convergence of the eyes, focus of the eyes, motion parallax.

Group or classify those depth cues according to the following, different criteria:

1. Which ones can be generated by software only (i.e., using a standard PC and monitor), and which ones require hardware beyond the standard PC/monitor (e.g., tracking or special displays)?
2. Which ones can be seen with one eye only and which ones require binocular vision?[1]
3. Which depth cues require tracking of the user? (E.g. tracking her/his eyes on the HMD)
4. Is there a fundamental difference, in your opinion, between the depth cue "familiar size of objects learned by experience" and all the others? Please elaborate. (Note: there is no right or wrong answer here)

Hint: Note that most depth cues will fall into several classes and classification according to criteria (1), (2), and (3) are orthogonal (independent) to each other.

Can you find other depth cues that are not in the above list?

Exercise 2 (Disparity and parallax, *2 Credits*)

1. Consider the case where the two eyes are converged at infinity, What is the theoretical range of the disparity of objects? (assuming ideal eyes, i.e., pinhole lens, retina stretches around the whole interior of the eyeball, etc.)
2. Is it possible that a (virtual) point can have zero parallax yet nonzero disparity? Explain.

Exercise 3 (Practical horopter construction, *3 Credits*)

Construct the horopter using real-world building blocks or other cylindrical objects.

[1] The latter class of depth cues consequently needs the software to generate two separate images, and it needs two channels by which the two images can be conveyed to their respective eyes without getting into the "wrong" eye.

Place a subject (e.g., one of your group members) close to a table, such that she/he looks edge-on at the table. Make sure that the background that the subject sees behind the table is as uniform as possible (e.g., white wall). Position one block on the table. Have someone else (the experimenter) place other blocks on the table, *guided by the subject*, such that they all appear to have exactly the same depth as the first block, see to that the subject should always fixate (focus) on the first block while observing other blocks. (This usually requires some iterations, moving the new block back and forth.) The experimenter should try to help the subject such that the experimenter does not influence or bias the subject.

Capture the final setup of the blocks by a photo exactly from the top. Interpolate the block positions by a freeform curve (you can do that very approximately by hand).

Repeat the above experiment with first block at 3 different depths (note that differences in these depths should be at least 20 cm). (for instance at 10 cm, 50 cm and 100 cm)

Exercise 4 (Pre-distortion for HMDs, *2 Credits*)

Consider the problem, as discussed in class, that the optical systems (lenses) in HMDs usually incur a considerable distortion of the image, in order to achieve a field-of-view that is much wider than the LCD panels would provide without it.

Let (x'', y'') be a pixel position in the image *after the lens distortions*. Denote the distortion function of the HMD's lenses by f, i..e, $(x'', y'') = f(x', y')$, where (x', y') is a pixel position in the frame buffer (i.e., *before* the distortion).

Assume that we have rendered the virtual environment in a first pass in an off-screen buffer, and we render in the second pass a single quad over the whole frame buffer, so that the pixel shader gets invoked once per output pixel (x', y'). Obviously, $(x', y') = g(x, y)$, where (x, y) is a pixel position in the off-screen buffer from the first pass, and g is the pre-distortion function.

Which function does the fragment shader (a.k.a. pixel shader) need to implement?