Department of Computer Science	Exam **Computer Graphics**
Vrije Universiteit	Exam code: 4001061
Dr. T. Kielmann, Dr. F. Seinstra	18 – 12 – 2009

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	a	b	c	a	b	a	b	c	d	e	a	a	b	a	b	c	a
P	2	4	3	6	3	3	7	4	4	4	3	3	8	9	5	6	6	6	4

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. One problem of such a camera is that it produces upside-down images. What 'trick' is used to correct for this problem, and why does this work?

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

2. Events and Interaction

a) Interactive (graphics) applications can obtain measures from input devices in three different modes: Request Mode, Sample Mode, and Event Mode. Give at least 3 advantages of the Event Mode over the other two modes.

b) Explain how the Java Event Model works. Explain the involved components and their relationshsips.

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. 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).

4. Shearing

Figure 1 shows a cube, as well as a version of the same cube sheared along the x-axis (relative to the y-axis). Figure 2 shows how this shear along the x-axis is characterized, given a single angle θ. It also presents the related shearing matrix.

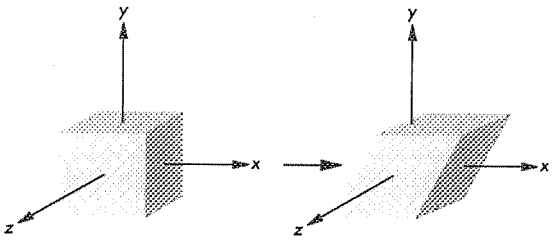
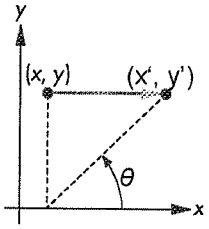


Figure 1



$$H_x(\theta) = \begin{bmatrix} 1 & \cot\theta & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Figure 2

a) Implement a Java method that manipulates OpenGL's Current Transformation Matrix by way of the shearing matrix $H_x(\theta)$:

```
public void shearX_relativetoY(float theta) {
        // Your implementation should follow here.
        // Pure Java is preferred; Java-like or C-like pseudo code is fine, too.
}
```

b) Give pictorial characterizations similar to Figure 2 (left) for shearing along the y-axis (relative to the z-axis), and shearing along the z-axis (relative to the x-axis), using angles α and β respectively.

c) Write down the transformation matrices for the two transformations of part b) above.

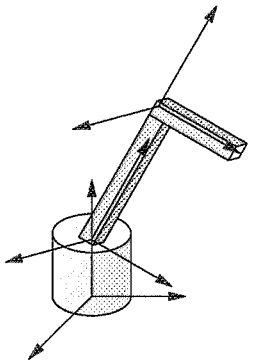
d) Write down a single shearing matrix that combines the effects of the three shearings of Figure 2 and question part b) above.

e) Suppose we have implementations available for methods *shearX_relativetoY*, *shearY_relativetoZ*, and *shearZ_relativetoX*:
is it possible to implement a single method *shearXYZ_relativetoYZX(theta, alpha, beta)* by a combination of calls to the first three methods? If not: why not? If so: how?

5. 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.



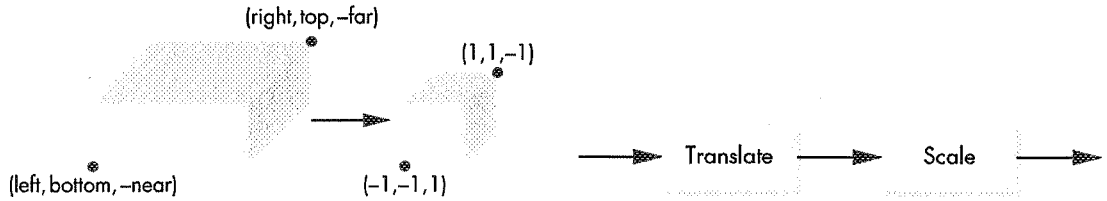
6. Lighting

a) We distinguish between different kinds of light sources. For **ambient light**, **point light sources**, and **spotlights** explain how they illuminate a given object within a scene. Explain which factors contribute to this illumination. If you can, give formula's for the respective illumination function.

b) For **specular reflection**, how (and based on which parameters) is the light intensity computed for the surface of a given object? Again, give a formular if you can.

7. 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!

8. Polygon Filling

Explain the concept of *winding numbers* and how they can be used for polygon filling. For the polygon shown, give the winding numbers for the areas A, B, and C.