Graphics 2009/2010

T2

Endterm exam

Mon, Nov 02, 2009, 08:30–10:30

- **Do not open this exam until instructed to do so.**
- **Read the instructions on this page carefully.**

- You may write your answers in English, Dutch, or German.
  Use a pen, not a pencil. Avoid usage of the color red.
- Write down your name and student number on every paper you want to turn in.
  Additional paper is provided by us. You are not allowed to use your own paper.
- You may not use books, notes, or any electronic equipment
  (including your cellphone, even if you just want to use it as a clock).

- The exam should be doable in 1.5 hours. You have max. 2 hours to work on the questions.
  If you finish early, you may hand in your work and leave, except for the first half hour of the exam.
- When you hand in your work, have your student ID ready for inspection and write your name and student number on the list of participants.

- The exam consists of 7 problems printed on 6 pages (including this one).
  It is your responsibility to check if you have a complete printout.
  If you have the impression that anything is missing, let us know.
- The maximum number of points you can score is 18.
  You need at least 17 points to get the best possible grade.

Good luck!

1

Problem 1: Perspective projection

Subproblem 1.1 [1.5 pt] In order to transform objects in world space coordinates into objects in camera coordinates, we need an orthonormal base $\vec{u}$, $\vec{v}$, $\vec{w}$ with origin at the eye position $\vec{e}$. Explain how we can construct such an orthonormal basis given the eye position $\vec{e}$, the gaze vector $\vec{g}$, and a view-up vector $\vec{t}$. (Note: construct the orthonormal base in a way that we are looking into the negative $\vec{w}$-direction, as we did in the tutorials.)

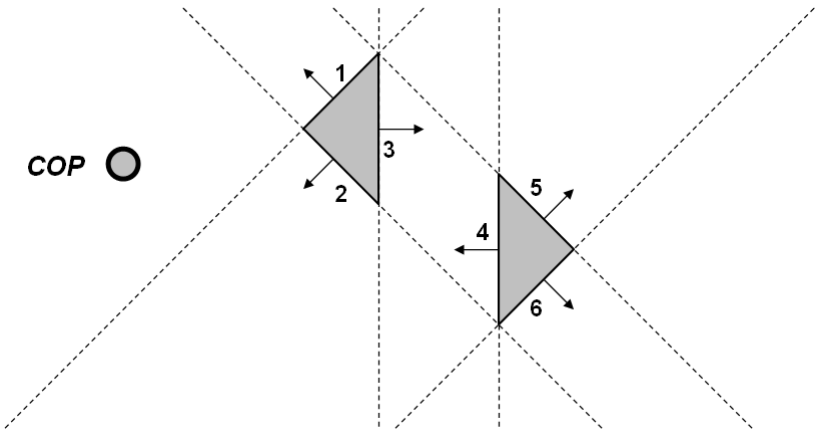
Subproblem 1.2 [0.5 pt] We have seen that we can do perspective projection by matrix multiplication. The precise matrix needed is not relevant here; let's call it M_p. Assume (x,y,z) is a point in $\mathbb{R}^3$. Before the homogeneous divide, we have

$$M_p \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} x \\ y \\ z\frac{n+f}{n} - f \\ \frac{z}{n} \end{pmatrix}$$

Now assume that we have two random points $\vec{p}_1 = (x_1, y_1, z_1)$ and $\vec{p}_2 = (x_2, y_2, z_2)$ within the view frustum. Show that M_p does not change their order along the z-axis.

Problem 2: Hidden surface elimination

The scene below consists of two triangles that are defined by the line segments 1, 2, 3 and 4, 5, 6, respectively, and a camera view point (i.e. the center of projection *COP*). The arrows indicate the normal vectors of the related segments. The dashed lines are not part of the input, but have been included to illustrate the positions of the objects with respect to each other.



Subproblem 2.1 [1 pt] Create a Binary Space Partitioning tree (BSP tree) for the scene illustrated above. (Note: add the segments in the order $1, 2, \ldots, 6$ to your tree.)

2

Subproblem 2.2 [1 pt] Assume we have a scene that contains only closed polygons that are defined by connected line segments (in 2D) or connected triangles (in 3D). Further assume we have normal vectors for each segment/triangle that point to the outside of the closed polygone. Also, the camera position is not within any of the closed polygones. (Note: the above illustration is a good example for such a scene in 2D) Use this setup to explain what *backface culling* is and how we can easily implement it.

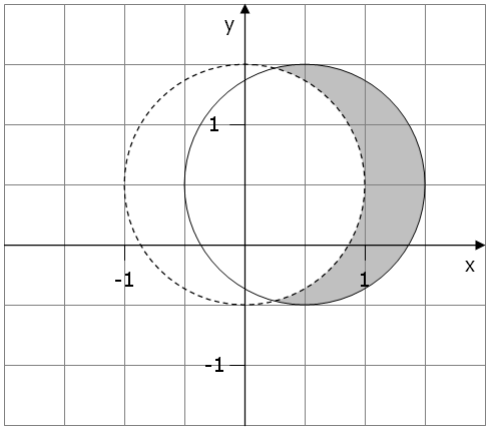
Subproblem 2.3 [0.5 pt] Assume you want to get the correct drawing order for the segments in the scene illustrated above by first applying backface culling and then using the BSP tree approach. Create the BSP tree you would get after backface culling (add segments to your tree in the same order as in 2.1).

Subproblem 2.4 [0.5 pt] How do the trees created in 2.1 and 2.3 change if the COP is moved to the top-right corner of the scene, i.e. the area defined by the negative side of the lines through 4 and 6 and the positive side of the line through 5?

Problem 3: Ray tracing

Subproblem 3.1 [1 pt] Assume $\vec{p}$ is a point in barycentric coordinates, i.e. $\vec{p} = \vec{a} + \beta(\vec{b} - \vec{a}) + \gamma(\vec{c} - \vec{a})$ with two parameters β, γ and the three vectors $\vec{a}$, $\vec{b}$, $\vec{c}$ that define our barycentric coordinate system. How can we easily check if $\vec{p}$ lies within the triangle defined by $\vec{a}$, $\vec{b}$, $\vec{c}$? (Note: "within" here means including the borders of the triangle.)

Subproblem 3.2 [1 pt] Assume we have two circles C_1 and C_2 in $\mathbb{R}^2$ both with radius 1 centered around $\vec{c}_1 = (0, 0.5)$ and $\vec{c}_2 = (0.5, 0.5)$, respectively (cf. image below).



(a) Explain how we can use these two circles and Constructive Solid Geometry (CSG) to create the image of a waxing moon as illustrated by the gray area in the image above.

(b) Explain how we can calculate the intersection of the ray defined by $y = 0.5$ and the moon image created in (a) with CSG and give the values of the intersection points. (Note: you have to explain how you can get these intersection points. Just writing them down without explanation will not give you any credit!)

Subproblem 3.3 [1.5 pt] Assume we have a 2D image containing many triangles. Explain how we can use BSP trees to separate the space in a way that each of the resulting cells contains only two triangles at the most. (Note: it is required to explain how we split the space and how we create the related BSP tree. It is however *not* required to explain how we traverse this tree when doing ray/object intersection tests and doing so will not give you any extra credit.)

Problem 4: Texture mapping

Subproblem 4.1 [1.5 pt]

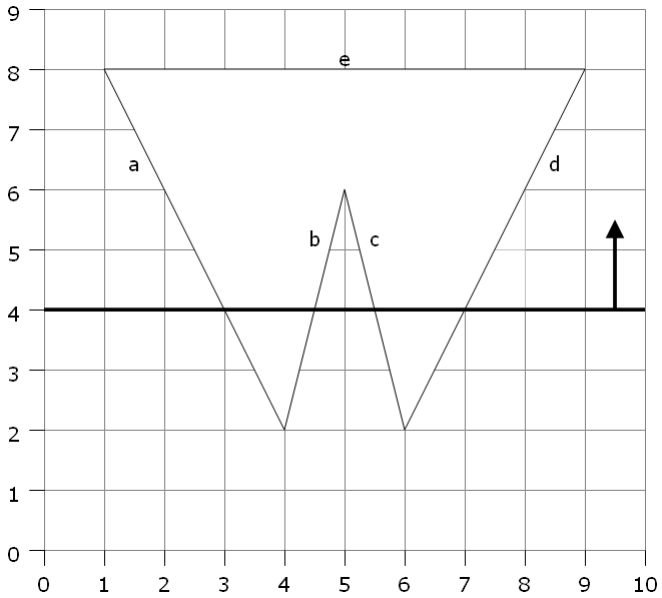
(a) Give a procedure that creates a checker board-like 2D texture consisting of equal sized squares with width = length = π and alternating black and white color.

(b) How do you have to change your procedure from (a) in order to allow users to control the size of the squares with a parameter w?

Subproblem 4.2 [1 pt] Give a short description of environment mapping.

Problem 5: Triangle rasterization

Subproblem 5.1 [2 pt] Assume we want to rasterize the polygone with edges a, b, c, d, and e that is illustrated in the image below using the scanline approach (note: we assume a horizontal scanline that moves vertically from the bottom to the top as illustrated by the arrow in the image; also, we assume that the integer coordinates shown in the image represent pixel centers).



For the scanline approach, we use the two data structures *Edge Table* (ET) and *Active Edge Table* (AET).

(a) How and where is edge a, i.e. the line segment between the points $(4,2)$ and $(1,8)$ (cf. image above) represented in the Edge Table? Explain why this is a well-defined representation of this edge/segment. (Note: do not just write down the values but explain their meaning.)

(b) What values are stored in the Active Edge Table at scanline number 4 which is illustrated in the image? Write the values down, explain what they mean, and specify what points we have to rasterize.

Subproblem 5.2 [1 pt] Assume we have two vertices $p_1 = (x_1, y_1)$ and $p_2 = (x_2, y_2)$ of a triangle with colors c_1 and c_2. We want to use *Gouraud shading* to color the edge between these two vertices, i.e. we want to linearly interpolate between the two color values. (Note: for simplicity, we assume that a color c_i is not expressed by three values (e.g. RGB) but by a single scalar.) We can easily integrate this into the scanline approach that we are using for rasterization by calculating a Δc-value similarly to the Δx-value needed for rasterization.

Give the concrete formula for calculating this Δc-value and explain how it is used to calculate the colors for each pixel on the edge.

Problem 6: Radiosity and shadows

Subproblem 6.1 [1 pt] Explain the meaning of the following formula, which can be used to calculate the radiosity B_i of a patch A_i:

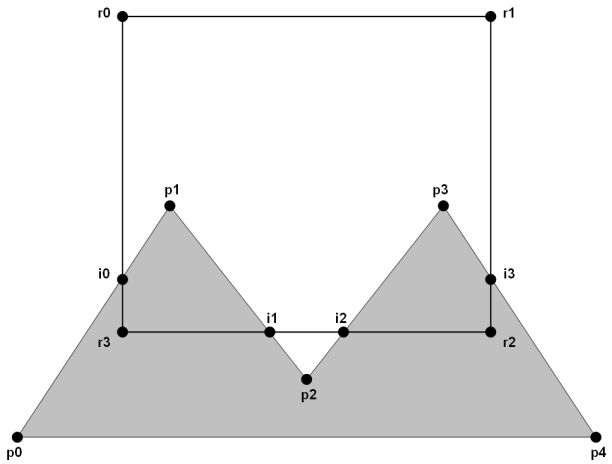
$$B_i = E_i + \rho_i \sum_j B_j F_{ij}$$

Subproblem 6.2 [1 pt] One approach to integrate shadows into our images is via *shadow volumes*. In practice, they are often implemented using a Stencil buffer which in turn contains a counter for each pixel that we want to rasterize.

Explain what this counter represents and how it is used to decide if we have to draw an object in a shadow or not. (Note: you just have to explain the meaning of the counter and what the related value represents. It is *not* required to explain the Stencil buffer and doing so will not give you any extra credit.)

Problem 7: Clipping

Subproblem 7.1 [1 pt] We want to use the Weiler-Atherton algorithm to clip the gray polygone illustrated below agains the clipping area represented by the rectangle. Construct the graph that is used by this algorithm for the given rectangle and polygone.



Subproblem 7.2 [1 pt] In the pseudocode for using the graph from the Weiler-Atherton algorithm to draw the clipped polygones, we have the following line:

- *Continue, changing from polygone boundary to clipping region and the other way around at outgoing and incoming intersection vertices, respectively, until we reach the starting vertex.*

When applying this algorithm to the example from the previous subproblem, we only change once, namely from the clipping region to the polygone region. Give an example where we have to do more than one change (note: do this by drawing a rectangular clipping area and an appropriate polygone).