Department of Computer Science Vrije Universiteit Dr. T. Kielmann, Dr. F. Seinstra	Exam **Computer Graphics** Exam code: 4001061 7 – 2 – 2011

This is a *"closed book"* exam.
No printed materials or electronic devices are admitted for use during the exam.
You are supposed to answer the questions **in English**.

Wishing you lots of success with the exam!

Points per question (maximum)

Q	1			2			3	4		5		6		7	8		
	a	b	c	a	b	c		a	b	a	b	a	b	a	a	b	c
P	4	2	2	4	7	3	12	3	7	4	6	4	6	8	6	6	6

Total: 90 — To pass the exam, it is sufficient to get at least 45 points.

1. The Synthetic Camera Model

a) The Synthetic Camera Model is based on the working of an old-fashioned bellows camera. Describe how an image is constructed using the Synthetic Camera Model.

b) Which 4 parameters of the Synthetic Camera Model determine which objects appear in the image?

c) One problem of a bellows camera (or the Synthetic Camera Model) is that it produces upside-down images. What 'trick' is used to correct for this problem, and why does this work?

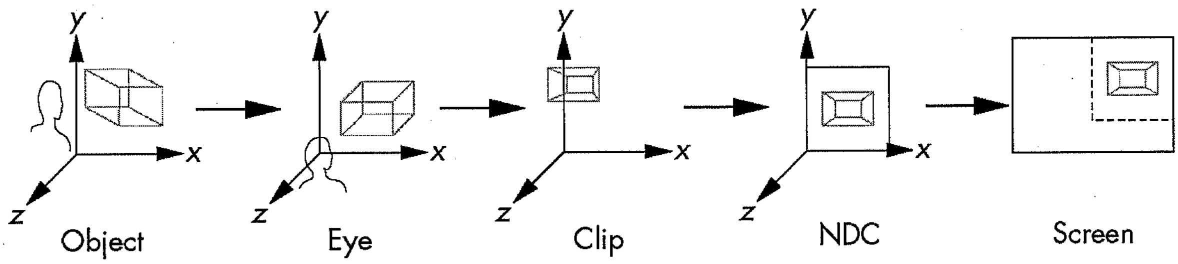
2. Events and Interaction

a) For which purpose do interactive graphics environments (like OpenGL) use 'events', and give at least 3 advantages of the Event Mode over other possible forms of interaction.

b) Explain how the Java Event Model works, covering each of its constituent components and their relationships.

c) Name at least 3 interface methods that are called automatically by the OpenGL run-time system – and mention for each of these the situation(s) in which these will be called.

3. Coordinate System Transformations



This image shows the transformations from objects to the screen. Explain the diagram from left to right, focusing on the following terms (use not more than one sentence per term):

1. clip coordinates
2. clipping
3. eye coordinates
4. model-view matrix
5. NDC coordinates
6. object coordinates
7. orthographic projection
8. projection
9. rasterization
10. screen coordinates
11. viewport
12. view volume

4. Picking

a) Explain (briefly) the logical input operation called *picking*. What is the major problem caused by the pipeline rendering architecture for implementing picking?

b) Explain how OpenGL allows to implement picking. Do so by explaining the following terms: rendering mode, object stack, pick matrix, clipping, select buffer, and changes/additions to the scene rendering code and the MouseListener interface implementation (or: mouse callback functions).

5. Illumination

a) Explain the difference between a *local* illumination model and a *global* illumination model, and give an example of each of these.

b) Explain *in detail* how the diffuse component of OpenGL lighting is computed from (1) the normal vector n of a vertex, and (2) a vector l, the direction of light.

6. Level-of-Detail

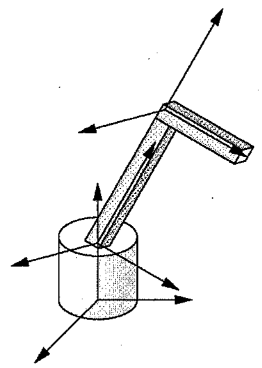
a) Explain the differences between *discrete* and *continuous* level of detail algorithms, and give an example of both techniques.

b) Which are the four OpenGL magnification and minification functions for mapping a texture to a surface when using mipmaps? Explain how each function gives a mapping from texels in the mipmaps to a pixel.

7. Robot Components

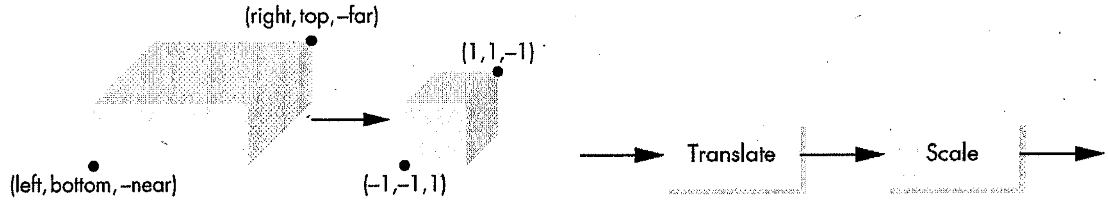
The robot arm shown on the right can be moved in three ways, as shown in the diagram: The base can be rotated on the ground by an angle α, the lower arm can be rotated by angle β relative to the base, and the upper arm can be rotated by angle γ relative to the lower arm.

Write a function `robot(alpha,beta,gamma)` (in Java or in C) that draws the robot arm! You can assume there are functions `base()`, `lower()` and `upper()` that draw the respective robot parts. Dimensions of the three parts are available as you need them. For the rest, use OpenGL function calls.



8. glOrtho

A call to `glOrtho(left, right, bottom, top, near, far)` defines a viewing volume for orthogonal projection. The implementation (the code) of `glOrtho` multiplies a matrix P to the current transformation matrix (CTM).



As shown in the diagram (left), the effect of the maxtrix P is to map the viewing volume to the canonical volume. The right side of the diagram indicates that this mapping can be done in two steps, first a *translate* (moving the center of the viewing volume to the origin) and then a *scale*, bringing the volume to the size $2 \times 2 \times 2$.

Hint: To save work, simply abbreviate *left, right, bottom, top, near, far* by their first letters: *L, R, B, T, N, F*.

a) The translation step can be performed by using a translation matrix $T(dx, dy, dz)$. Compute dx, dy, dz and show that your $T(dx, dy, dz)$ translates the points $[L, B, -N, 1]^T$ and $[R, T, -F, 1]^T$ to coordinates that lie symmetrically around the origin!

b) The scaling step can be performed by using a matrix $S(sx, sy, sz)$. Compute sx, sy, sz and show that your $S(sx, sy, sz)$ scales the translated corners of the viewing volume (the results from part a) to the respective corners of the canonical viewing volume, namely $(-1, -1, 1)$ and $(1, 1, -1)$!

c) Compute the overall matrix P from $S(sx, sy, sz)$ and $T(dx, dy, dz)$! Does it matter if you compute either $P = S \cdot T$ or $P = T \cdot S$? Explain why!