Graphics 2007/2008

Second Exam

Thu, Jan 31, 2008, 16:30–18:30

Solutions (sketch) and comments

Errors and omissions excepted!

Problem 1: Perspective projection

[1 pt] Subproblem 1.1 (Camera transformation)
Given a camera position and a scene containing an object: What does the *gaze vector* specify in this context? Assume the center of the object is located at point (1,2,3) and we have the camera placed at position (3,2,1). Calculate the gaze vector for the given situation if we want the object to be placed directly in the center of the image.

[1 pt] Subproblem 1.2 (Perspective transformation)
After multiplication with the perspective transformation matrix M_p and the following homogenization, the point $\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$ contains the values $\begin{pmatrix} nx/z \\ ny/z \\ n+f-\frac{fn}{z} \\ 1 \end{pmatrix}$.

Show that all points on the near clipping plane n have been projected onto themselves by these operations.

[2 pt] Subproblem 1.3 (Windowing transformation)
In the lecture, the orthographic projection matrix M_o was introduced as the matrix resulting from the multiplication of the following three matrices:

$$M_o = \begin{pmatrix} \frac{m}{2} & 0 & 0 & \frac{m}{2}-\frac{1}{2} \\ 0 & \frac{n}{2} & 0 & \frac{n}{2}-\frac{1}{2} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \frac{2}{r-l} & 0 & 0 & 0 \\ 0 & \frac{2}{t-b} & 0 & 0 \\ 0 & 0 & \frac{2}{n-f} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & -\frac{l+r}{2} \\ 0 & 1 & 0 & -\frac{b+t}{2} \\ 0 & 0 & 1 & -\frac{n+f}{2} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Describe what each of these matrices does. (Note: l,r,t,b,n,f specify the left, right, top, bottom, near, and far plane of the orthographic view volume, respectively, and mxn is the size of the projected image.)

1.1 The gaze vector is the vector specifying the viewing direction.
Here, it is "object center" (1,2,3) minus camera position (3,2,1) $\rightsquigarrow \begin{pmatrix} -2 \\ 0 \\ 2 \end{pmatrix}$

Note: This problem is (almost) identical to problem 1, Tutorial 5

1.2 Point on near clipping plane $\Leftrightarrow z = n$ (x,y don't matter)
$\rightsquigarrow$ $n + f - \frac{fn}{n} = n$ ✓

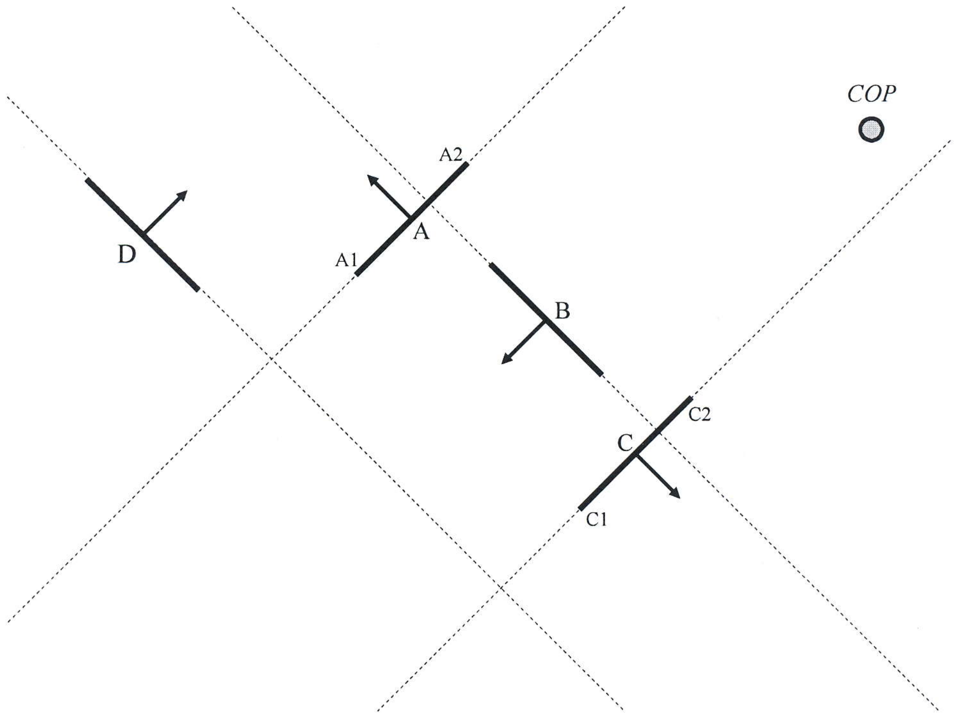
Note: This problem is similar to one given in last year's exam
(But here the easier case, i.e. after homogenization was requested)

1.3 From right to left:
1st matrix: move coords to the center of the orthographic view volu[me]
2nd matrix: scale everything into the canonical view volume
3rd matrix: project everything to the final image

Problem 2: Hidden surface elimination

[1 pt] Subproblem 2.1
The scene below consists of 4 line segments and a camera view point (i.e. the center of projection *COP*). The normal vectors of the segments point to the visible side. The dashed lines are not part of the input, but indicate where the supporting lines of the segments intersect the other segments.

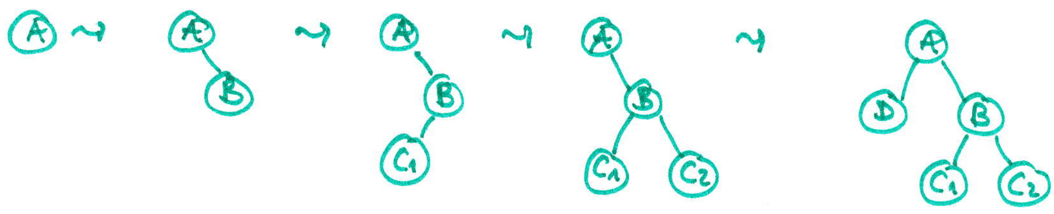
Illustrate the construction of a BSP tree for the situation shown in the image. **Important**: Illustrate how you build this tree by drawing a new tree for each new node that is added. Use the notation given in the image, i.e. specify the segments using the letters A, B, C, and D. If you have to split up a segment, use the notation A1, A2, C1, and C2, respectively, as indicated in the image.



[2 pt] Subproblem 2.2
Give a short explanation about how to get the projection order for a given BSP tree and camera view point *COP* using the situation in the image and the tree you constructed in the previous subproblem. Give the order in which the segments are drawn based on your tree.

[1 pt] Subproblem 2.3
How do your results from subproblem 2.1 and 2.2 change if the *COP* is in the center of the far left side of the image?

2.1



(Note: Other solutions exist since the BSP tree is not unique)

2.2 We start with the root and check if COP is on the positive or negative side (i.e. visible or not visible side) of the associated segment.

If it is on the positive/visible side, we draw the negative subtree 1st, then the root, then the positive subtree, in order to guarantee that no segment which might block the view between the root and COP is drawn after the root.

Similarly, if it is on the negative/not visible side, we draw the positive subtree 1st, then the root, then the negative subtree.

We do this recursively for every subtree.
In the example, we get D - A - C1 - B - C2

2.3 The BSP tree is independent of COP so nothing changes here. However, the drawing order ~~of~~ obviously changes with the COP. In the example we get C_2 - B - C_1 - A - D

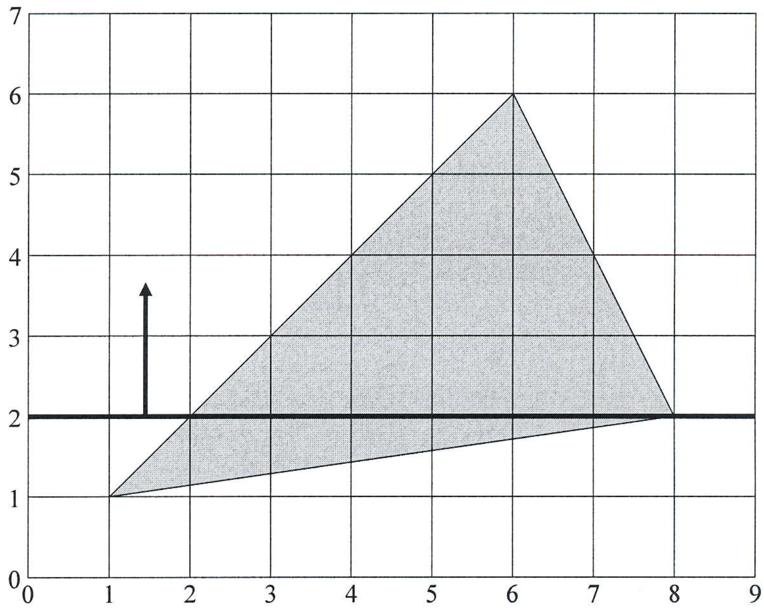
Notes: - 2.1 and 2.2 are very similar to problem 1, tutorial 6 (simpler example, but more explanations)

- I wasn't sure if I should bring this problem in the exam since I made a mistake in the lecture myself. However, this turned out to be the question with the most correct answers! Maybe I should make more mistakes in future lectures? :) (Seriously: It is a good example showing that you do not learn by listening but by practicing!)

Problem 3: Triangle rasterization

[1 pt] Subproblem 3.1
In the lecture, we used two data structures in the algorithm for triangle rasterization via scan-line conversion: The *edge table* and the *active edge table*. What does each of these tables contain?

[1 pt] Subproblem 3.2
Look at the image below (note: assume that the scan-line is horizontal and moves vertically from the bottom of the image to the top as illustrated by the black line and the associated arrow). Give the values for the edge table that are stored for scan-line number 2. Give the values for the active edge table that are stored for scan-line number 3.



[1 pt] Subproblem 3.3
Explain how Gouraud shading can be incorporated in the scan-line conversion algorithm.

3.1 see lecture slides

3.2 Edge table → 2: $(8, 6, -\frac{1}{2})$

Active edge table → 3: $(3, 6, 1)$, $(7.5, 6, -\frac{1}{2})$

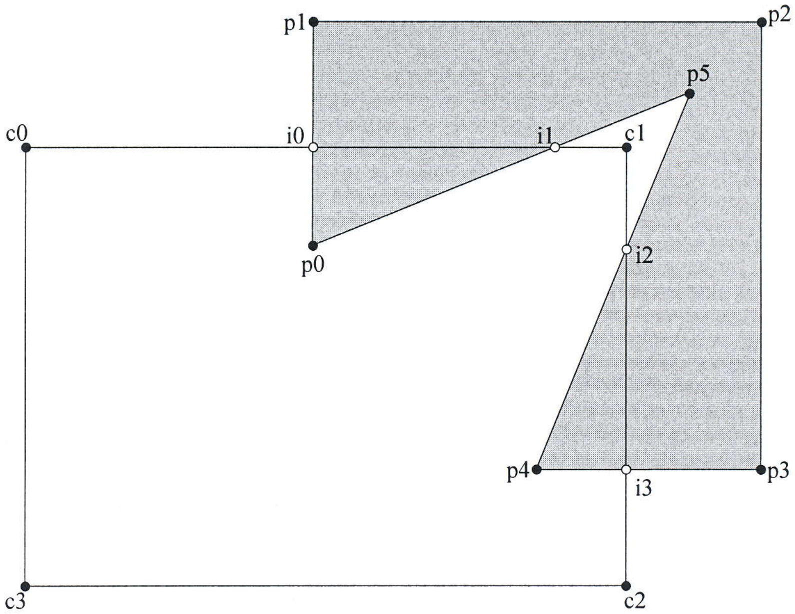
3.3 see problem 4, tutorial 6

} cf. lecture slid
+ problem 3, tutori

Problem 4: Clipping

[1 pt] Subproblem 4.1
Explain how the graph used in the Weiler-Atherton algorithm is constructed.

[1 pt] Subproblem 4.2
Construct this graph for the example given below.

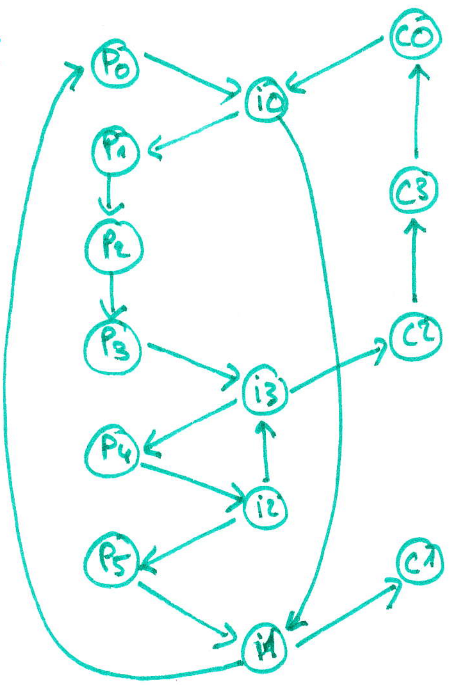


[1 pt] Subproblem 4.3
Explain how the graph is used to determine the resulting polygons.

4.1 cf. problem 2(a), tutorial 9

4.2 cf. problem 2(c), tutorial 9

4.3



Notes: Many students did not do this exercise at all, which was quite surprising for me.

I actually included this problem in the exam in order to give everyone a chance to pass, even if you didn't prepare that well (It was almost identical to the tutorials, only that a slightly easier polygone was used)

Problem 5: Texture mapping

[2 pt] Give a short description of the following techniques:

- Bump mapping
- Environment mapping

A bump map is a 2D or 3D array of vectors. These are added to the normals at the points for which we do shading calculations.

The effect is an apparent change of the ~~geo~~ geometry of the object.

The goal of environment mapping is to make objects appear to reflect their surrounding specularly.

For this, we 1st place a cube around the object, and project the environment of the object onto the planes of the cube in a preprocessing stage (≙ texture map)

During rendering, we compute a reflection vector, and use that to look-up texture values from the cubic texture map.

(Note: The 2nd part of this problem was taken from last year's retake)

Problem 6: Radiosity

[2 pt] Explain the meaning of the following formula, which is used to calculate the radiosity B_i of a patch A_i:

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

The radiosity of a patch A_i is the sum of

→ the energy E_i emitted by the object itself (eg. if it is a light source) and

→ the reflection of the light which in turn is reflected by the other objects and defined by

→ the sum of the radiosities emitted by all objects B_j multiplied with the form factor F_{ij} which specifies "how well 2 objects see each other"

→ weighted by the reflective factor ρ_i of patch A_i (which depends on the material of the object's surface and specifies how much light is reflected)

Notes: Obviously, there are many different ways to describe this correctly. Informal descriptions (such as the one for the form factor given above) are ok.

Problem 7: Shadows

[2 pt] What is a stencil buffer and what kind of operations does it support?

The stencil buffer is a buffer with a one-to-one correspondence of pixels in the frame buffer.

Each entry is a counter and the following operations are supported:

→ resetting, incrementing, decrementing the counters

→ idem, but conditionally, depending on a test against the z-buffer

→ conditional drawing in the frame buffer

Note: There were no tutorials on shadows, but this problem is identical to one given in the final exam 2006/2007.