Lecture 13

review of Exercises 2-11

Exercises 2   Q2
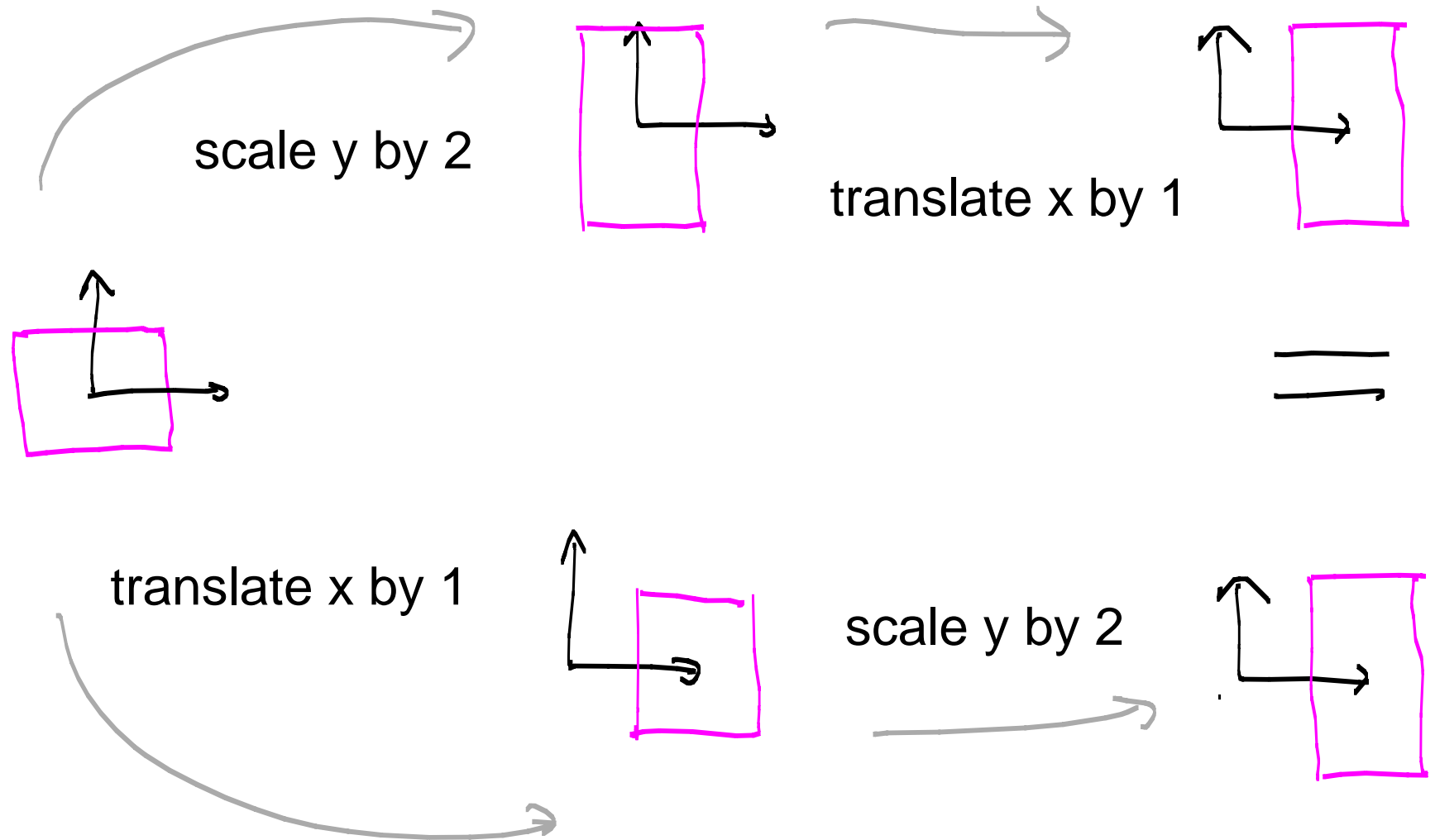
2. We say that two matrices **A** and **B** commute if **A** **B** = **B** **A**. Matrices do not commute in general, but certain types of matrices do. Which of the following types of matrices commute? Assume matrices are 4×4.

- scaling matrices $\mathbf{S}_1\mathbf{S}_2$
- translation matrices $\mathbf{T}_1\mathbf{T}_2$

  } yes

- rotation matrices, $\mathbf{R}_1\mathbf{R}_2$
- rotation and translation, **RT**
- scaling and translation, **ST**
- scaling and rotation, **SR**

  } no

scale y by 2

translate y by 1

translate y by 1

scale y by 2

≠
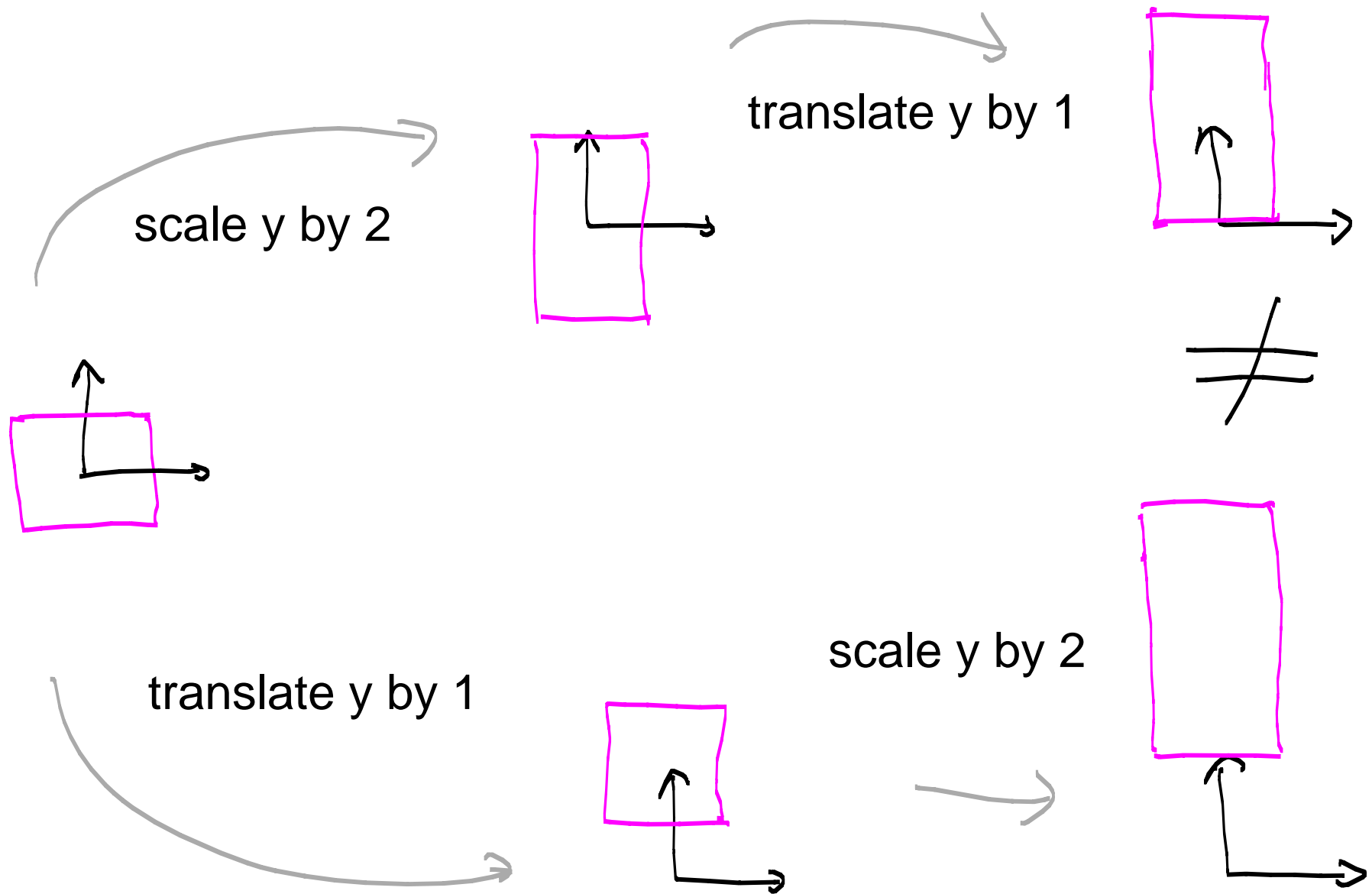
But note... scaling and translation sometimes do commute.

scale y by 2

translate x by 1

translate x by 1

scale y by 2

1. Write the following as a sequence of 4×4 matrices. Scale the scene by a factor 3 in the y direction, while leaving the position of the point $(x, y, z) = (4, -1, 1)$ unchanged.
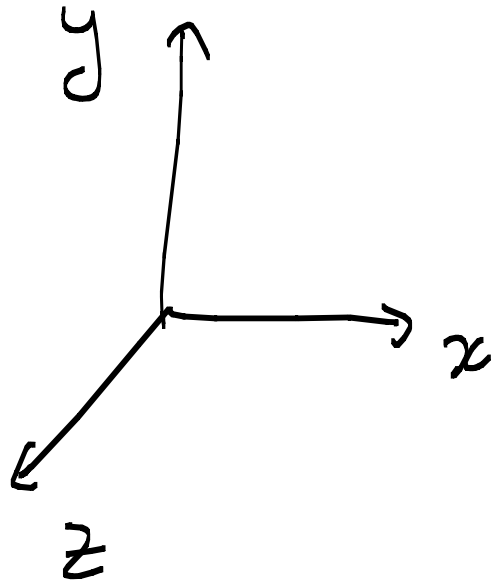
$(4, -1, 1)$

solution 1:

$$\begin{bmatrix} 1 & 0 & 0 & 4 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -4 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

solution 2:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Variation on the problem: rotate the scene
about the point (4, 1, -1) and about some axis.

$$\begin{bmatrix} 1 & 0 & 0 & 4 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -4 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

R

Exercises 3   Q5

5. One can define an ellipsoid surface in $\Re^3$ by scaling, rotating, and translating a unit sphere. Define an ellipsoid whose:

- center is $(8, 1, 3)$ in world coordinates,
- x, y, z axes are of length $1, 1, 4$, respectively.
- z axis is in direction $(1, 1, 0)$ in world coordinates,

For your answer, it is sufficient that you specify suitable matrices $\mathbf{S}, \mathbf{R}, \mathbf{T}$, and then use them to define a 4×4 quadric matrix $\mathbf{Q}$.
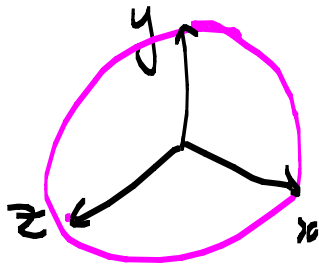
unit sphere $\vec{x}$

ellipsoid $\vec{x}'$

$$\vec{x}' = T R S \vec{x}$$

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \mathbf{T}\,\mathbf{R}\,\mathbf{S} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 8 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ 0 & -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

We have the equation of a unit sphere.
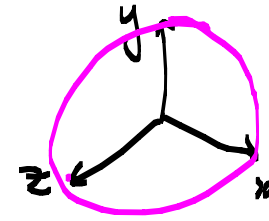
Unit sphere $\vec{x}$

. For a unit sphere $x^2 + y^2 + z^2 = 1$,

$$
\begin{bmatrix} x & y & z & 1 \end{bmatrix}
\begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & -1
\end{bmatrix}
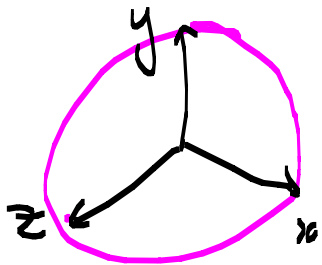\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = 0
$$

and so

$$
Q = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
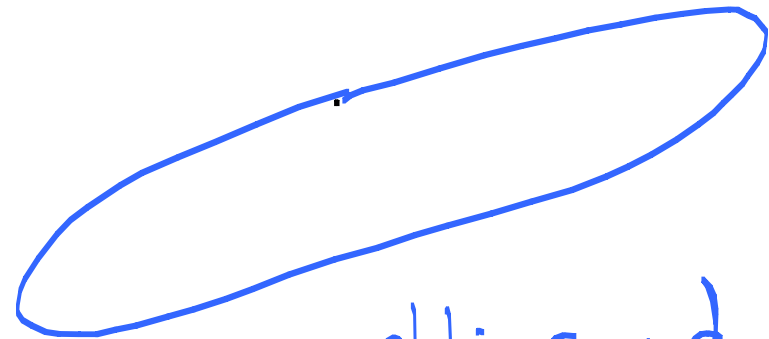0 & 0 & 1 & 0 \\
0 & 0 & 0 & -1
\end{bmatrix}.
$$

$$
\vec{x}^{\,T} \, Q \, \vec{x} = 0
$$

$$(TRS)^{-1} \vec{x}' = \vec{x}$$

substitute

$$\vec{x}^T Q \vec{x} = 0$$

unit sphere $\vec{x}$

ellipsoid $\vec{x}''$

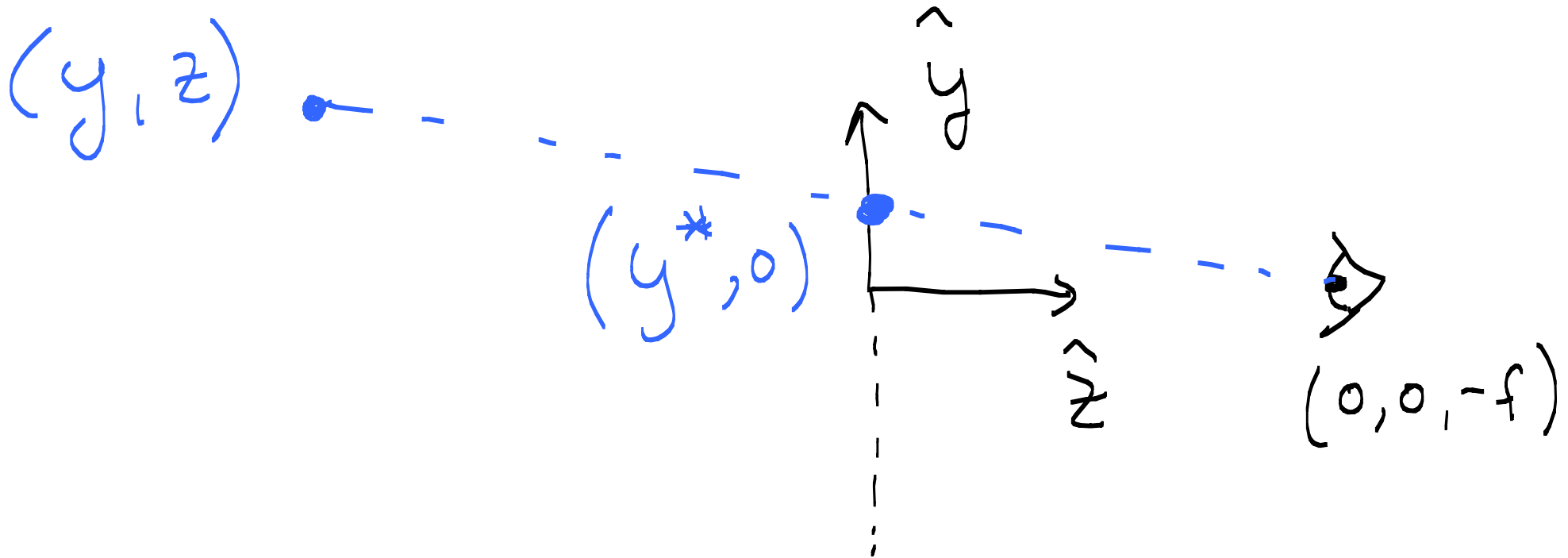$$\vec{x}'^{T} \underbrace{(TRS)^{-T} Q (TRS)^{-1}}_{Q'} \vec{x}' = 0$$
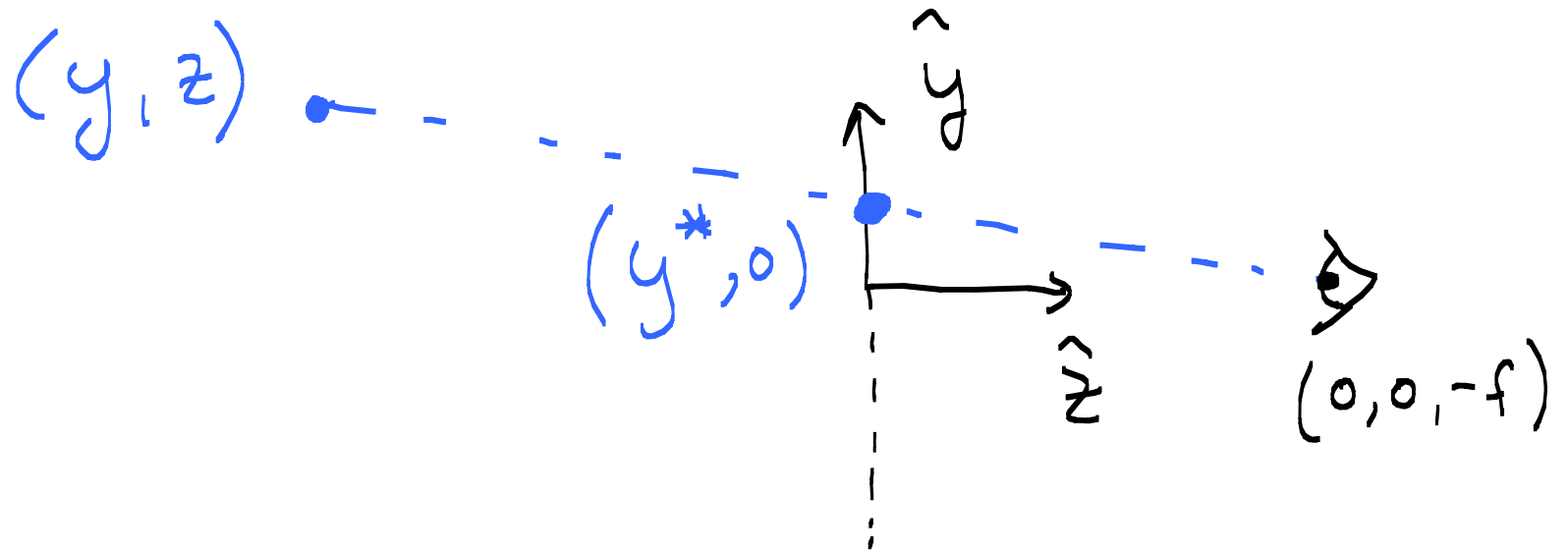
Q'
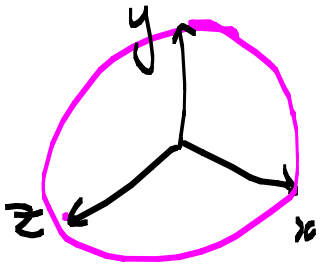
unit sphere $\vec{x}$
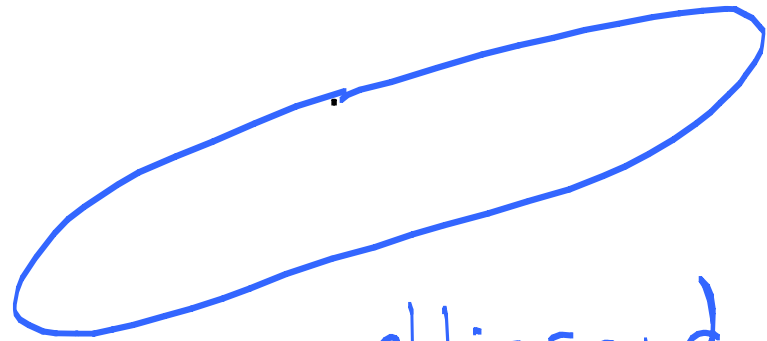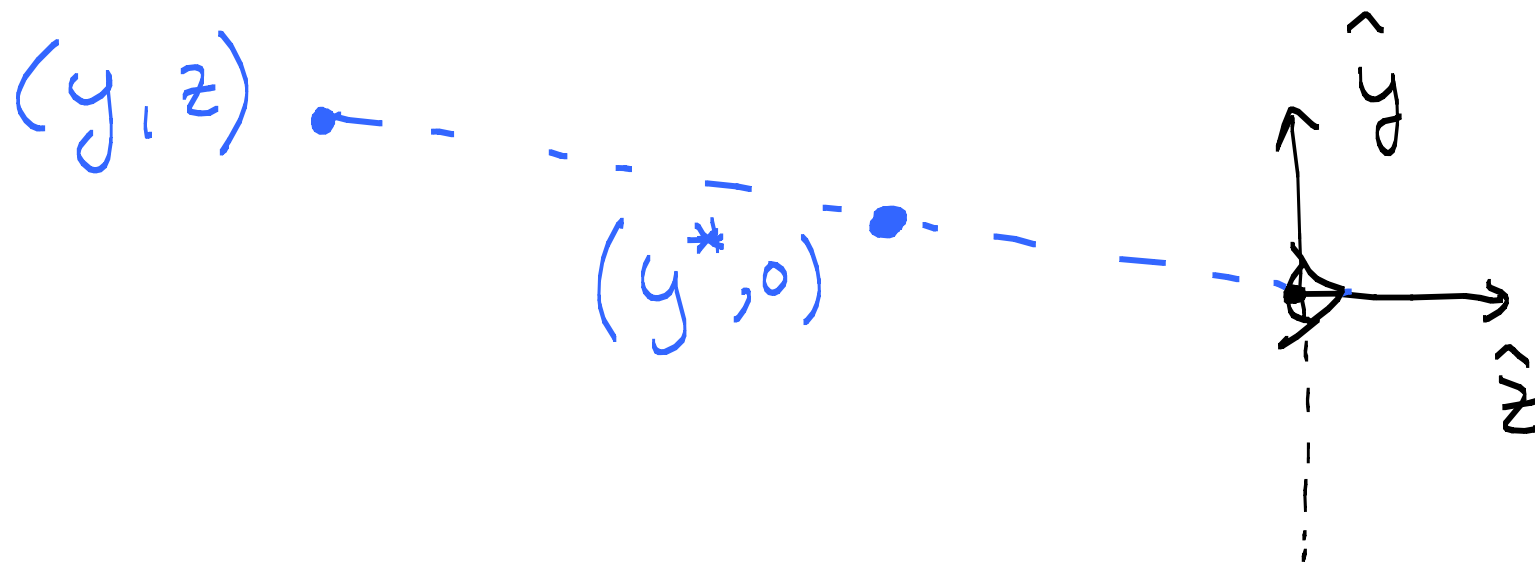
ellipsoid $\vec{x}'$

Exercises 4 Q2 and Q3

2. Suppose the eye is at $(0, 0, -f)$ where $f < 0$ and the projection plane is $z = 0$. Also suppose the camera's y and z axes are the same as those axes for world coordinates. What is the projection matrix?

3. For the previous question, what happens when $f \to -\infty$?

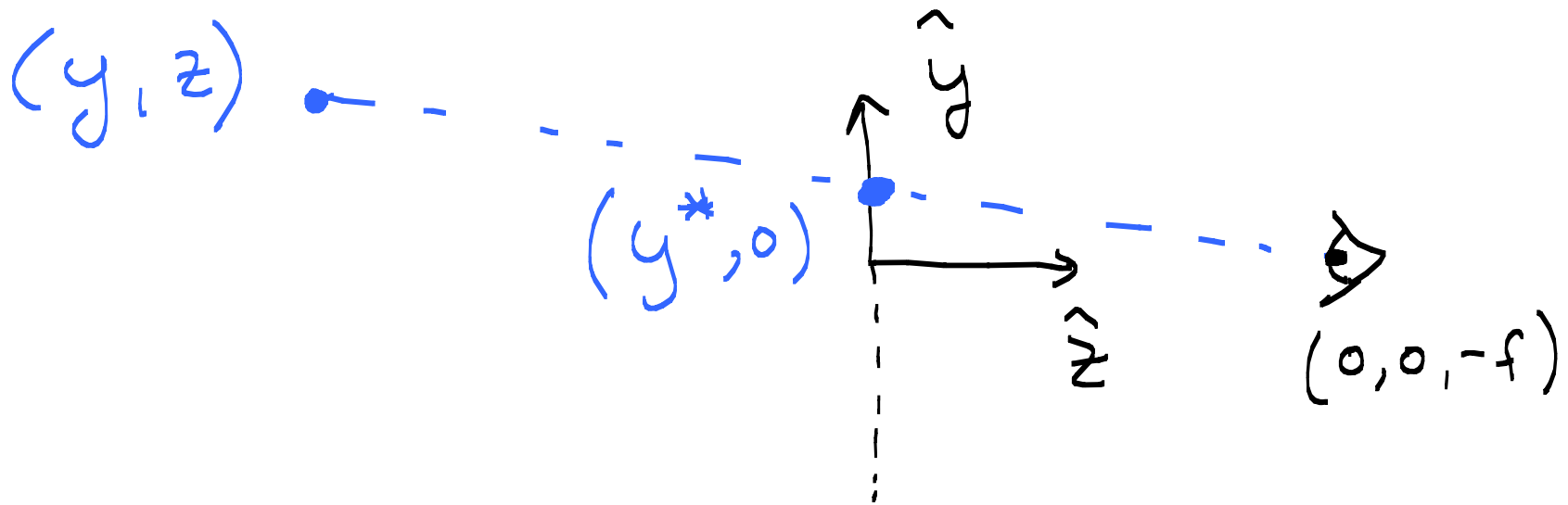Translation by f gives the situation presented in class.
Note f < 0.

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -f \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & f & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & f \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & f \end{bmatrix},$$
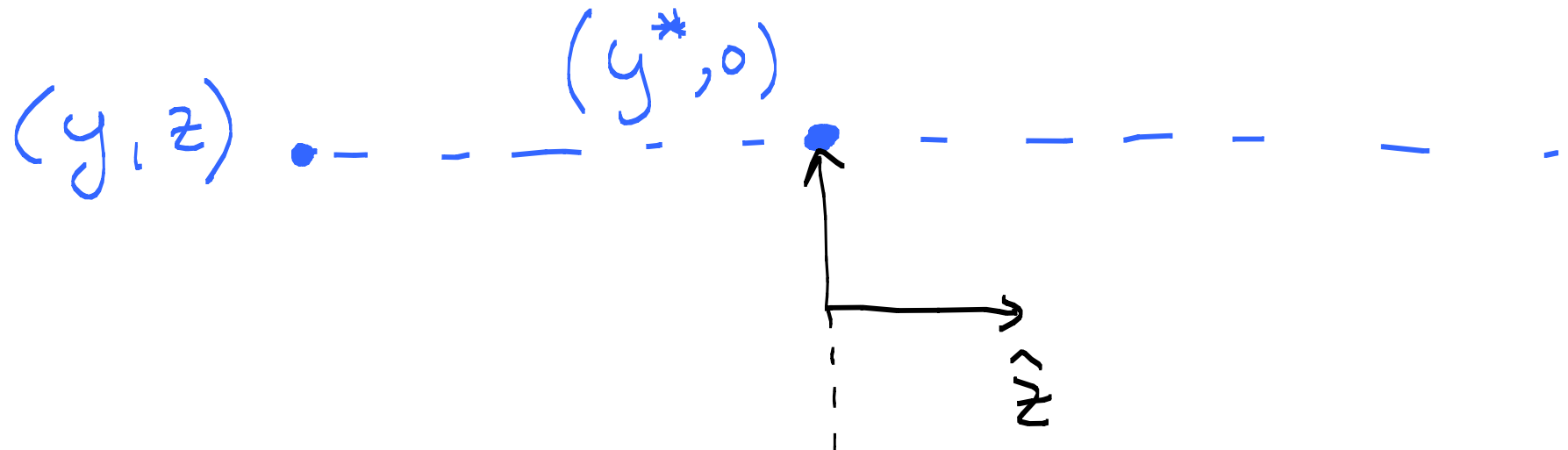
translate by -f

translate by f

projection given in lecture
(to origin)

(y, z)

$(y^*, 0)$

$\hat{y}$

$\hat{z}$

$(0, 0, -f)$

Q: What happens if f goes to -infinity ?

$(y^*, 0)$

(y, z)

$\hat{z}$

A: orthographic projection

$$\begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & f \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{f} & 1 \end{bmatrix}$$
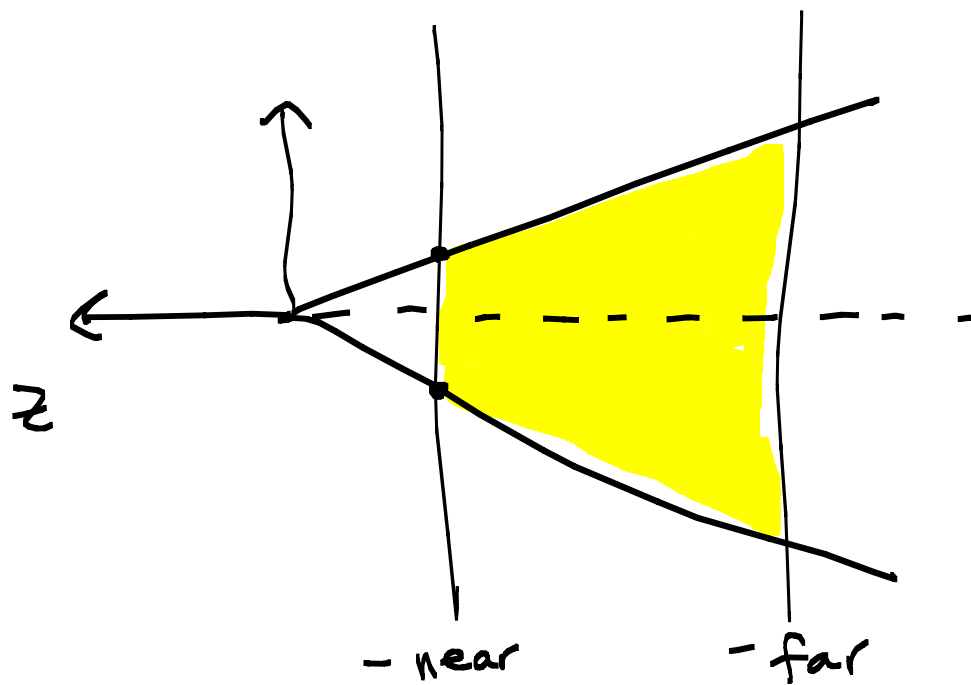
When f goes to -infinity,   1/f goes to 0.

Exercises 5 Q2

2. Give the 4×4 projective transformation matrix that is defined by

$$\text{gluPerspective(theta, aspect, near, far)}.$$

Rather than just plugging in the values given in the lecture, think about the special constraints on gluPerspective, as opposed to glFrustum.

z

$\uparrow$

$\leftarrow$

- near   - far

$$\mathbf{M}_{projective} = \begin{bmatrix} \text{near} & 0 & 0 & 0 \\ 0 & \text{near} & 0 & 0 \\ 0 & 0 & \text{near} + \text{far} & \text{near} * \text{far} \\ 0 & 0 & -1 & 0 \end{bmatrix}.$$

z

$\uparrow$

$\leftarrow$
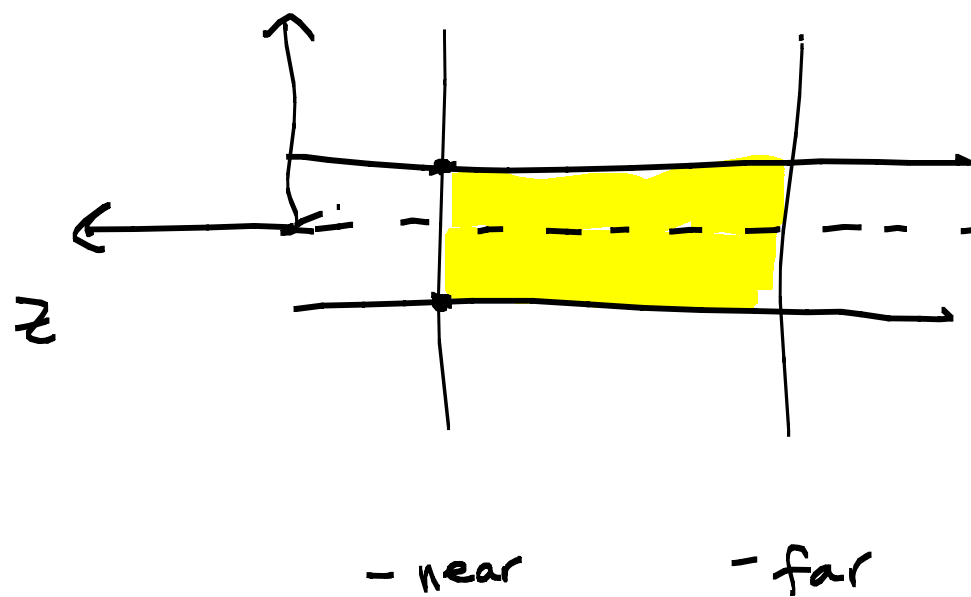
- near   - far

glFrustum(left, right, bottom, top, near, far)

For gluPerspective:

left = - right
bottom = - top

(right, top)

(left, bottom)
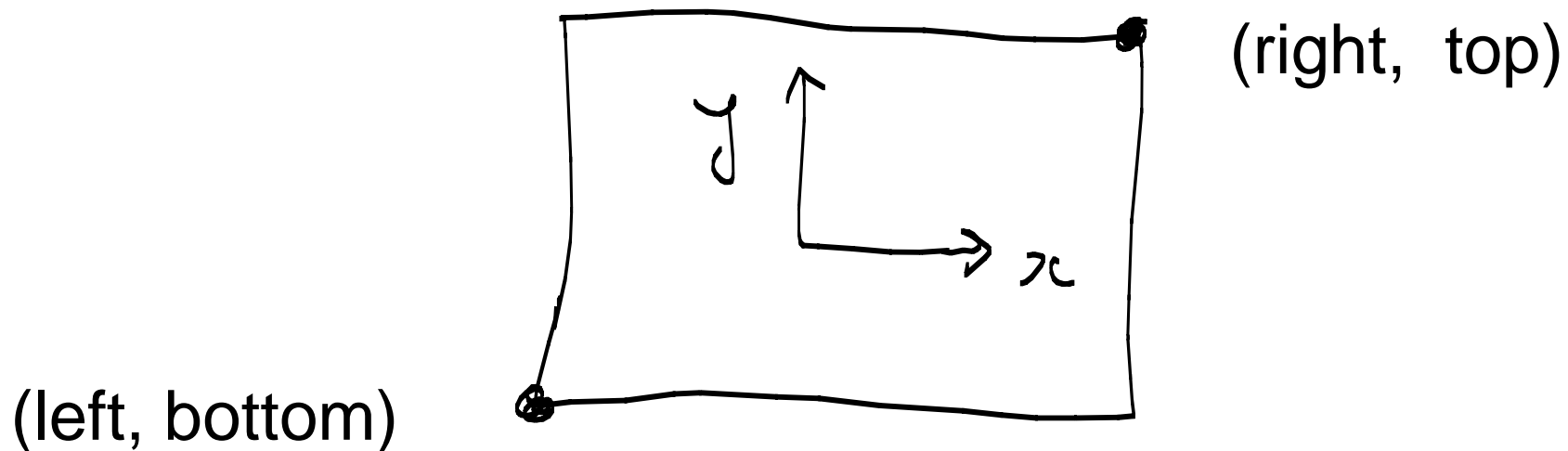
These are defined in the plane: z = -near

- near   - far

z
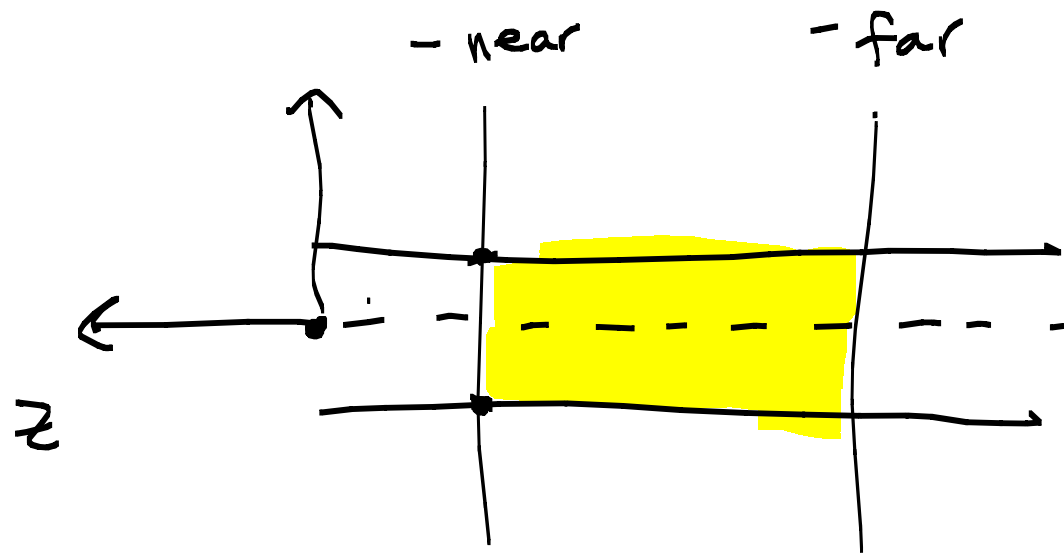
$M_{normalize} = T_z(1) \, S \, T_z(near)$

-1   1
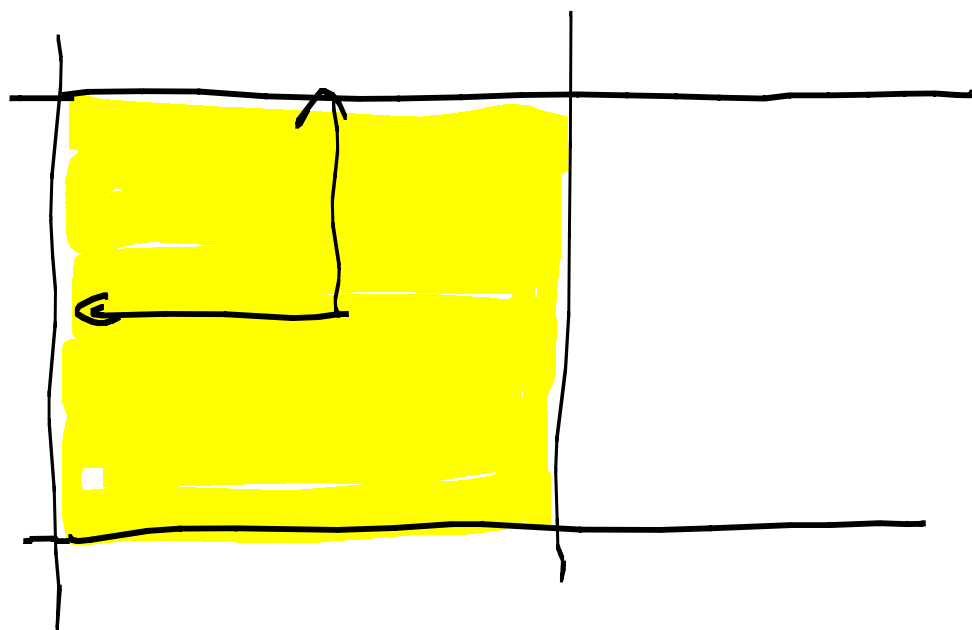
$$\mathbf{M}_{normalize} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{2}{\text{near}*\mathbf{aspect}*2*\tan(\frac{\theta}{2})} & 0 & 0 & 0 \\ 0 & \frac{2}{\text{near}*2*\tan(\frac{\theta}{2})} & 0 & 0 \\ 0 & 0 & \frac{2}{\text{near}-\text{far}} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \text{near} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$
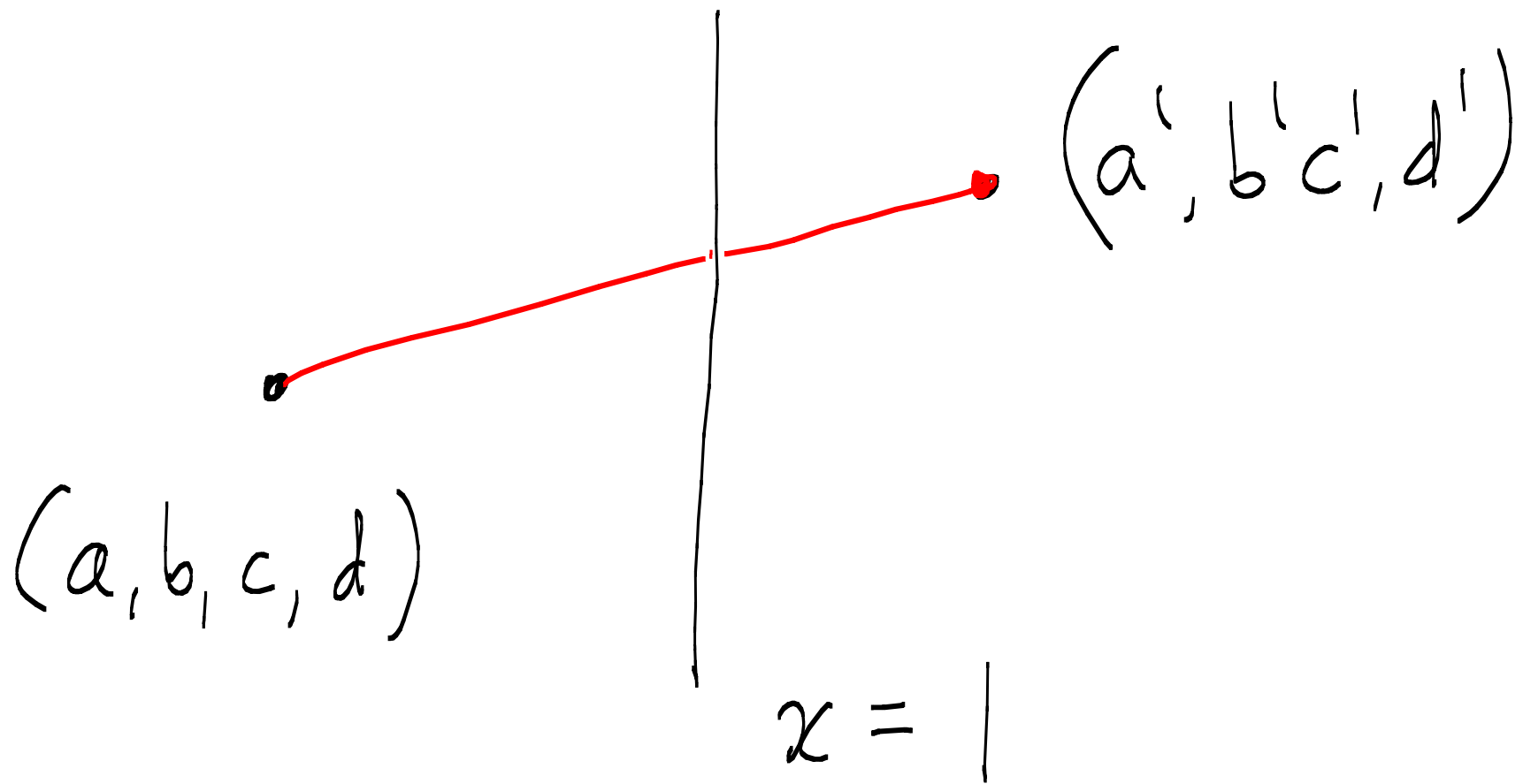
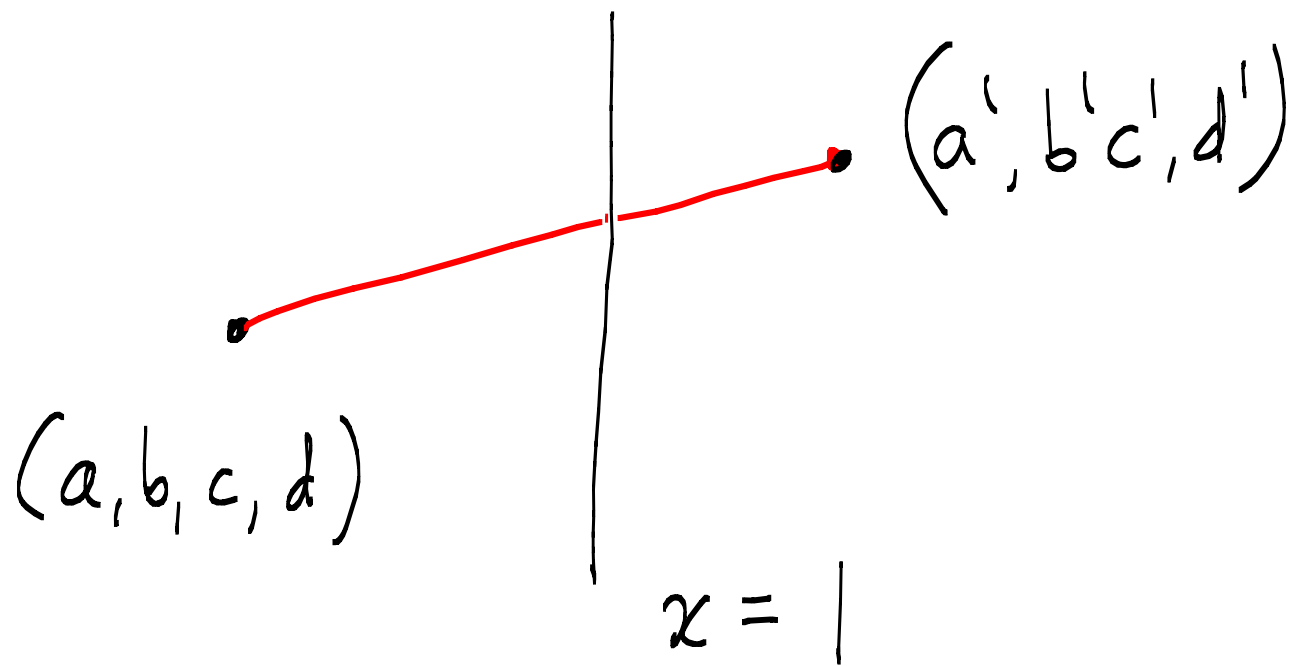$$T_z(-1) \qquad\qquad S \qquad\qquad T_z(near)$$

Exercises 6   Q2

2. Suppose you have a line segment joining two points (a, b, c, d) and (a', b', c', d') in clip coordinates and you wish to clip the line segment to a boundary plane, say $x = 1$ of the normalized view volume. Show how you can do this directly in clip coordinates. You do *not* need to do perspective division to find the intersection.

$(a', b'c', d')$

(a, b, c, d)

$x = 1$

$(a', b'c', d')$

(a, b, c, d)

$x = 1$

$$(a, b, c, d)\, t + (a', b', c', d')\,(1-t)$$

$(a',b'c',d')$

(a,b,c,d)

$x = 1$
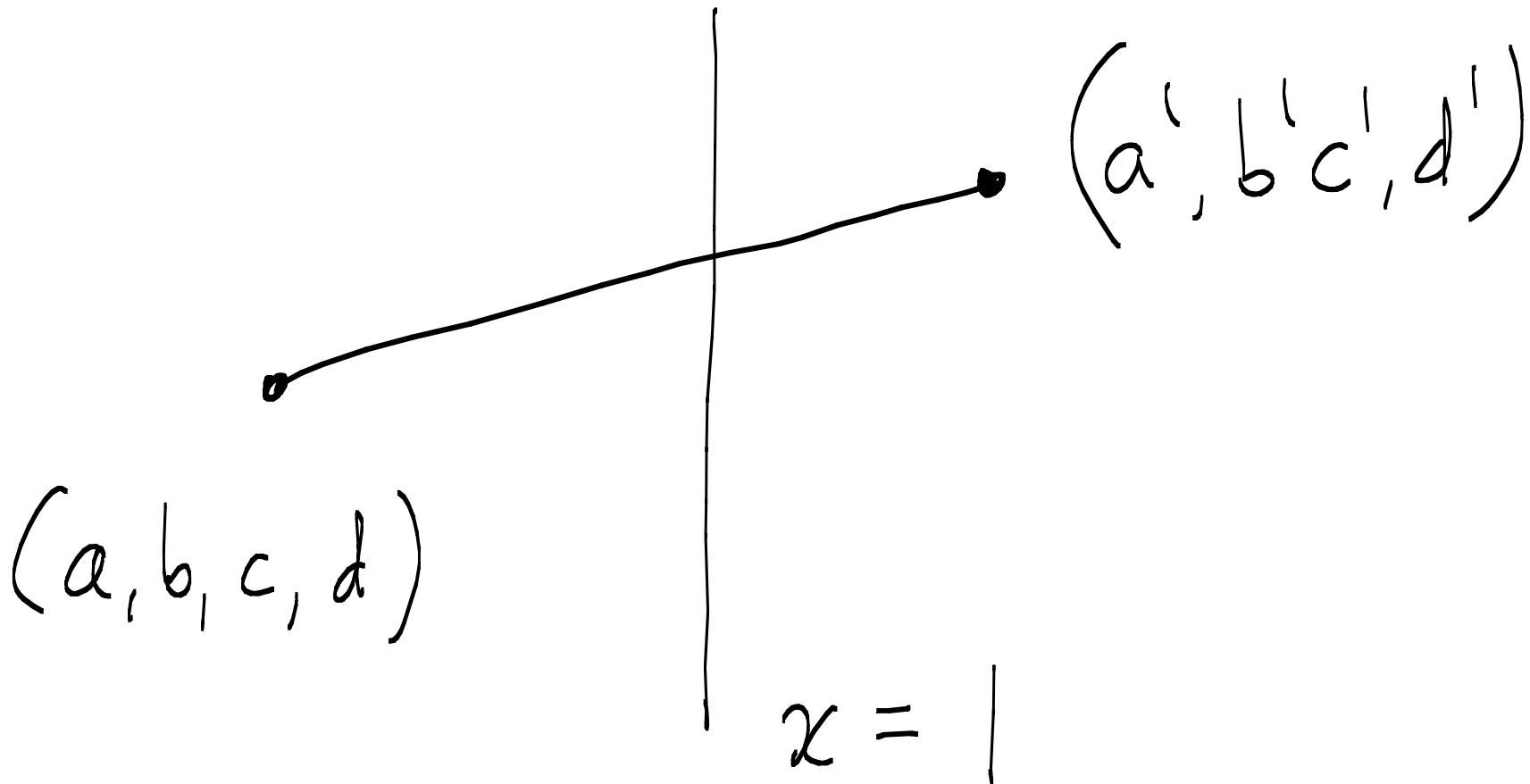
$(a, *, *, d) \, t \, + \, (a', *, *, d')(1-t)$

$$= \quad (w, *, *, w)$$

Two equations, two unknowns.
Solve for t.

Solve for t.

$$t = \frac{a' - d'}{(a - a') - (d - d')} \quad \Big\} \quad \begin{array}{l} \text{need to} \\ \text{do this} \\ \text{for} \end{array}$$

$$x = \pm 1, y = \pm 1,$$
$$z = \pm 1$$

i.e. six divisions

clip
coordinates

fragments

vertices →

vertex
processing

"primitive
assembly"
and clipping

rasterization

fragment
processing

pixels
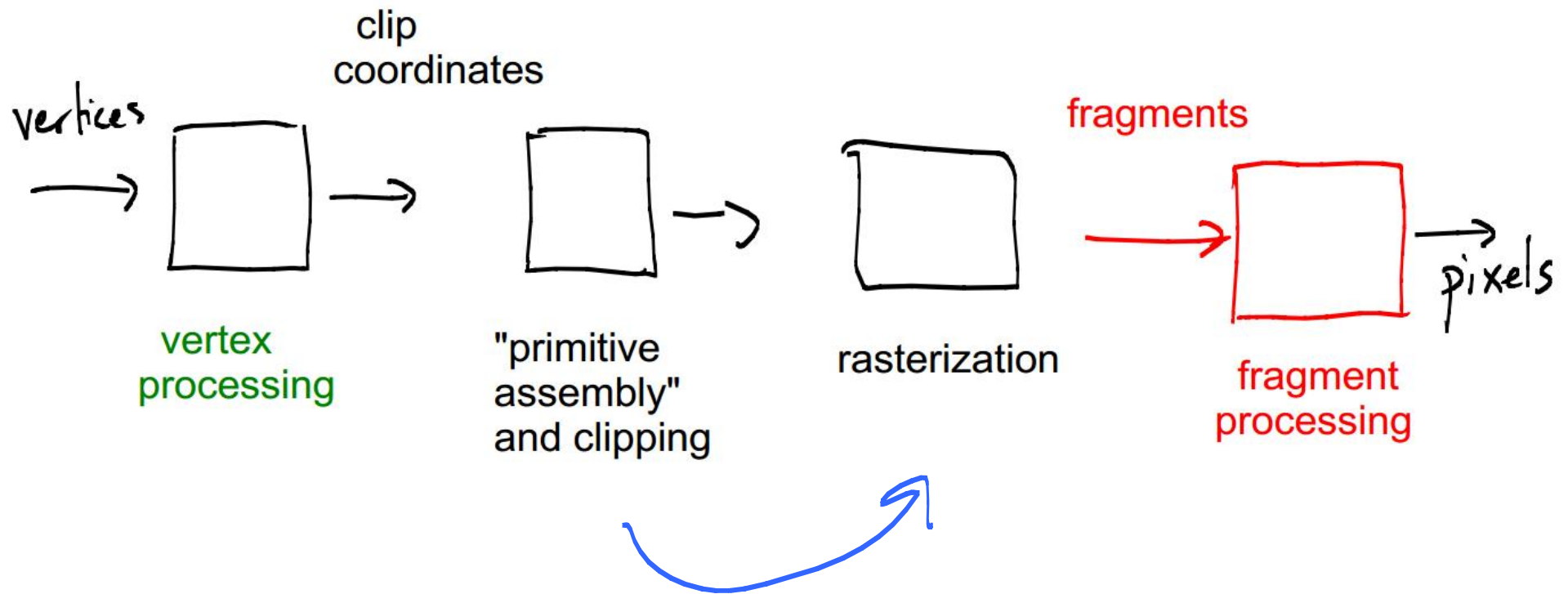
In the pipeline lecture (7),   I did not mention "perspective division"
(clip coordinates to normalized device coordinates) and the
mapping from NDC to window coordinates.   Be aware that this
mapping does need to be computed before rasterization.

Perspective division also requires six divisions.

$$\left(\frac{a}{d}, \frac{b}{d}, \frac{c}{d}\right) \qquad \left(\frac{a'}{d'}, \frac{b'}{d'}, \frac{c'}{d'}\right)$$

Exercises 6   Q4

4. How do you do line clipping if one of the vertices has $w < 0$? Are any special considerations required or can you just go ahead and use the method of Q2 above?

Hint: there are issues. What are they? Look at the ABCD...JK figure from lecture 5 and think about what happens if you clip.
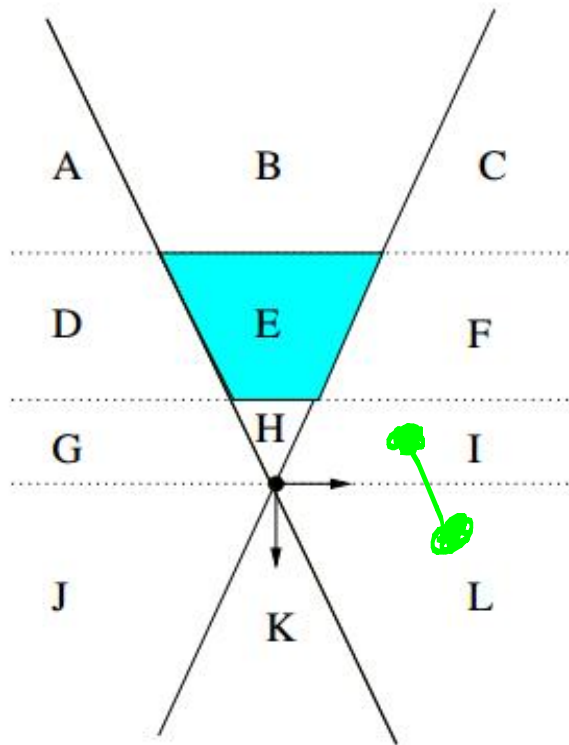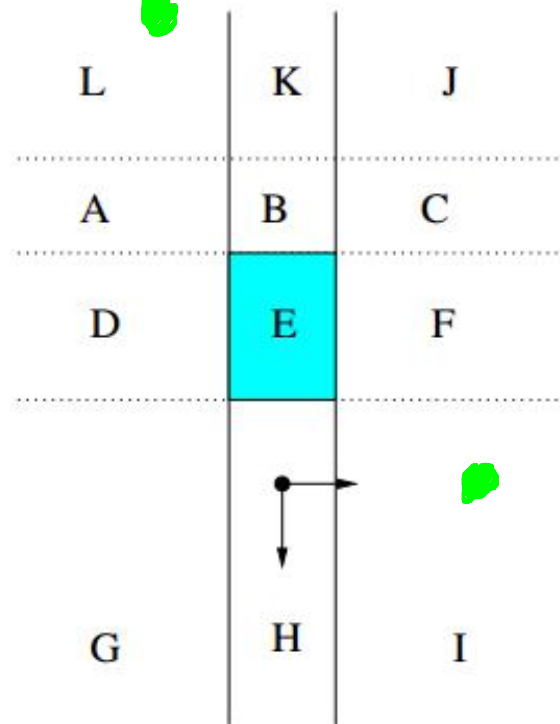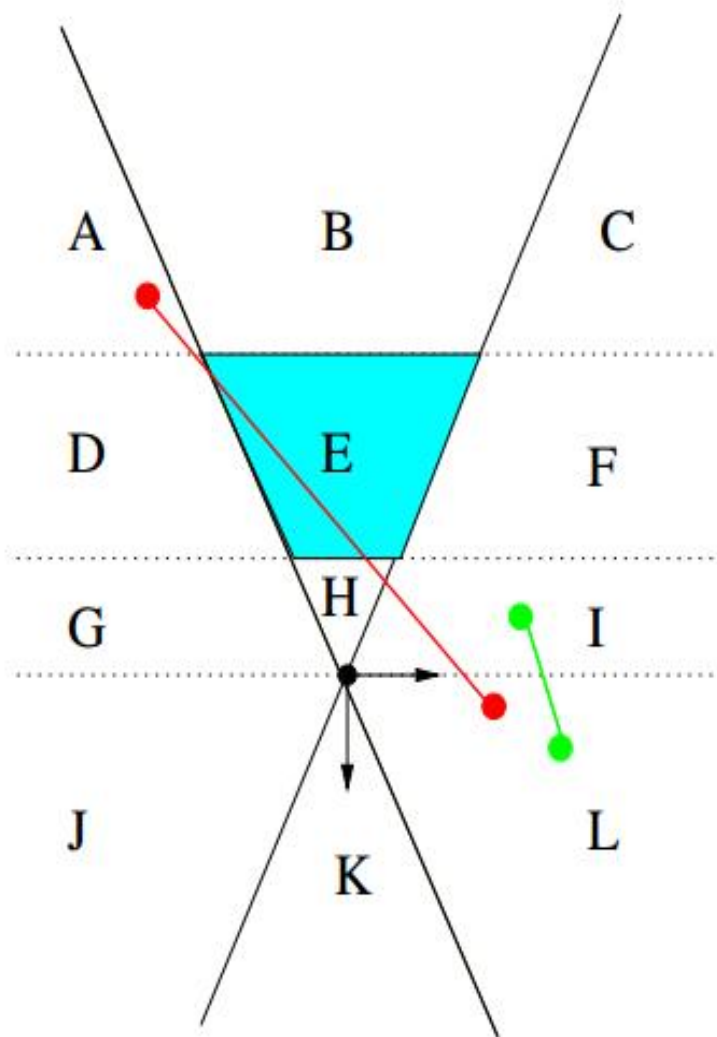
camera coordinates

$z = f0 + f1$

$z = f1$

$z = f0$

$z = 0$
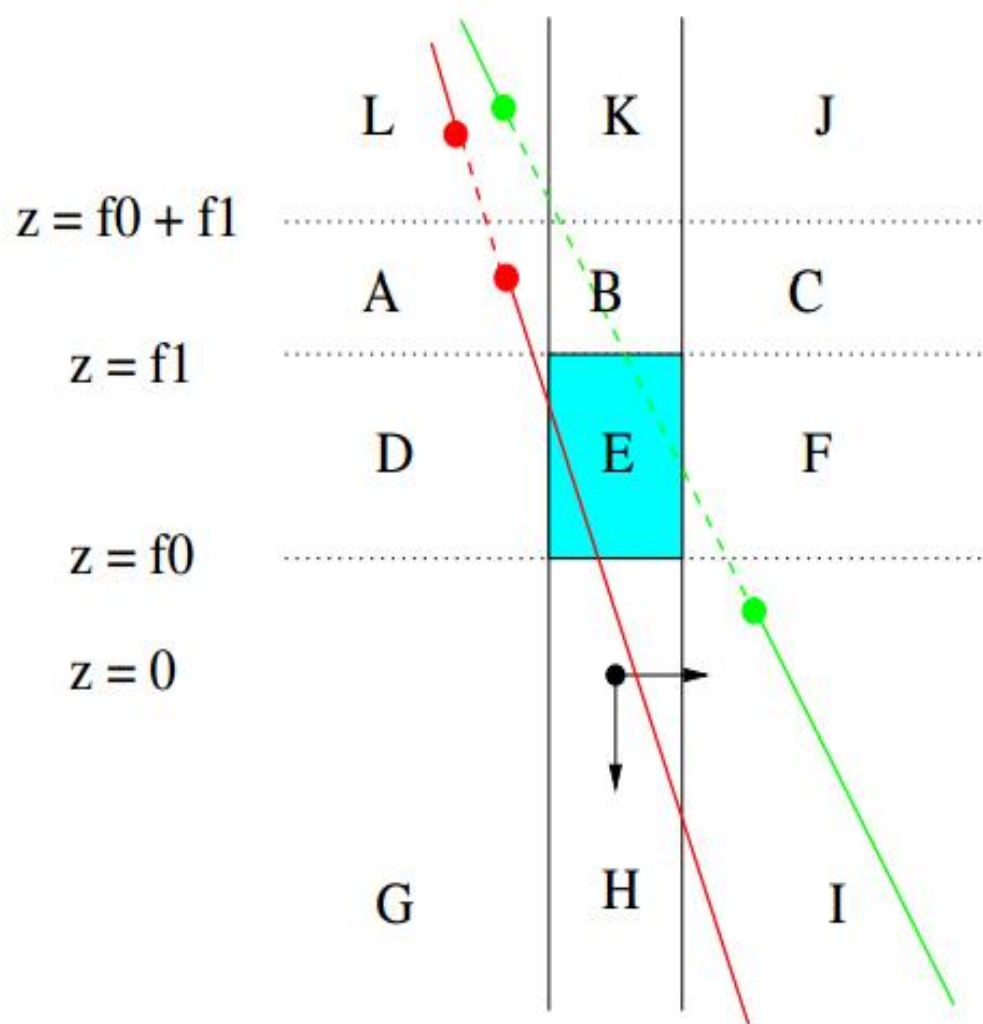
(non−normalized) projective coordinates

Exercises 7 - Q2

2. The depth buffer partitions the near to far range of depth into intervals using fixed precision (typically 24 bits, hence 2^{24} intervals). The mapping from continuous to discrete depth happens late in the pipeline, in particular, it maps from the normalized device depth coordinate to a (24 coding) of the range $[0,1]$.

The depth "bins" do *not* correspond to uniform sized depth bins in the original scene space, however. Why not? Are the close bins bigger or smaller than the far bins, when measured in the original scene space ?
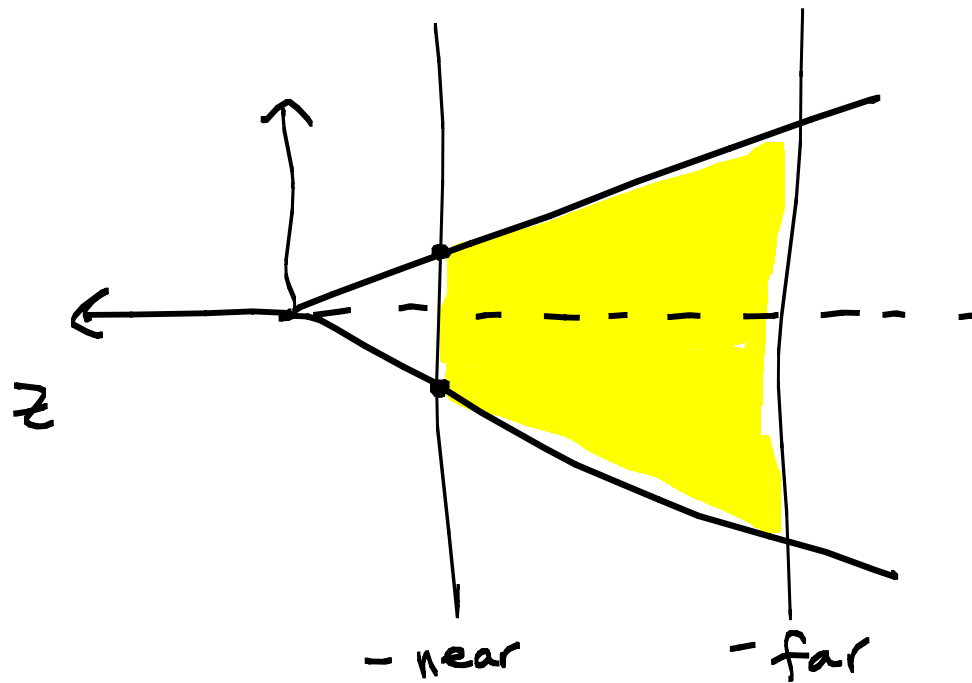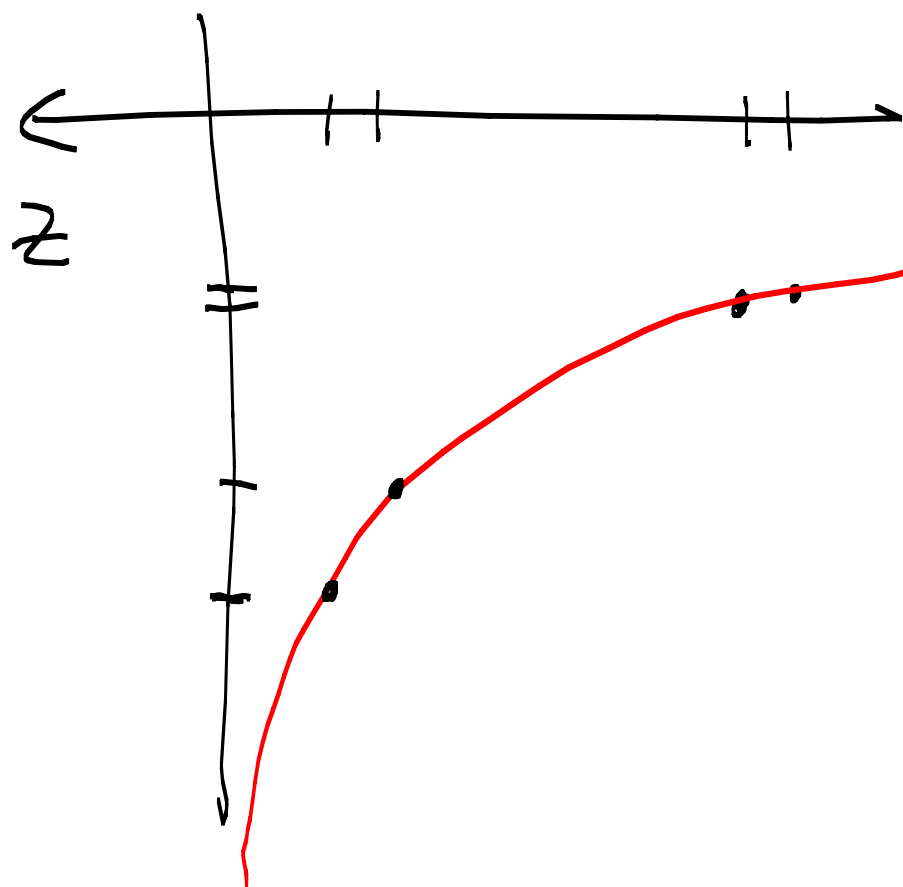
$$M_{projective} = \begin{bmatrix} \text{near} & 0 & 0 & 0 \\ 0 & \text{near} & 0 & 0 \\ 0 & 0 & \text{near} + \text{far} & \text{near} * \text{far} \\ 0 & 0 & -1 & 0 \end{bmatrix}.$$
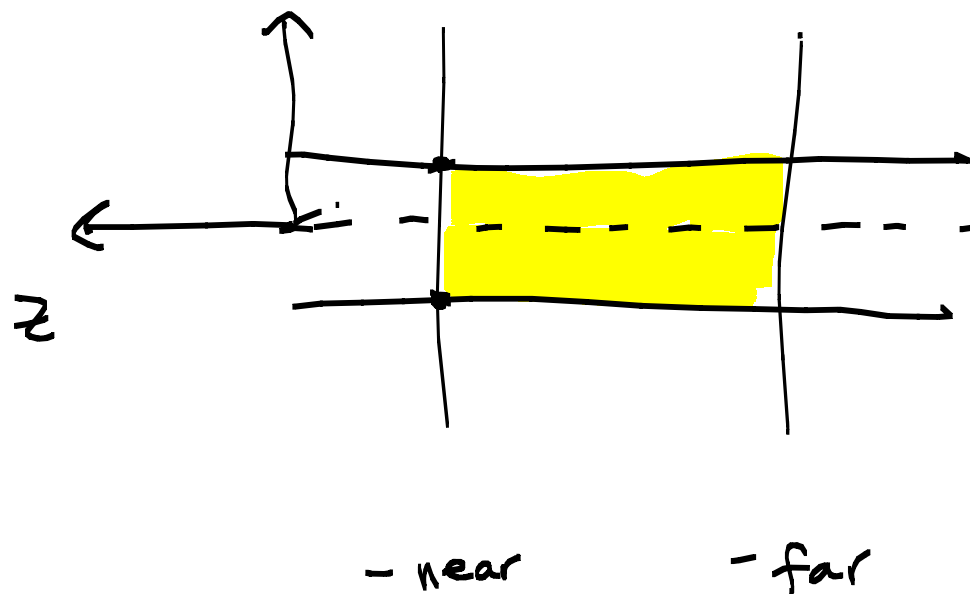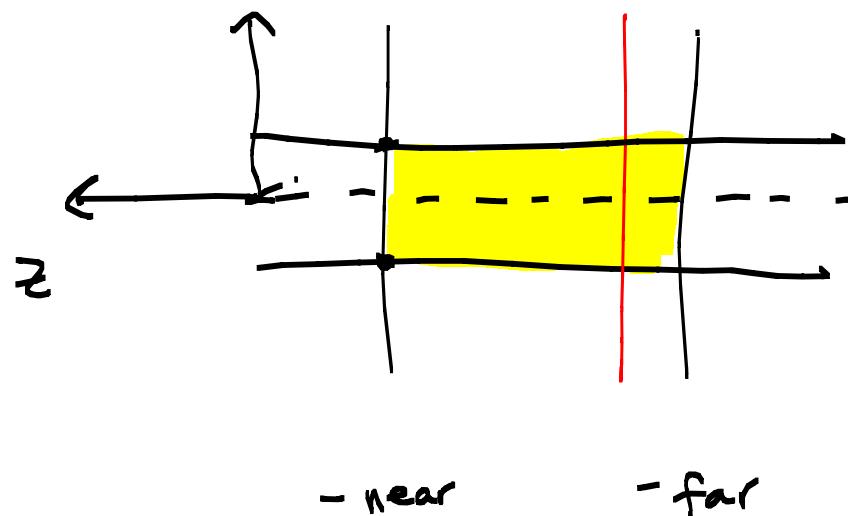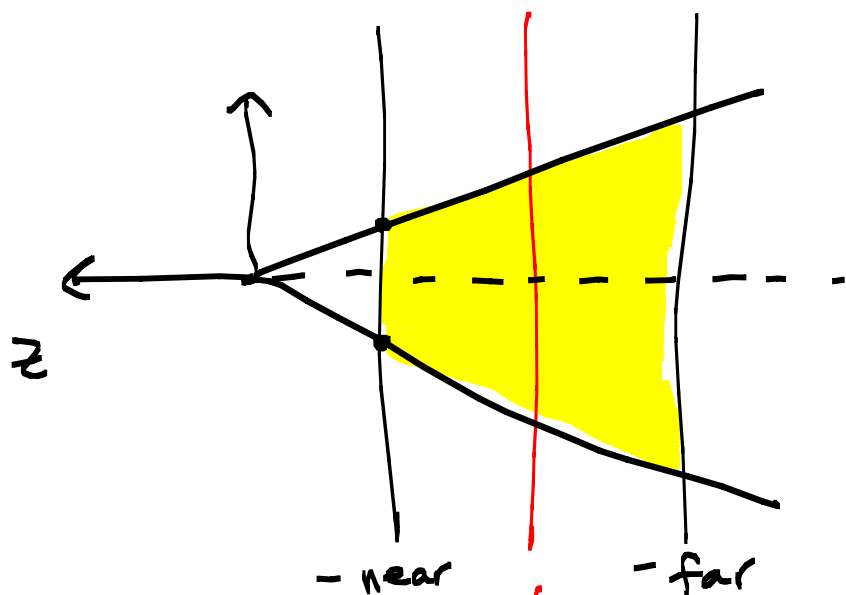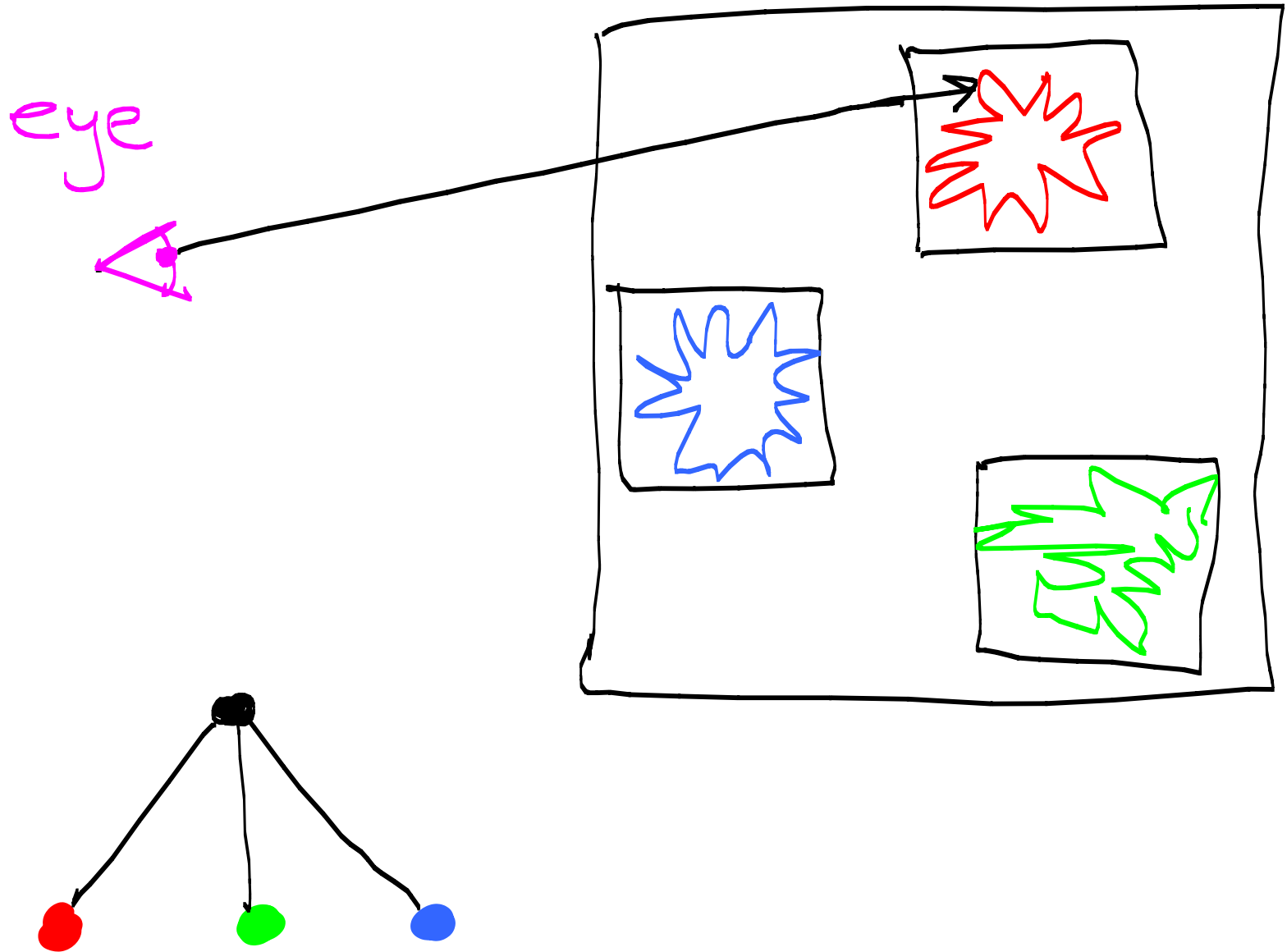
$$z = f_0 + f_1 - \frac{f_0 f_1}{z}$$

Exercises 8 Q6

6. One problem with the hierarchical bounding volume algorithm discussed in class is that there is no attempt to visit the children of a node (volume) in an optimal order. This is problem because you easily expand a subtree that turns out to be entirely hidden.

   An alternative approach is to traverse the tree by only expanding a node (bounding volumes) when the intersection with this node is closer than the intersection of other "candidate" nodes. This can be done by maintaining a separate data structure of nodes – a priority queue of nodes – whose keys are the distances to the node. How would this work? Write out such an algorithm.

eye
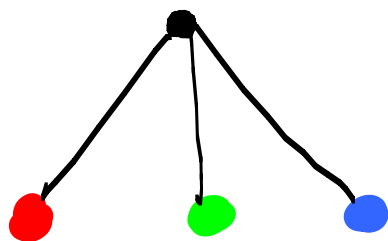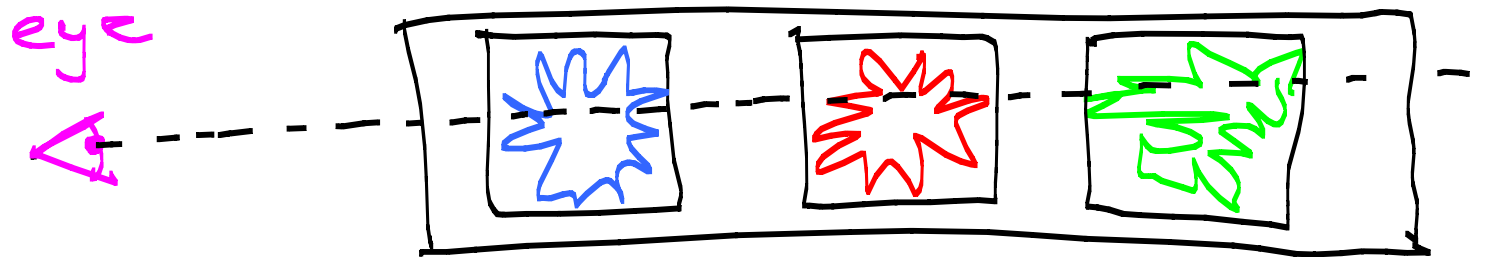
smart HBV algorithm would expand this one first

eye

priority queue

min $\longrightarrow$ 🟦 $\rightarrow$ 🔴 $\rightarrow$ 🟢
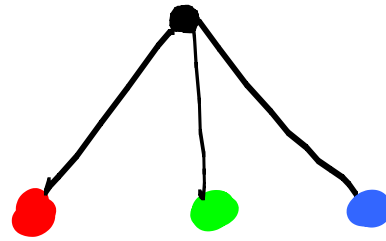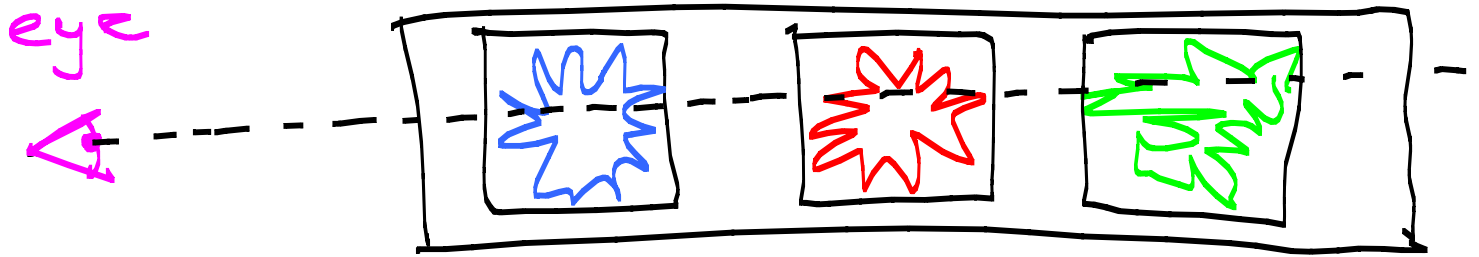
```
p = NULL
t = infinity

if ray hits root's BV{
    insert root into heap
}
while (heap is not empty and distance to heap.peek < t) {
    curNode = remove min from heap
    if curNode is a leaf{
        intersect ray with leaf's surface
        if (t_intersect < t)
            t = t_intersect
            p = surface
    }
    else
        for each child of curNode{
            intersect ray with child's BV
            if t_intersect < t
                insert child into heap
        }
}
```

General idea: at any time, there is a queue of (candidate) bounding volumes still to be expanded.
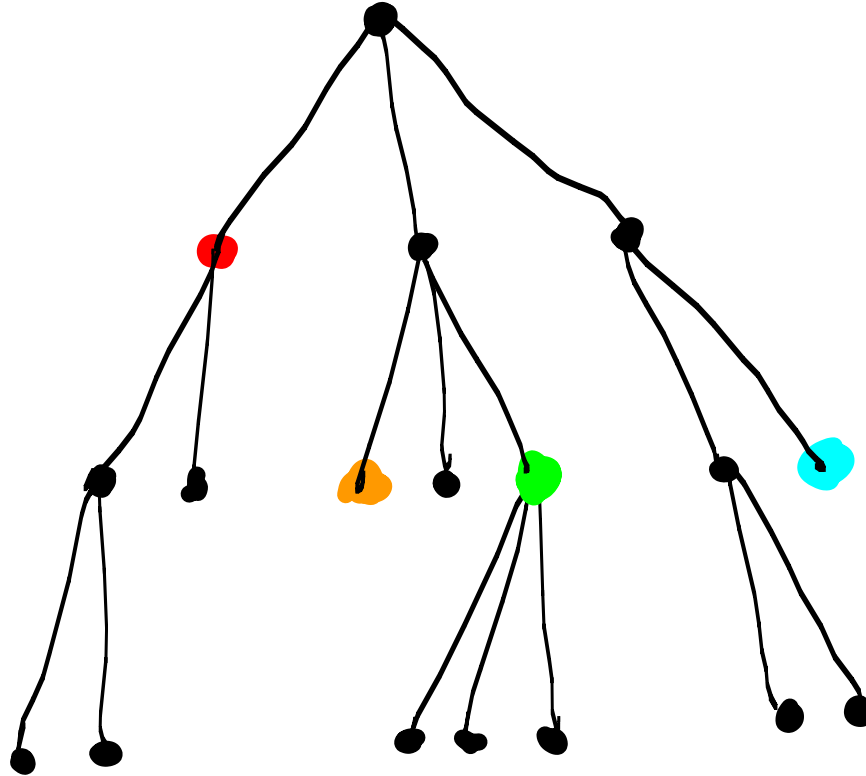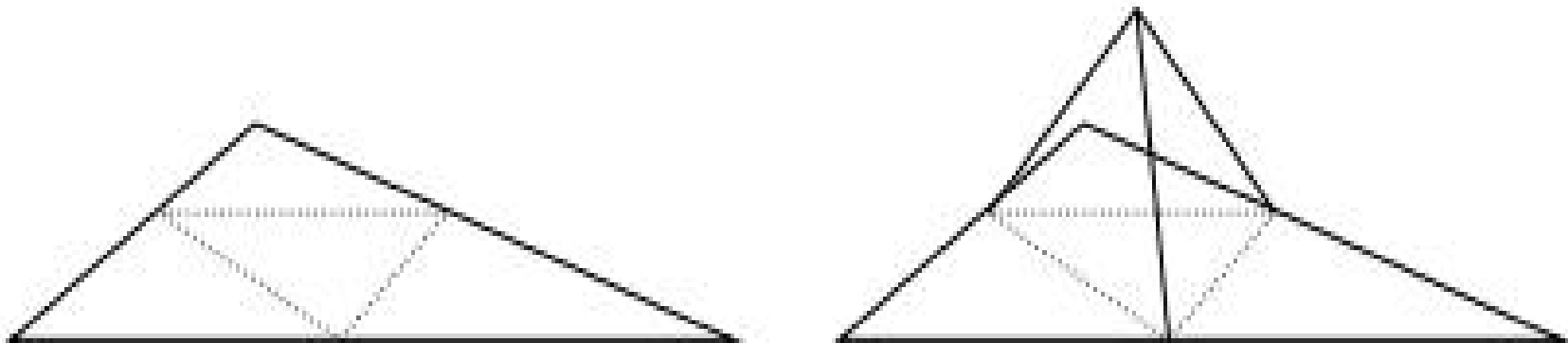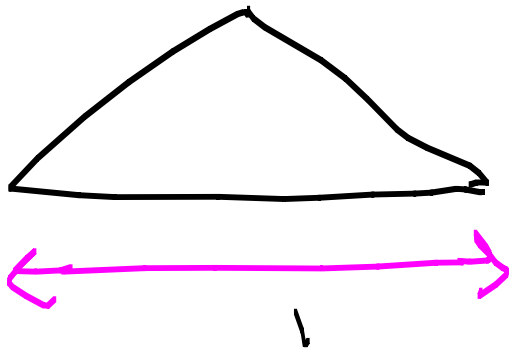

priority queue   min ⟶ 🟠 ⟶ 🟢 ⟶ 🔴 ⟶ 🔵

2. Consider a Koch-like surface defined by the recursive production rule illustrated in the figure. A equilateral triangle is partitioned into 4 subtriangles, and the central one is replaced by a 3 sided pyramid. Repeating this recursive procedure infinitely many times, what is the dimension of the resulting surface ?
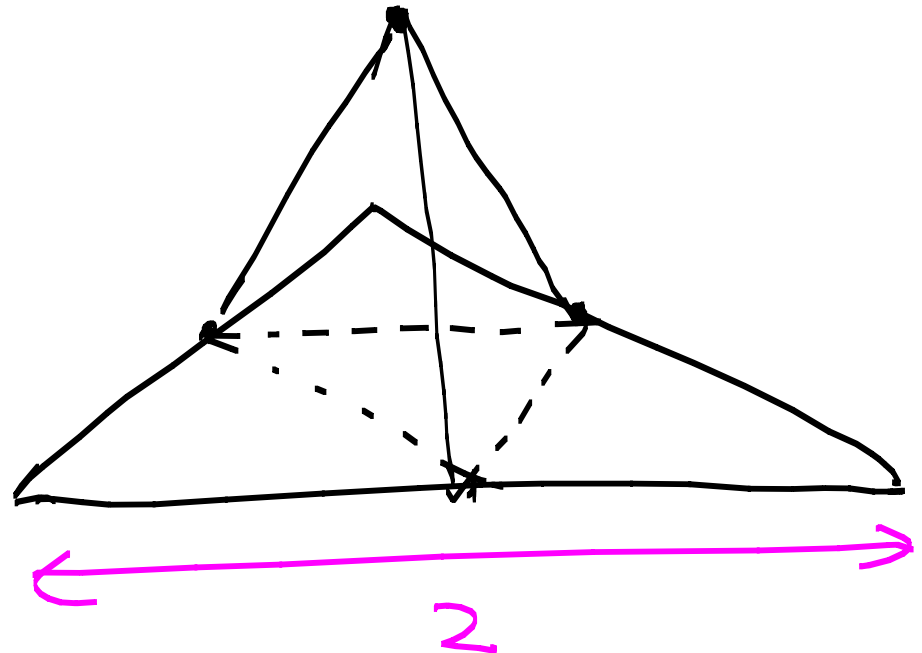
$C = S\text{^}D$

$S = 2 \qquad C = 6$

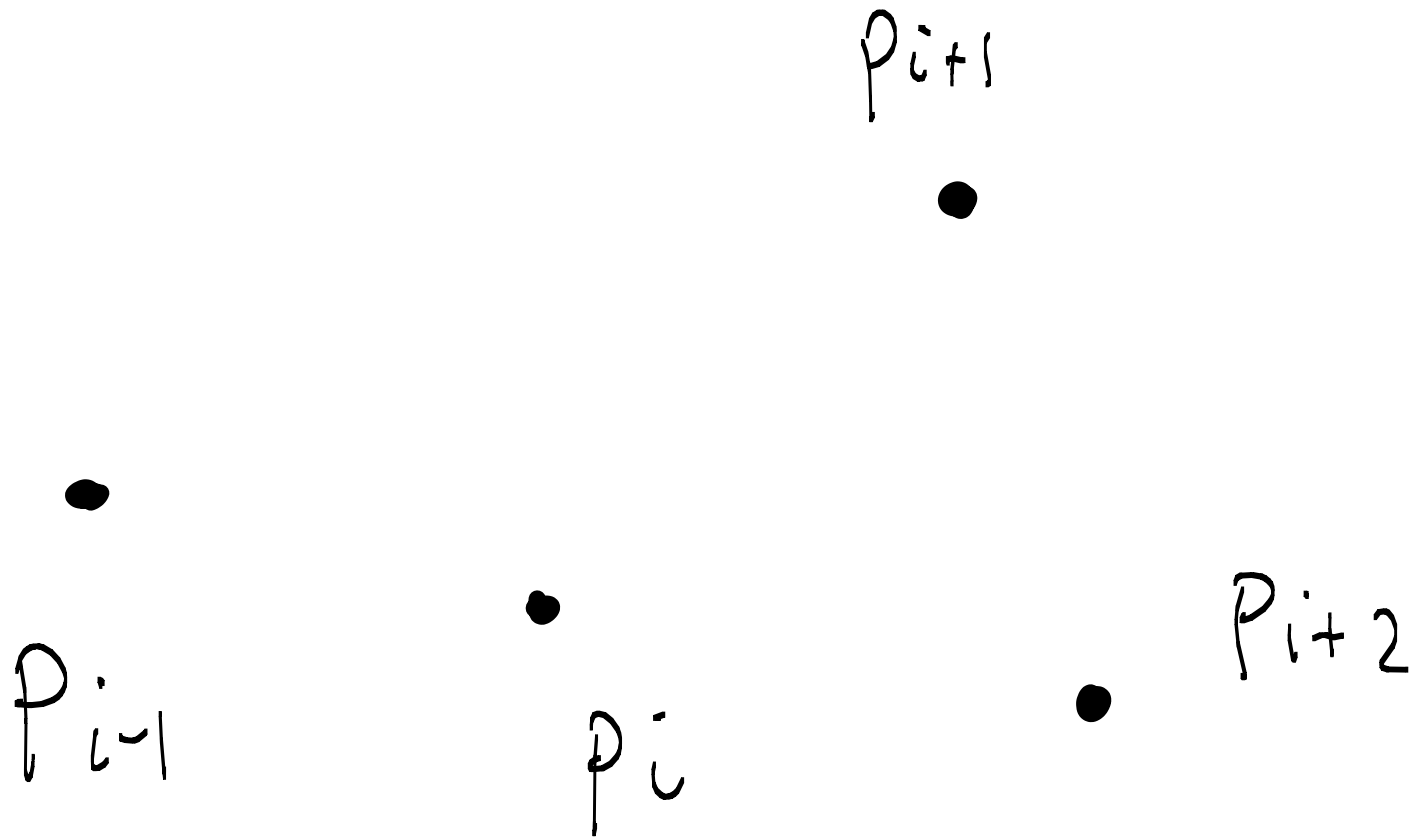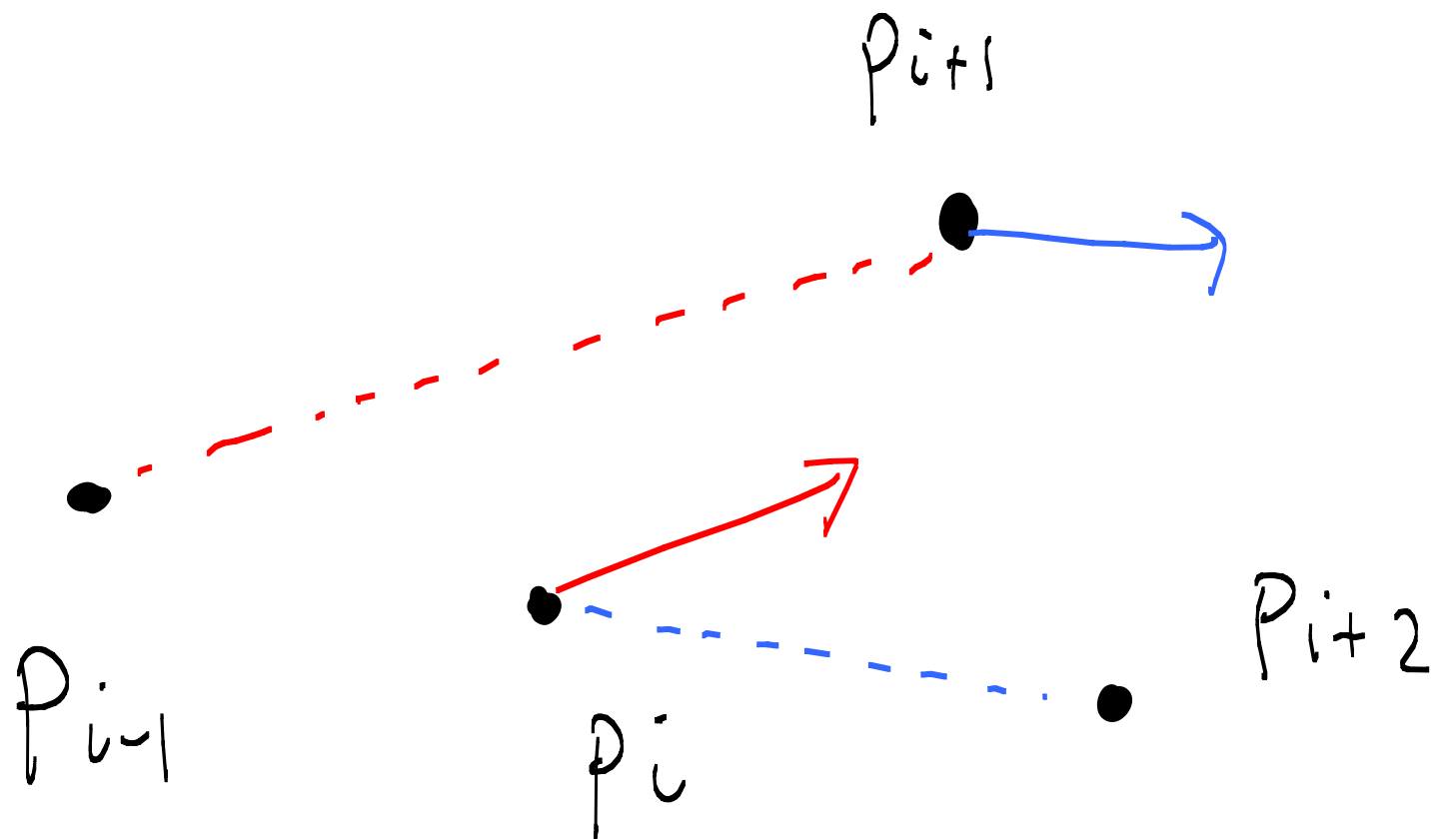$D = \log C / \log S = \log 6 / \log 2. \sim 2.58$

$S = 2$

1. What are geometry and blending matrices for Catmull-Rom spline?

$$\bar{p}(t) = G \, B \begin{bmatrix} t^3 \\ t^2 \\ t \\ 1 \end{bmatrix}$$

Recall scenario for Catmull-Rom spline :

P_{i+1}

P_{i-1}

P_i

P_{i+2}

$$P_i' = \frac{1}{2}(P_{i+1} - P_{i-1})$$

$$P_{i+1}' = \frac{1}{2}(P_{i+2} - P_i)$$

$$G_{\text{Hermite}}$$

$$\begin{bmatrix} \mathbf{p}(i) & \mathbf{p}(i+1) & \mathbf{p}'(i) & \mathbf{p}'(i+1) \end{bmatrix}$$

$$= \begin{bmatrix} \mathbf{p}(i-1) & \mathbf{p}(i) & \mathbf{p}(i+1) & \mathbf{p}(i+2) \end{bmatrix} \begin{bmatrix} 0 & 0 & -\frac{1}{2} & 0 \\ 1 & 0 & 0 & -\frac{1}{2} \\ 0 & 1 & \frac{1}{2} & 0 \\ 0 & 0 & 0 & \frac{1}{2} \end{bmatrix}$$

$$G$$

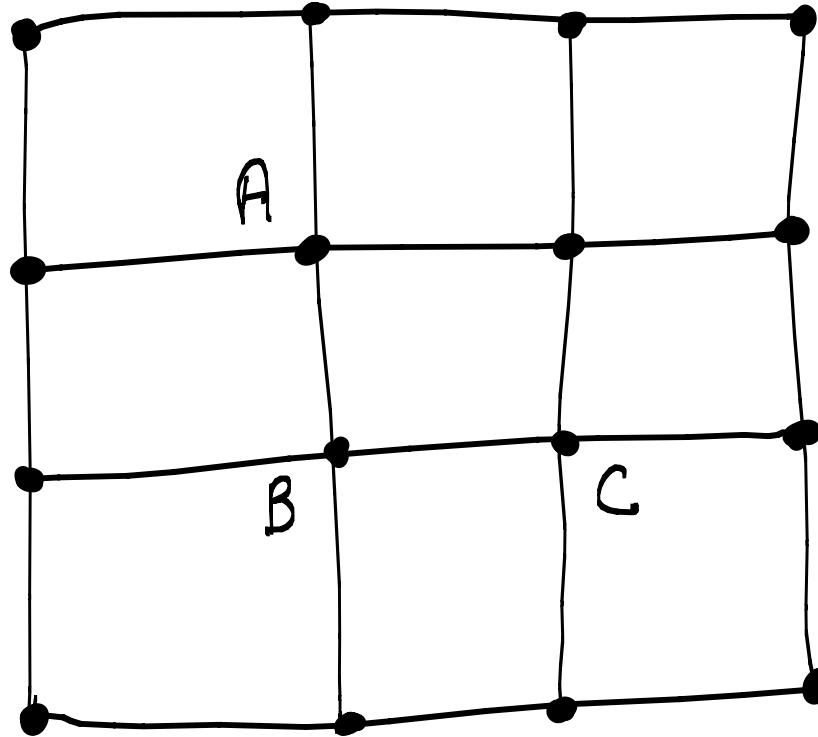$$\overbrace{\mathbf{G}_{Hermite}\mathbf{B}_{Hermite} = \mathbf{G} \begin{bmatrix} 0 & 0 & -\frac{1}{2} & 0 \\ 1 & 0 & 0 & -\frac{1}{2} \\ 0 & 1 & \frac{1}{2} & 0 \\ 0 & 0 & 0 & \frac{1}{2} \end{bmatrix}}^{B_{catmull-Rom}} \mathbf{B}_{Hermite} = \mathbf{G}\mathbf{B}_{Catmull-Rom}$$
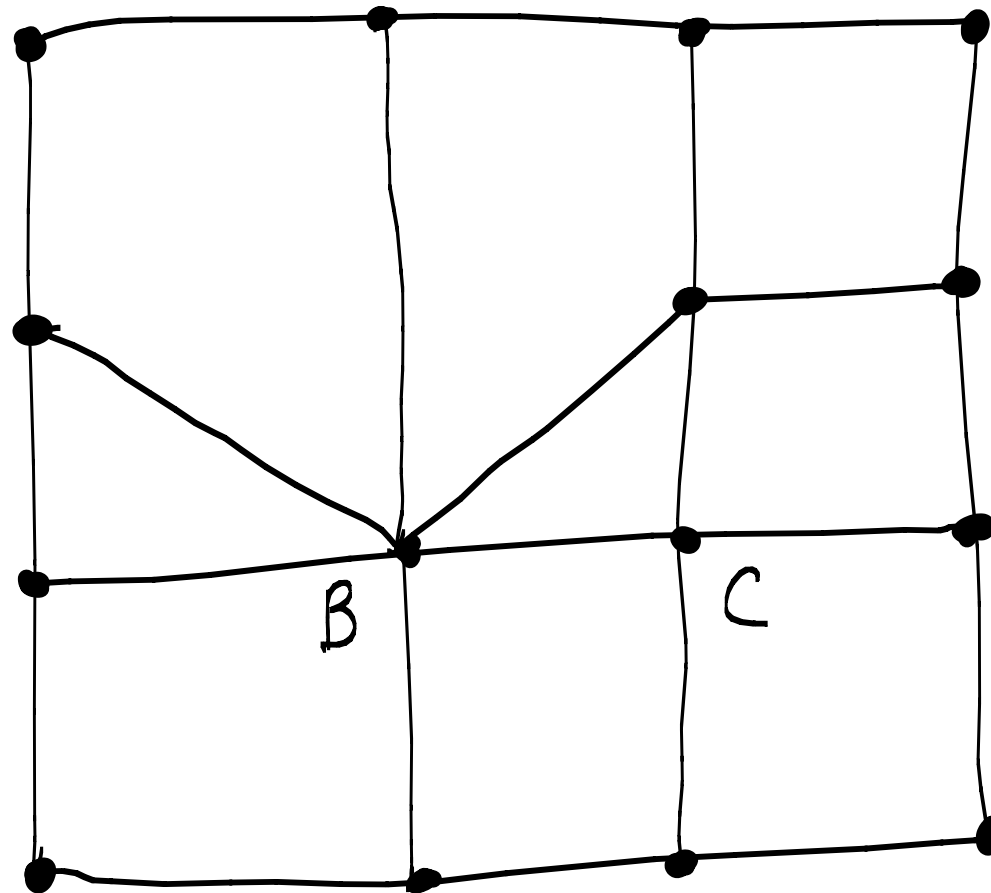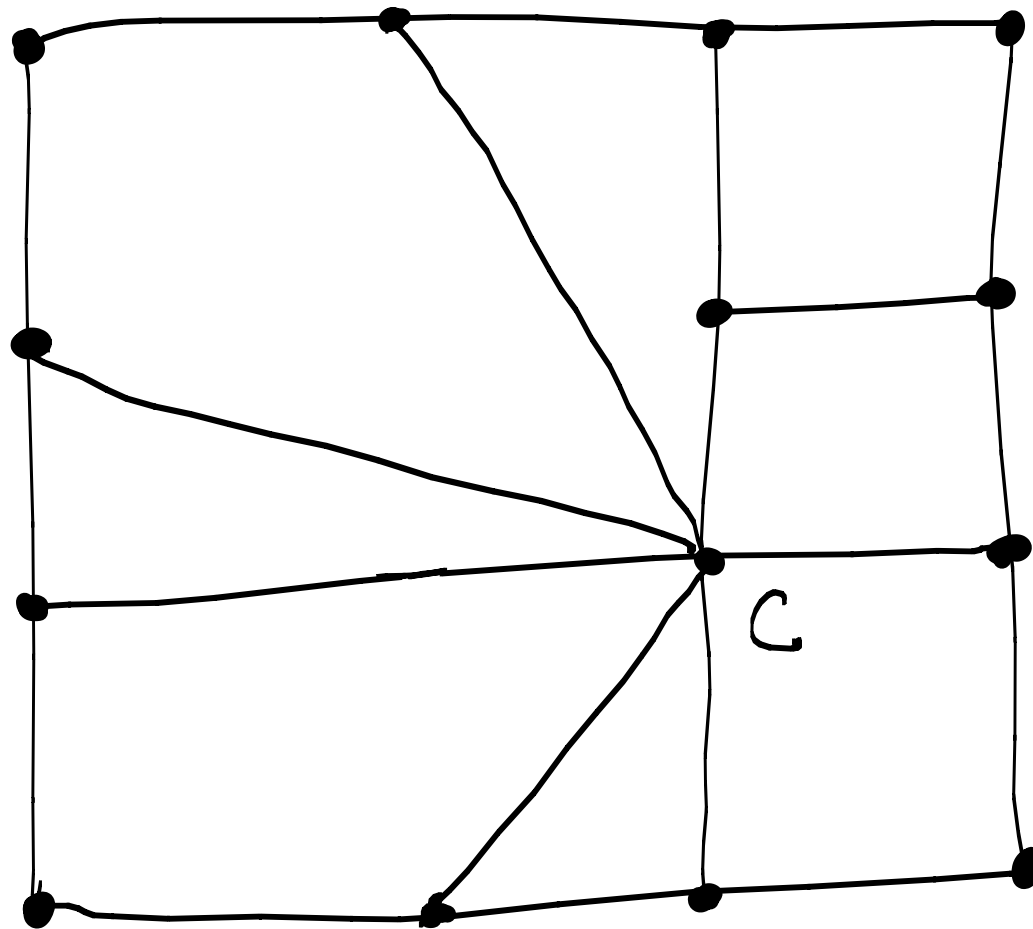
previous
slide

What happens to the mesh when we collapse AB to B
and then BC to C ?

How many vertices, edges, and faces are lost ?

One vertex and one edge are lost, but no faces.

One vertex, two edges, and one face are lost.

Announcements

Thursday: midterm

   Last name A-P (Trottier 0100),
   Last name Q-Z (Rutherford Physics 114)

There will be 10 questions, one from each lecture (2-11).

A2 : this week for sure

My office hours this week:
   Wednesday (most of afternoon)
   Thursday AM