Lecture 9

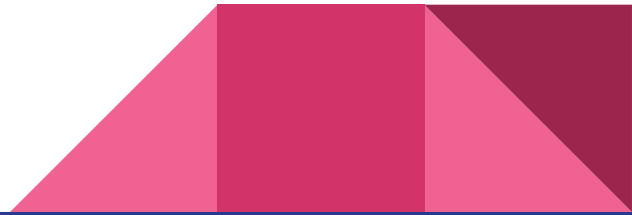
Announcements

Assignment 3 Debrief

- If you got attachment points right, congrats
- If you got Boids working, congrats even more
- Probably will post a solution this week
- Keep it private

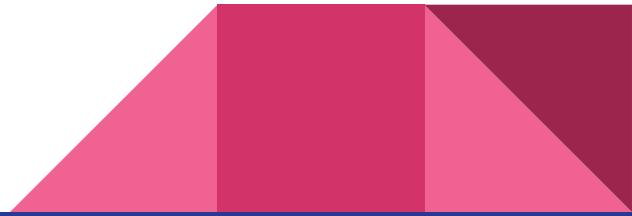
Assignment 4 Update

- Not quite ready
- You couldn’t really start it without me covering shading
- No time for that before midterm
- Consequence: Most of A5’s points will be absorbed by the group project’s points.



Survey Results!

- Survey suggestions:
  - Give hints for special topics for projects
  - Lectures linger on topics, pacing
  - Announce midterm coverage, practice sets
  - Want to know how certain low-level things
  - Post demo code



Survey Results!

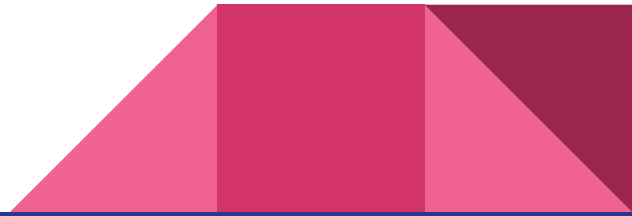
- People are mostly nervous about teams
- Remember:

Reminder: Teammate Search

- Teammate Search is open on Piazza!
- Don’t forget where it is (post #5)
  https://piazza.com/class/jtq98w8rgyg510?cid=5

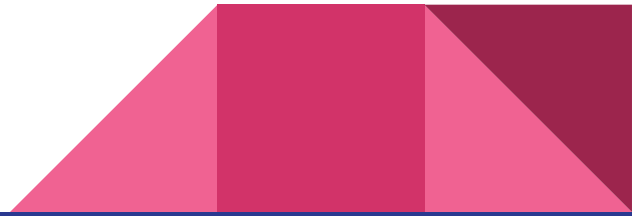
Teams

- Try Piazza team search now
- If you still don’t have a team by the midterm, I’ll help the few stragglers that are left
  - So you can get a proposal turned in.
- Most teams are not full
- Most have not discussed ideas yet
- Nobody has proposed early



Survey Results!

- Another issue: Piazza is making some people sign out a lot



Fix annoying “two Piazza accounts” issue:

- The requirement for g.ucla.edu e-mail addresses forced everyone to log out and back in to access some of their classes
- It’s much easier to just merge the two accounts
  - Just add your prior address as an additional e-mail
  - Merges automatically

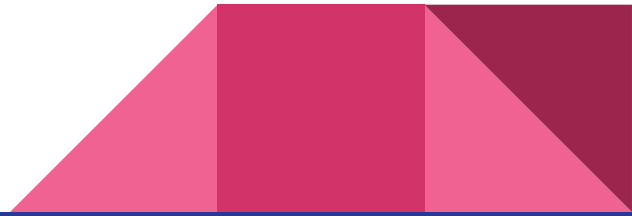
Fix annoying “two Piazza accounts” issue:

- Just visit this link to add your previous e-mail (and choose one as preferred) merging your accounts
- https://tinyurl.com/y34zg9q5

Midterm Coverage

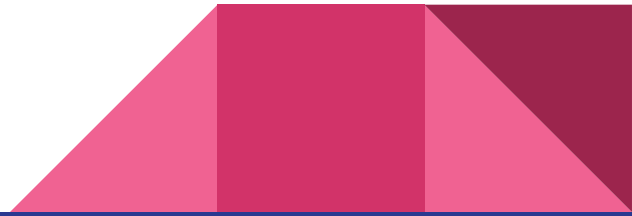
Midterm Coverage

- Everything through today. Especially:
  - Projections math
  - Matrix order
- Some pseudocode problems where you need to guess the output
- We'll provide scantrons.
- Closed book/notes/machines.



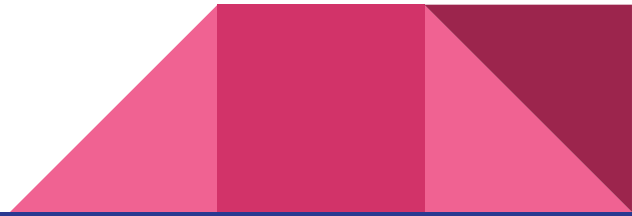
Midterm Coverage

- Book Chapters:
  - 5(!!) - viewing / projections
  - 4, (transforms)
  - 1 (graphics process)
  - 2 somewhat (graphics process)
  - Appendices B and C somewhat (algebra review)



Midterm Coverage

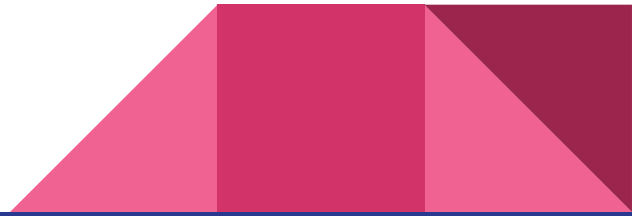
- Next week we'll do some review.
- Also practice problems



How did the look_at() line from EC1 work?

Recall how lookAt builds i,j,k vectors manually

```
model_transform = Mat4.translation( fly.position )
.times(
Mat4.inverse(
Mat4.look_at( Vec.of( 0,0,0 ), fly.velocity, Vec.of( 0,1,0 ) ) )
)
)
```

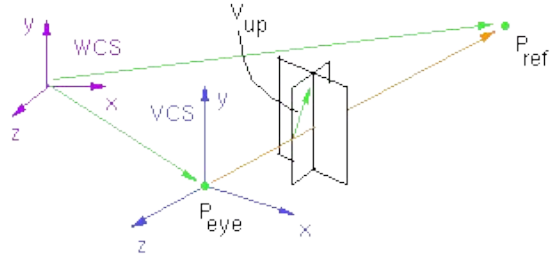


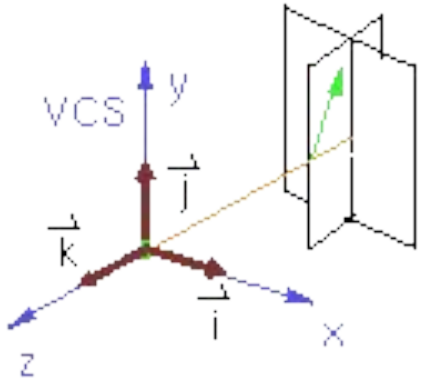
Camera Coordinate System

$$\mathbf{k} = \frac{P_{\text{eye}} - P_{\text{ref}}}{|P_{\text{eye}} - P_{\text{ref}}|}$$

$$\mathbf{i} = \frac{\mathbf{v}_{\text{up}} \times \mathbf{k}}{|\mathbf{v}_{\text{up}} \times \mathbf{k}|}$$

$$\mathbf{j} = \mathbf{k} \times \mathbf{i}$$





Building M_{cam} Inverse

Invert the smart way

$$\mathrm{M}_{\mathrm{cam}}^{-1} = \begin{bmatrix} i_x & j_x & k_x & 0 \\ i_y & j_y & k_y & 0 \\ i_z & j_z & k_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 1 & 0 & 0 & P\mathrm{eye}_x \\ 0 & 1 & 0 & P\mathrm{eye}_y \\ 0 & 0 & 1 & P\mathrm{eye}_z \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1}$$

Transpose

Negate

$$= \begin{bmatrix} i_x & i_y & i_z & 0 \\ j_x & j_y & j_z & 0 \\ k_x & k_y & k_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -P\mathrm{eye}_x \\ 0 & 1 & 0 & -P\mathrm{eye}_y \\ 0 & 0 & 1 & -P\mathrm{eye}_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$P_{\mathrm{vcs}} = \mathrm{M}_{\mathrm{cam}}^{-1} P_{\mathrm{wcs}}$$

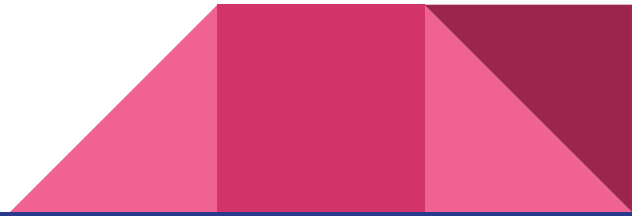
Group Project Idea Hints

What counts as an “advanced topic”?

Rest of the Course Will Switch Gears

Only two topics are left to cover:

1. Illumination and lighting
2. A million various advanced topics



Rest of the Course Will Switch Gears

1. ~~Illumination and lighting~~
2. **A million various advanced topics**

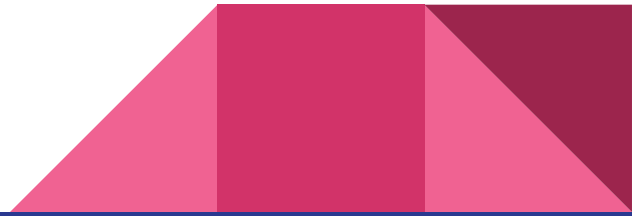
- For your group project:
  - It makes sense to individually choose topics from those to start looking into now
    - (besides illumination)

Potential “Advanced Topic” Ideas

- The following list will not be complete
- They’re only some of the possible branches you could follow into the various graphics industry techniques.

Potential “Advanced Topic” Ideas

- With some suggestions I’ll provide a Google search term or some other link to explore it more.
- For a few, I’ll give out free code later (during project 5)
  - I’ll indicate which topics those are
  - Those are still fair game for you to expand
  - I’ll just cover them too



Project Ideas - Further reading

Topics I WILL cover soon:

- Import OBJ files (new shapes from files)
  - Code is already on Piazza post 64
- Shape enhancement: Displacement maps
  - Assignment 4 extra credit? https://www.clicktorelease.com/blog/vertex-displacement-noise-3d-webgl-glsl-three-js/

Project Ideas - Further reading

Topics I WILL cover soon:

- Linear / rotational momentum
- Collision detection (shape intersection tests) and response
- Interpolated curved paths or shapes
  - Spline curves (wiki)

Project Ideas - Further reading

Topics I WILL cover soon:

- Texturing
  - Wrapping a 2D picture precisely around our shapes
  - Mip Maps (wiki)
- Texturing text onto objects
- Modifying Shaders

Project Ideas - Further reading

Topics I WILL cover soon:

- Shader effects
  - Shadow mapping (Projection onto surfaces)
  - Reflection/Environment Mapping
  - Bump mapping
- We'll modify our Shader class (read the class definition!)
- Good tutorials for those exist on Google
  - Post the good ones that you find

Project Ideas - Further reading

Topics I WILL cover soon:

- Ray Tracing
  - Computing colors one pixel at a time using line intersections with your scene
  - Normally done in fragment shaders
  - We can do it totally CPU side and render to an image if we want
    - I'll show a demo later

Project Ideas - Further reading

- Scene Graphs – manipulating a scene tree & its transforms
  - Chapter **9** in your textbook
    - Each node stores a JavaScript array of references to child nodes… a whole tree data structure results.
    - Each node also stores its delta matrix (to be post-multiplied to model transform)
  - The tree's transforms can be altered procedurally
    - My own 174 term project (individual): https://www.youtube.com/watch?v=NzpVd1Rk-xQ

Project Ideas - Further reading

Topics I probably won't cover:

- Procedural textures (perlin noise (wiki), shadertoy.com code)
- Shader particle effects (sparkles, fire, smoke, splashing, etc.)
- Procedural shapes (plants) – L-systems (wiki)

Project Ideas - Further reading

Topics I probably won't cover:

- Animation
  - Spring/damper physics
    - Uses algebra & neighborhood graphs
  - Movement interpolation - "tweening" (wiki)
  - Perfect arc swinging
    - Camera aim interpolation - Quaternion math, geodesic lines

Project Ideas - Further reading

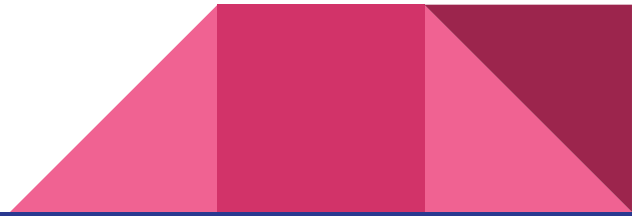
Topics I probably won't cover:

- Marching cubes (wiki) / drawing implicit fields & volumes
  - Level set method (wiki)
- Fractal generation & rendering – look up Mandelbulb videos
  - https://www.youtube.com/watch?v=Yb5MRbgNKSk
  - https://www.youtube.com/watch?v=cPbKP2ep05k
  - https://www.youtube.com/watch?v=KdJepLvW66U
- Mouse picking (clicking to correctly select objects)

Suggestion: Pick a “Theme” last

- Pick an advanced graphics topic, then:
  1. Implement it
  2. Game-ify it
  3. Theme it last

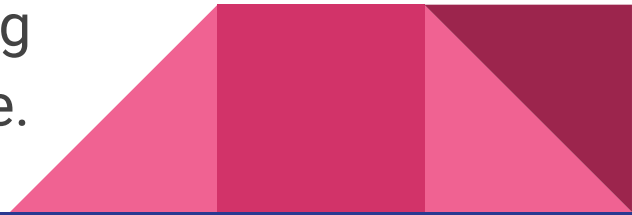
- If you do this, choose an advanced topic first.
  - Once it works, make it interactive or turn it into a challenge.



Suggestion: Pick a “Theme” last

- Alternatively, if you like modeling shapes more:
  - Build some models first
  - Theme based around whatever came out best

- If that’s more your thing, make 3D objects first, then tell a story about the objects you manage to make.
- Good stories are drawn from real events / existing media.
  - Even re-telling a story or re-implementing an established game can be worthwhile.



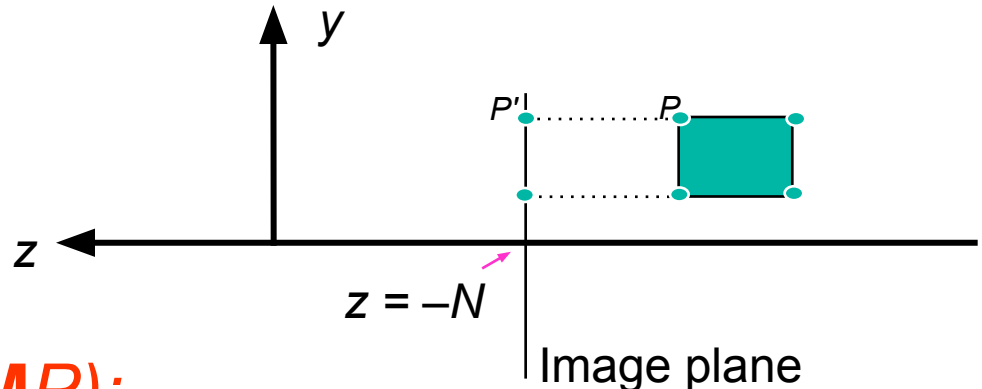
Finish last week’s projection math:

Basic Orthographic Projection

$$P'_x = P_x$$

$$P'_y = P_y$$

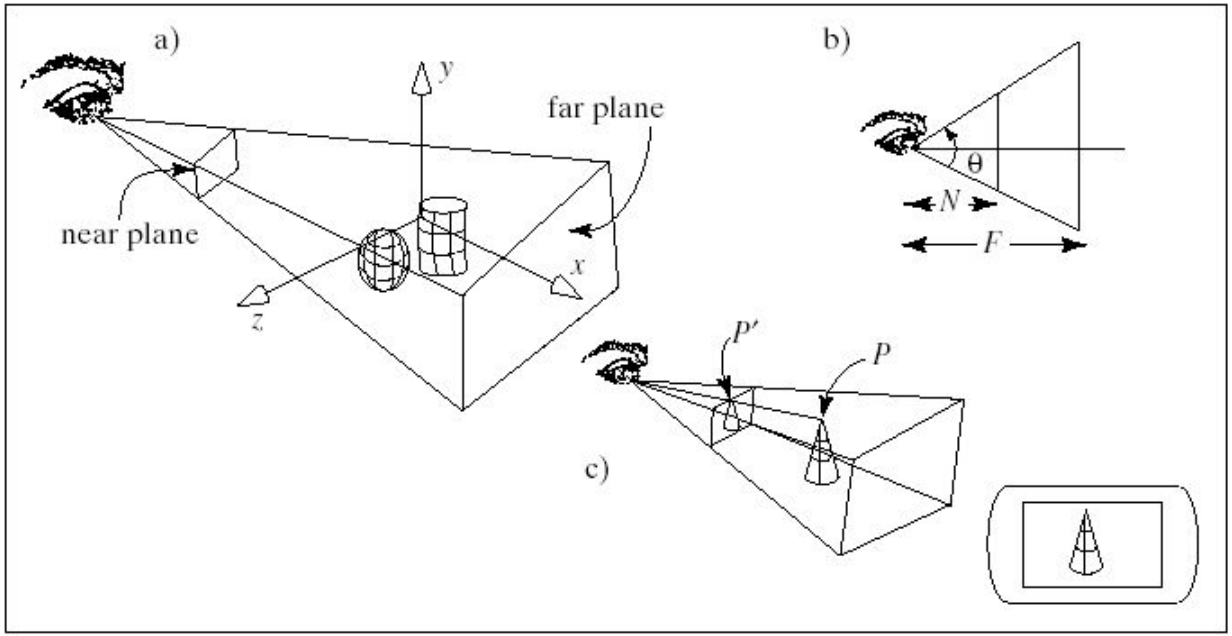
$$P'_z = -N$$



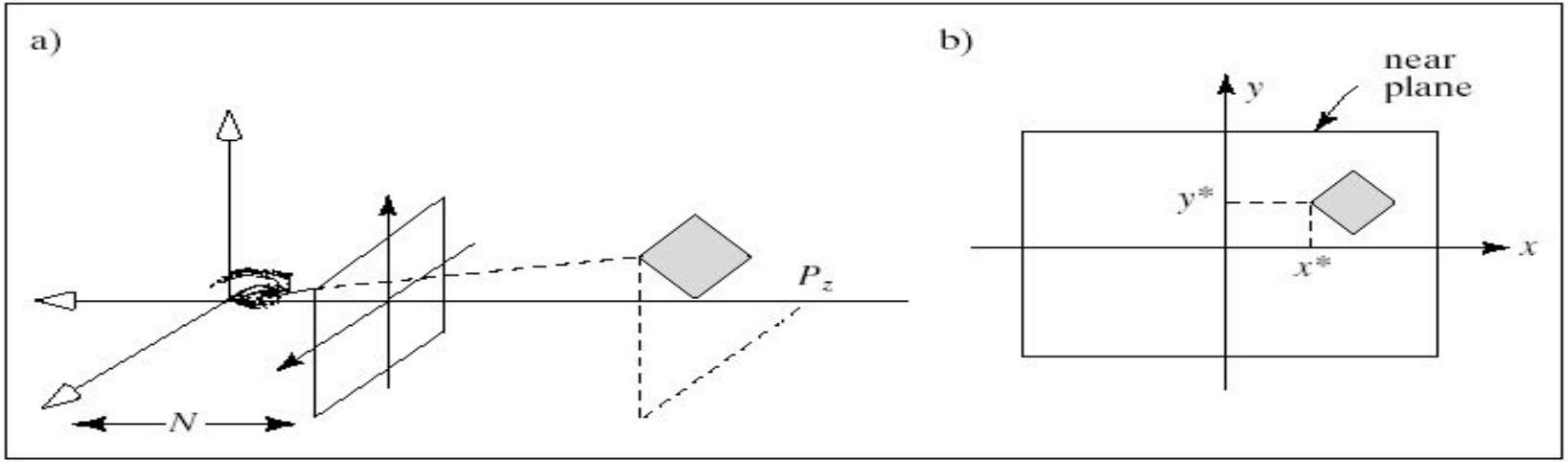
Matrix Form *(P′ =* **M***P):*

$$\begin{bmatrix} P'_x \\ P'_y \\ P'_z \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -N \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} P_x \\ P_y \\ P_z \\ 1 \end{bmatrix}$$

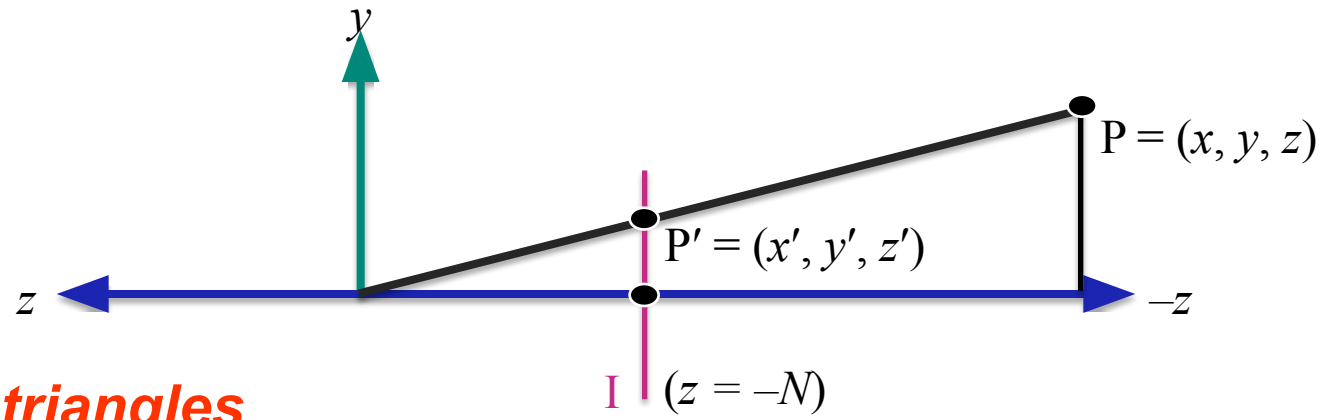
Perspective Projection



Perspective Projection of a Point



Basic Perspective Projection



Similar triangles

$$y' / N = y / -z \quad \Rightarrow \quad P_y' = P_y N / -P_z$$

Similarly

$$P_x' = P_x N / -P_z$$

$$P_z' = -N$$

This is a non-linear transformation!

Observations

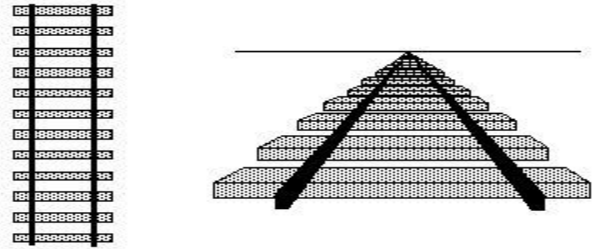
- Projection undefined for $P_z = 0$
- If P is behind the eye,
  P_z changes sign
- Near plane just scales the picture
- Straight line → straight line
- Perspective foreshortening

$$P'_x = -N\frac{P_x}{P_z}$$
$$P'_y = -N\frac{P_y}{P_z}$$
$$P'_z = -N$$

Be Able to Answer:

- ***Given a point in x, y, z space, how do we calculate where it appears on the screen?***
- ***How is the perspective projection different from affine transformations?***
- ***What do perspective projections preserve?***
  - Parallel lines?
  - Ratios of points along a line?

Perspective transforms



What happens to parallel lines during the transform?

What happens to ratios along straight lines?

In Homogeneous Matrix Form

Reminder:

$$\begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix} \rightarrow \begin{bmatrix} P_x \\ P_y \\ P_z \\ 1 \end{bmatrix} \xrightarrow[\times w]{} \begin{bmatrix} wP_x \\ wP_y \\ wP_z \\ w \end{bmatrix} \xrightarrow[\div w]{\text{homogenize}} \begin{bmatrix} P_x \\ P_y \\ P_z \\ 1 \end{bmatrix} \rightarrow \begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix}$$

(a line in 4D space)

Perspective projection:

$$\begin{bmatrix} P'_x \\ P'_y \\ P'_z \\ 1 \end{bmatrix} = \begin{bmatrix} P_x N/(-P_z) \\ P_y N/(-P_z) \\ -N \\ 1 \end{bmatrix} \xrightarrow[-P_z/N]{\times} \begin{bmatrix} P_x \\ P_y \\ P_z \\ -P_z/N \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -1/N & 0 \end{bmatrix} \begin{bmatrix} P_x \\ P_y \\ P_z \\ 1 \end{bmatrix}$$

Therefore:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -1/N & 0 \end{bmatrix} \begin{bmatrix} P_x \\ P_y \\ P_z \\ 1 \end{bmatrix} \xrightarrow[\div -P_z/N]{\text{and then: homogenize}} \begin{bmatrix} P'_x \\ P'_y \\ P'_z \\ 1 \end{bmatrix}$$

Matrix **M**

Homogenization step:
"Perspective Division"
(divide by $w = -P_z/N$)

Perspective Projection of a Line

$$L(t) = \mathbf{P} + \mathbf{v}t = \begin{bmatrix} P_x \\ P_y \\ P_z \\ 1 \end{bmatrix} + \begin{bmatrix} v_x \\ v_y \\ v_z \\ 0 \end{bmatrix} t$$

Perspective Division & drop fourth coordinate

Is it still a line?

Original: $L(t) = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} P_x + v_x t \\ P_y + v_y t \\ P_z + v_z t \end{bmatrix}$

Projected: $L'(t) = \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} -Nx/z \\ -Ny/z \\ -N \end{bmatrix} = \begin{bmatrix} -N(P_x + v_x t)/(P_z + v_z t) \\ -N(P_y + v_y t)/(P_z + v_z t) \\ -N \end{bmatrix}$

$$x' = -N(P_x + v_x t)/(P_z + v_z t) \Rightarrow x'(P_z + v_z t) = -N(P_x + v_x t) \Rightarrow$$

$$x' P_z + x' v_z t = -NP_x - Nv_x t \Rightarrow \begin{cases} x' P_z + NP_x = -(x' v_z + Nv_x)t \\ \text{and similarly for y:} \\ y' P_z + NP_y = -(y' v_z + Nv_y)t \end{cases}$$

Is it still a line? (cont'd)

$$\left.\begin{aligned} x'P_z + NP_x &= -(x'v_z + Nv_x)t \\ y'P_z + NP_y &= -(y'v_z + Nv_y)t \end{aligned}\right| \Rightarrow \left.\begin{aligned} x'P_z + NP_x &= -(x'v_z + Nv_x)t \\ (y'v_z + Nv_y)t &= -(y'P_z + NP_y) \end{aligned}\right| \Rightarrow$$

Multiply the two equations and divide through by *t*

$$(x'P_z + NP_x)(y'v_z + Nv_y) = (x'v_z + Nv_x)(y'P_z + NP_y) \Rightarrow$$

$$x'P_z y'v_z + x'P_z Nv_y + NP_x y'v_z + N^2 P_x v_y = x'v_z y'P_z + x'v_z NP_y + Nv_x y'P_z + N^2 P_y v_x \Rightarrow$$

$$(P_z Nc_y - v_z NP_y)x' + (NP_x v_z + Nv_x P_z)y' + N^2(P_x v_y + P_y v_x) = 0 \Rightarrow$$

$\Rightarrow$ $ax' + by' + c = 0$ which is the equation of a line in the x'-y' plane

But is There a Difference?

Original: $$L(t) = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} P_x + v_x t \\ P_y + v_y t \\ P_z + v_z t \end{bmatrix}$$

Projected: $$L'(t) = \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} -Nx / z \\ -Ny / z \\ -N \end{bmatrix} = \begin{bmatrix} -N(P_x + v_x t) / (P_z + v_z t) \\ -N(P_y + v_y t) / (P_z + v_z t) \\ -N \end{bmatrix}$$

But is There a Difference?

The "speed along the lines" if $v_z \neq 0$

$$\text{Original:}\quad L(t) = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} P_x + v_x t \\ P_y + v_y t \\ P_z + v_z t \end{bmatrix} \Rightarrow \frac{\partial L(t)}{\partial t} = \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} = \mathbf{v}$$

$$\text{Projected:}\quad L'(t) = \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} -Nx/z \\ -Ny/z \\ -N \end{bmatrix} = \begin{bmatrix} -N(P_x + v_x t)/(P_z + v_z t) \\ -N(P_y + v_y t)/(P_z + v_z t) \\ -N \end{bmatrix} \Rightarrow$$

$$\frac{\partial x'}{\partial t} = -N\frac{\partial}{\partial t}((P_x + v_x t)/(P_z + v_z t)) = -N\frac{v_x(P_z + v_z t) - (P_x + v_x t)v_z}{(P_z + v_z t)^2} = -N\frac{v_x P_z - P_x v_z}{(P_z + v_z t)^2} \Rightarrow$$

$$\frac{\partial L'(t)}{\partial t} = \frac{-N}{(P_z + v_z t)^2}\begin{bmatrix} v_x P_z - P_x v_z \\ v_y P_z - P_y v_z \\ 0 \end{bmatrix}$$

As time *t* tends to infinity, the speed along the projected line *L'* tends to zero

Effect of Perspective Projection on Lines

Line equations

$$\text{Original:}\quad L(t) = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} P_x + v_x t \\ P_y + v_y t \\ P_z + v_z t \end{bmatrix}$$

$$\text{Projected:}\quad L'(t) = \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} -Nx/z \\ -Ny/z \\ -N \end{bmatrix} = \begin{bmatrix} -N(P_x + v_x t)/(P_z + v_z t) \\ -N(P_y + v_y t)/(P_z + v_z t) \\ -N \end{bmatrix}$$

If lines in space are parallel to the image plane then:

$$v_z = 0 \rightarrow L'(t) = -\frac{N}{P_z}\begin{bmatrix} P_x + v_x t \\ P_y + v_y t \\ P_z \end{bmatrix}$$

slope of line: $\frac{v_y}{v_x}$

So, parallel lines parallel to the image plane remain parallel

Effect of Perspective Projection on Lines

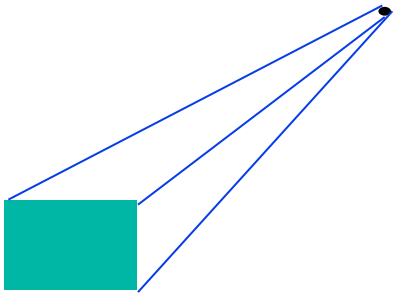
Line equations (again)

$$\text{Original:}\quad L(t) = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} P_x + v_x t \\ P_y + v_y t \\ P_z + v_z t \end{bmatrix}$$

$$\text{Projected:}\quad L'(t) = \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} -Nx/z \\ -Ny/z \\ -N \end{bmatrix} = \begin{bmatrix} -N(P_x + v_x t)/(P_z + v_z t) \\ -N(P_y + v_y t)/(P_z + v_z t) \\ -N \end{bmatrix}$$

If lines are not parallel to the image plane then:

$$v_z \neq 0 \rightarrow \lim_{t \to \infty} L'(t) = \begin{bmatrix} -Nv_x/v_z \\ -Nv_y/v_z \\ -N \end{bmatrix}$$



Lines converge to a **vanishing point**

Summary

Perspective projection is non-linear

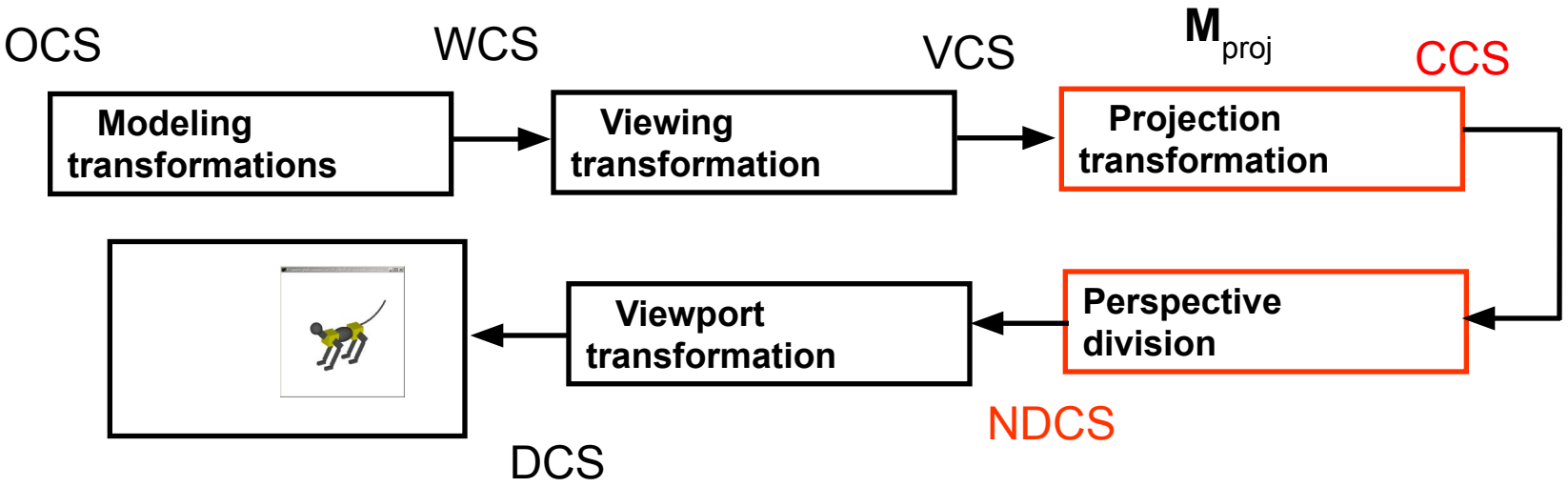
Lines project to lines

Parallel lines either project to parallel lines or they intersect at the vanishing point

Foreshortening of projected lines and the "Inbetweeness" relationship

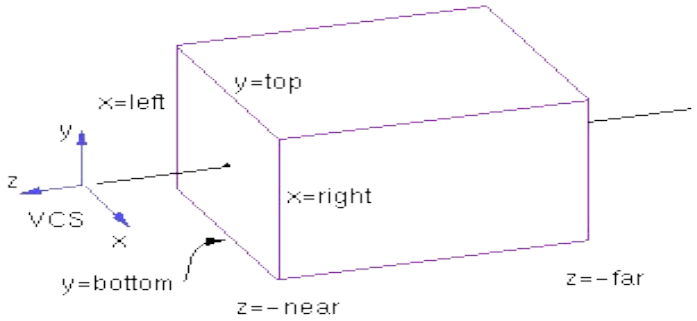
View Volumes

Graphics Pipeline



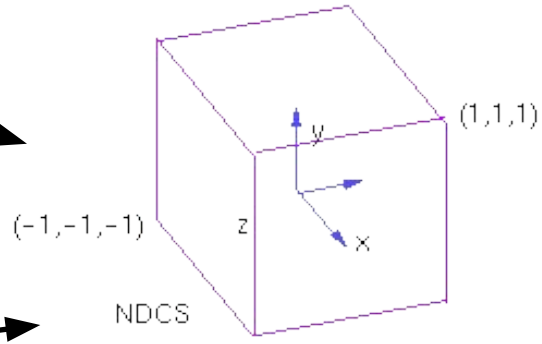
View Volumes

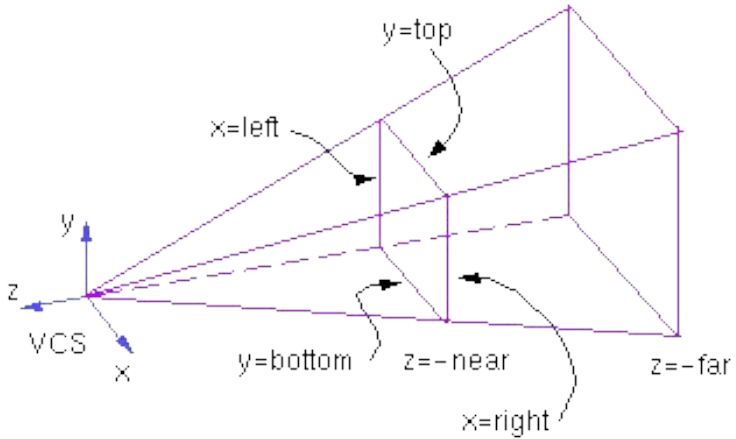
We display only the geometry in a view volume.



Orthographic

The “Canonical View Volume” or “Normalized Display Coord. System” (NDCS)





Perspective

A left-handed coordinate system at the center of a 2 x 2 x 2 cube.

Nonlinearity of Perspective Transformation

Railroad tracks:

z = –infinity to +infinity

Left track: x = –1, y = –1

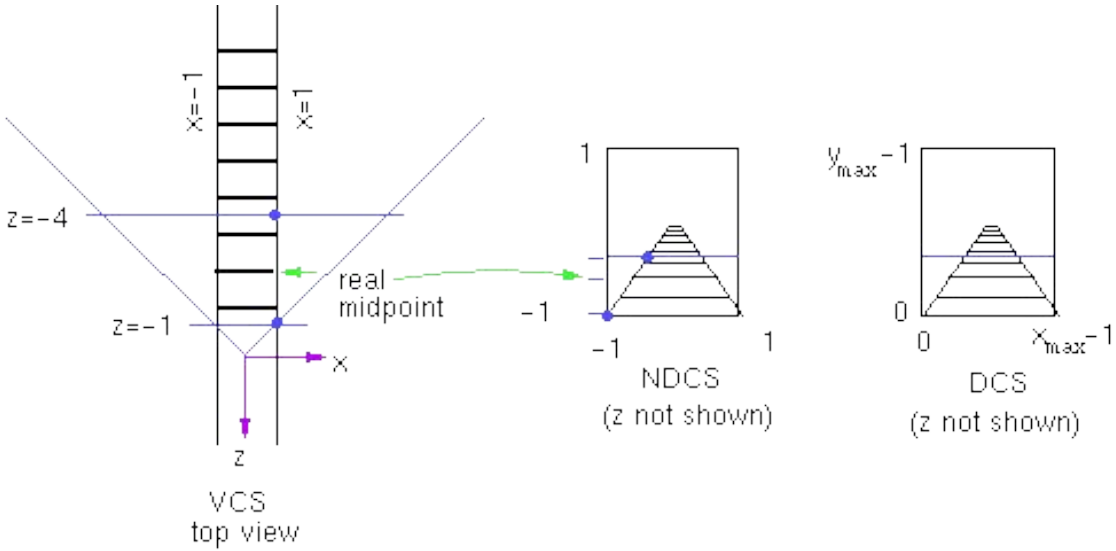
Right track: x = +1, y = –1

View volume:

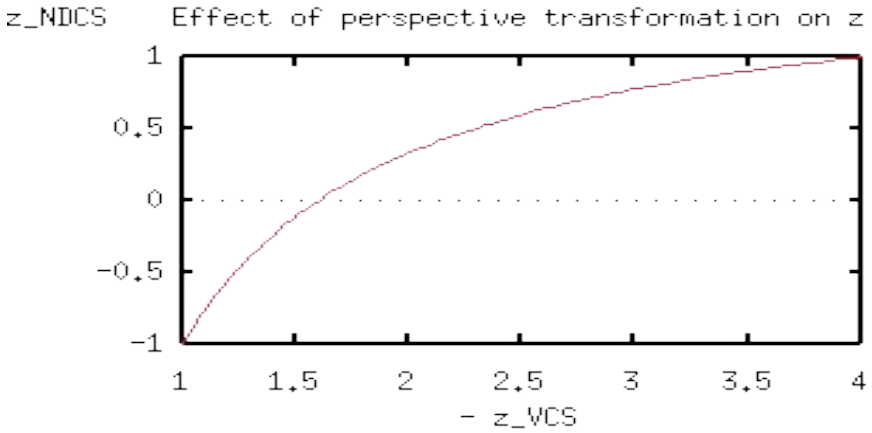
Left = –1, Right = +1

Bottom = –1, Top = +1

Near = +1, Far = +4



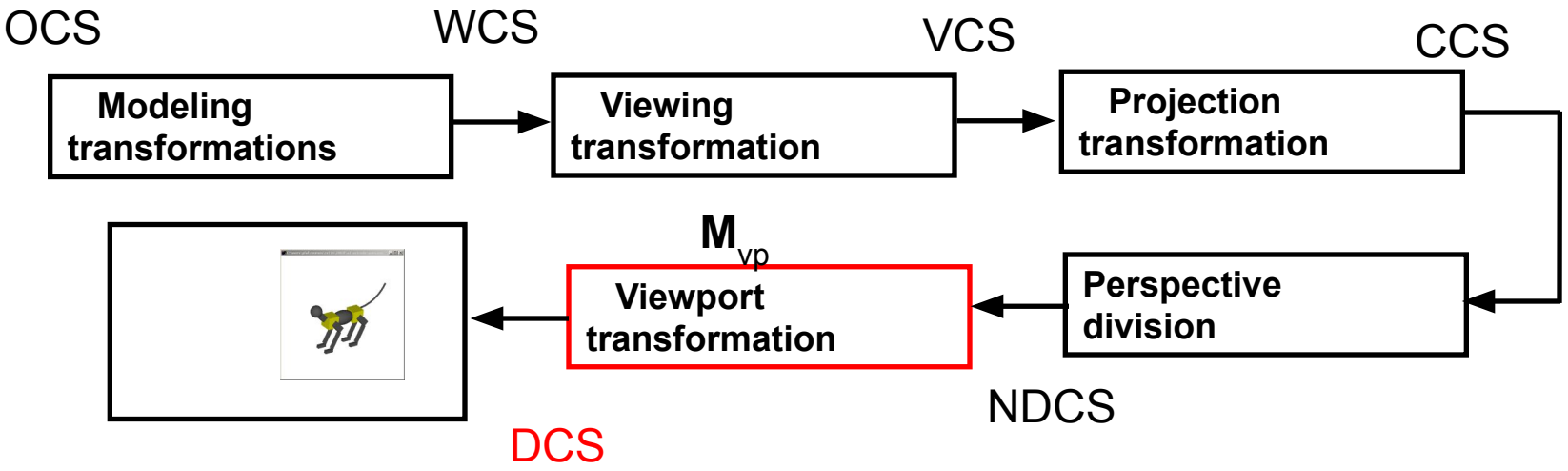
z in NDCS vs –*z* in VCS



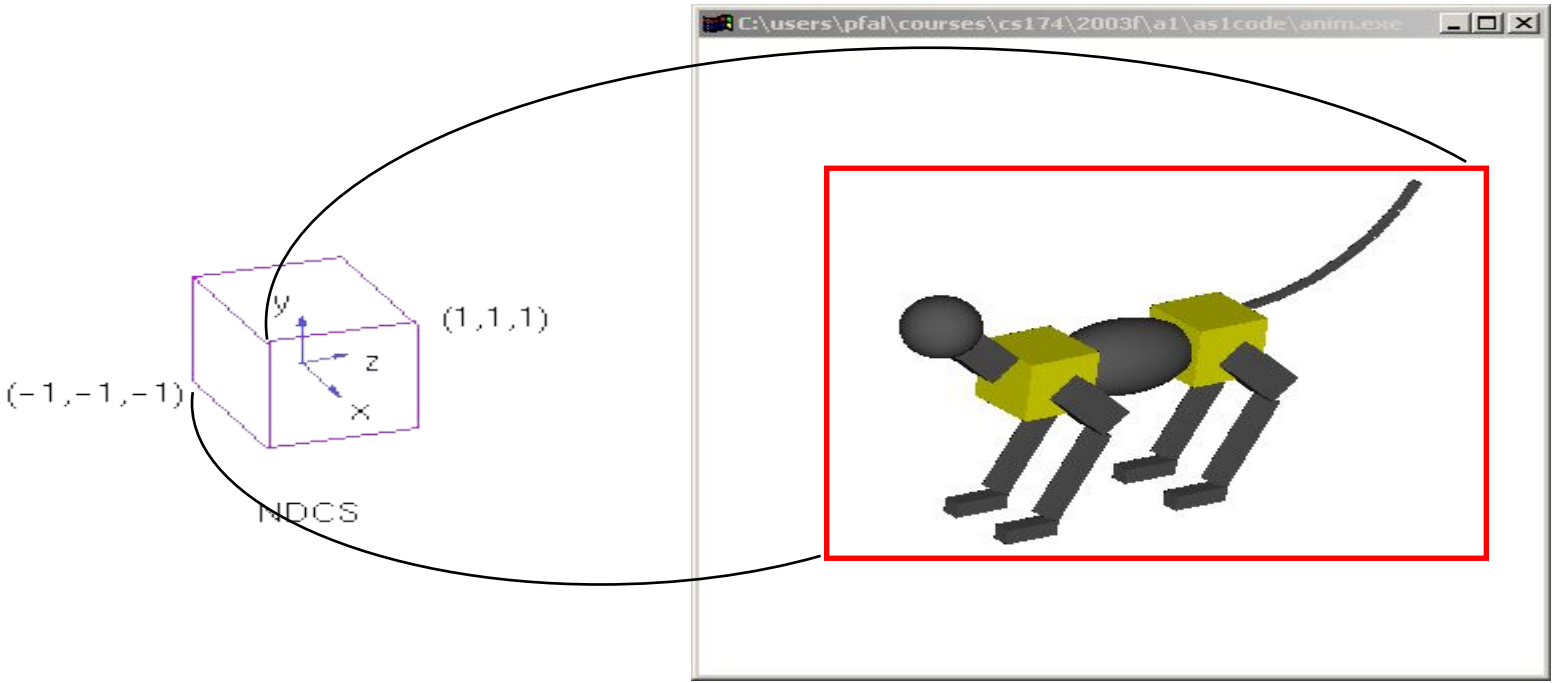
near plane

far plane

Viewport Transformation



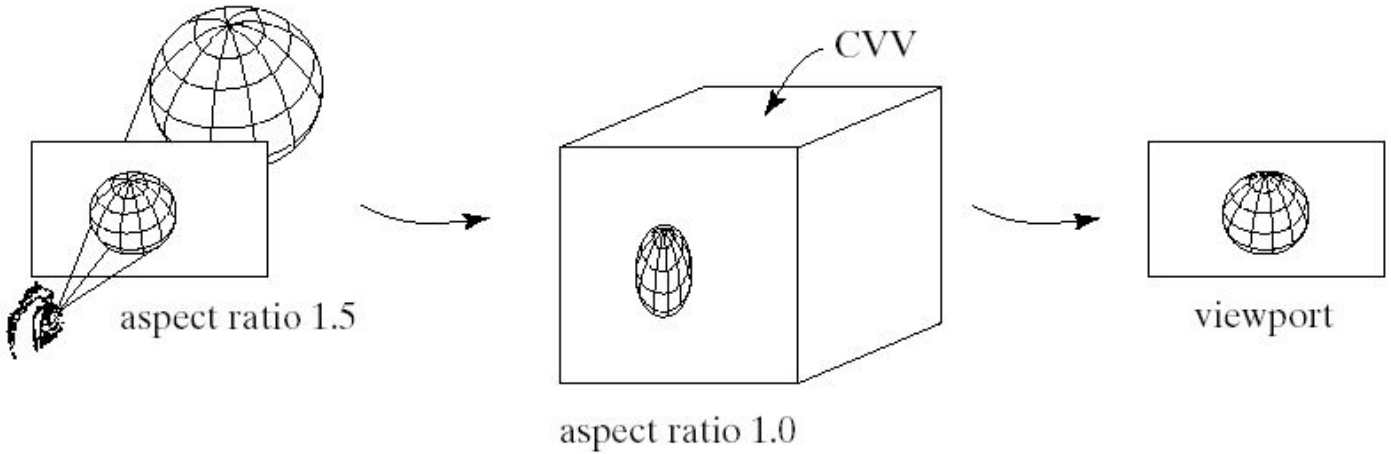
Viewport


C:\users\pfal\courses\cs174\2003f\a1\as1code\anim.exe
y
z
x
(1,1,1)
(-1,-1,-1)
NDCS

Why Viewports?

Undo the distortion of the projection transformation

Map to pixel coordinates on screen



Stereo Views