3D Viewing and Rasterization

CS 4620 Lecture 15

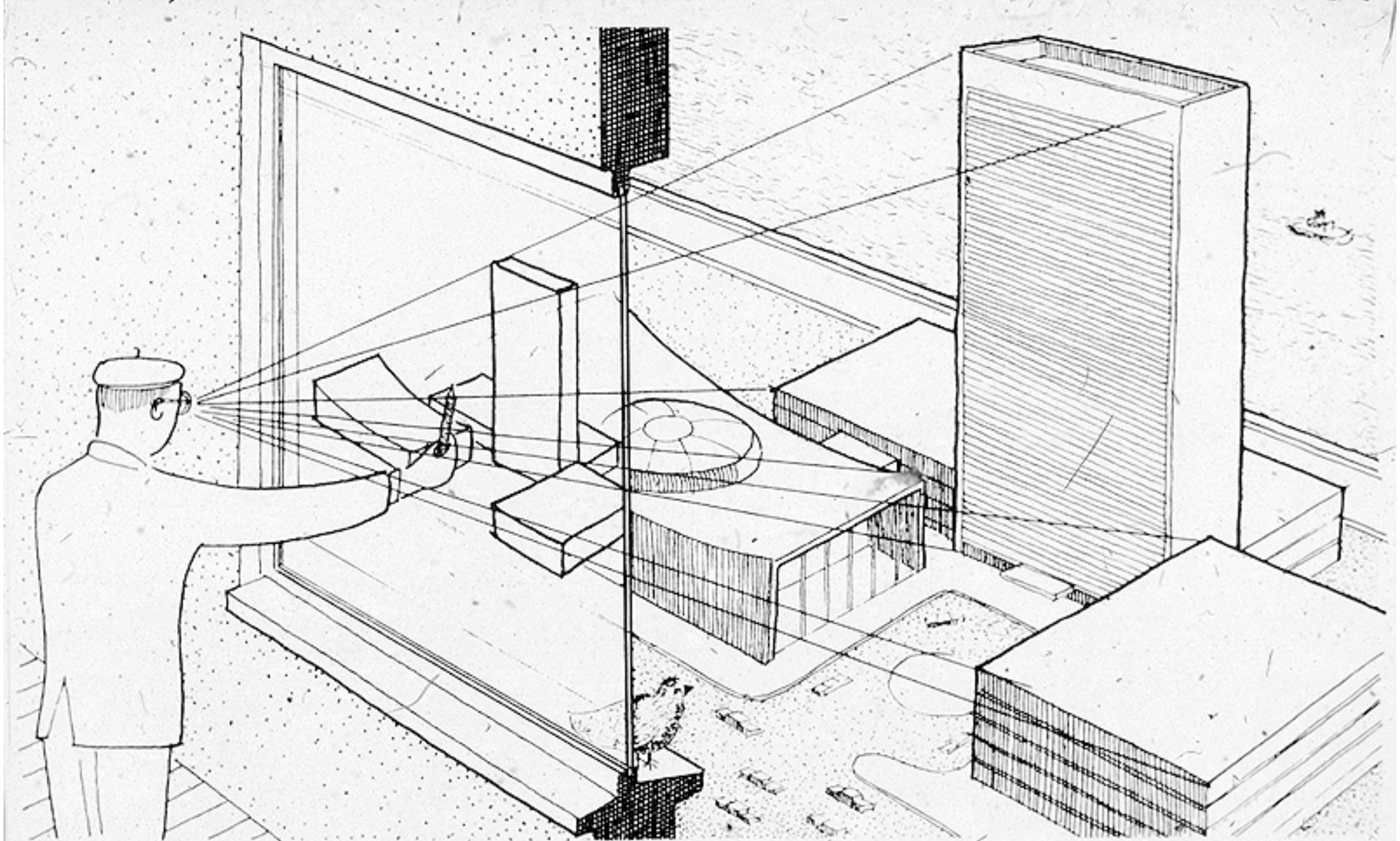
Announcements

- A3 due next Thu
  - Will send mail about grading once finalized

- No 4621 class today

Plane projection in drawing

Basis Of Perspective – Lines Of Sight Through A Picture Plane [19]

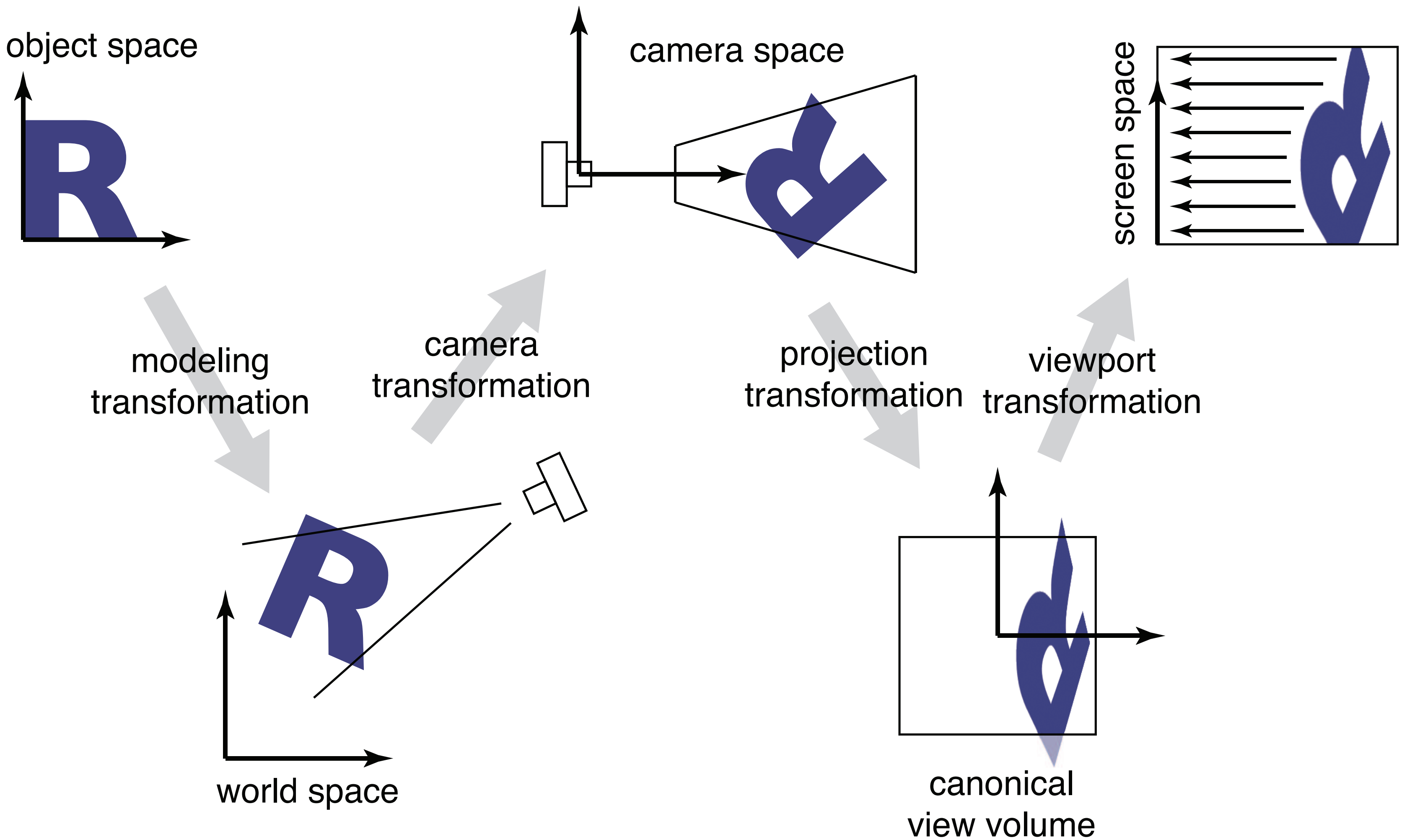


The concept of the picture plane may be better understood by looking through a window or other transparent plane from a fixed viewpoint. Your lines of sight, the multitude of straight lines leading from your eye to the subject, will all intersect this plane. Therefore, if you were to reach out with a grease pencil and draw the image of the subject on this plane you would be "tracing out" the infinite number of points of intersection of sight rays and plane. The result would be that you would have "transferred" a real three-dimensional object to a two-dimensional plane.

source unknown

Pipeline of transformations

- Standard sequence of transforms



Orthographic transformation chain

- Start with coordinates in object's local coordinates
- Transform into world coords (modeling transform, M_m)
- Transform into eye coords (camera xf., $M_{cam} = F_c^{-1}$)
- Orthographic projection, M_{orth}
- Viewport transform, M_{vp}

$$\mathbf{p}_s = \mathbf{M}_{\mathrm{vp}}\mathbf{M}_{\mathrm{orth}}\mathbf{M}_{\mathrm{cam}}\mathbf{M}_{\mathrm{m}}\mathbf{p}_o$$

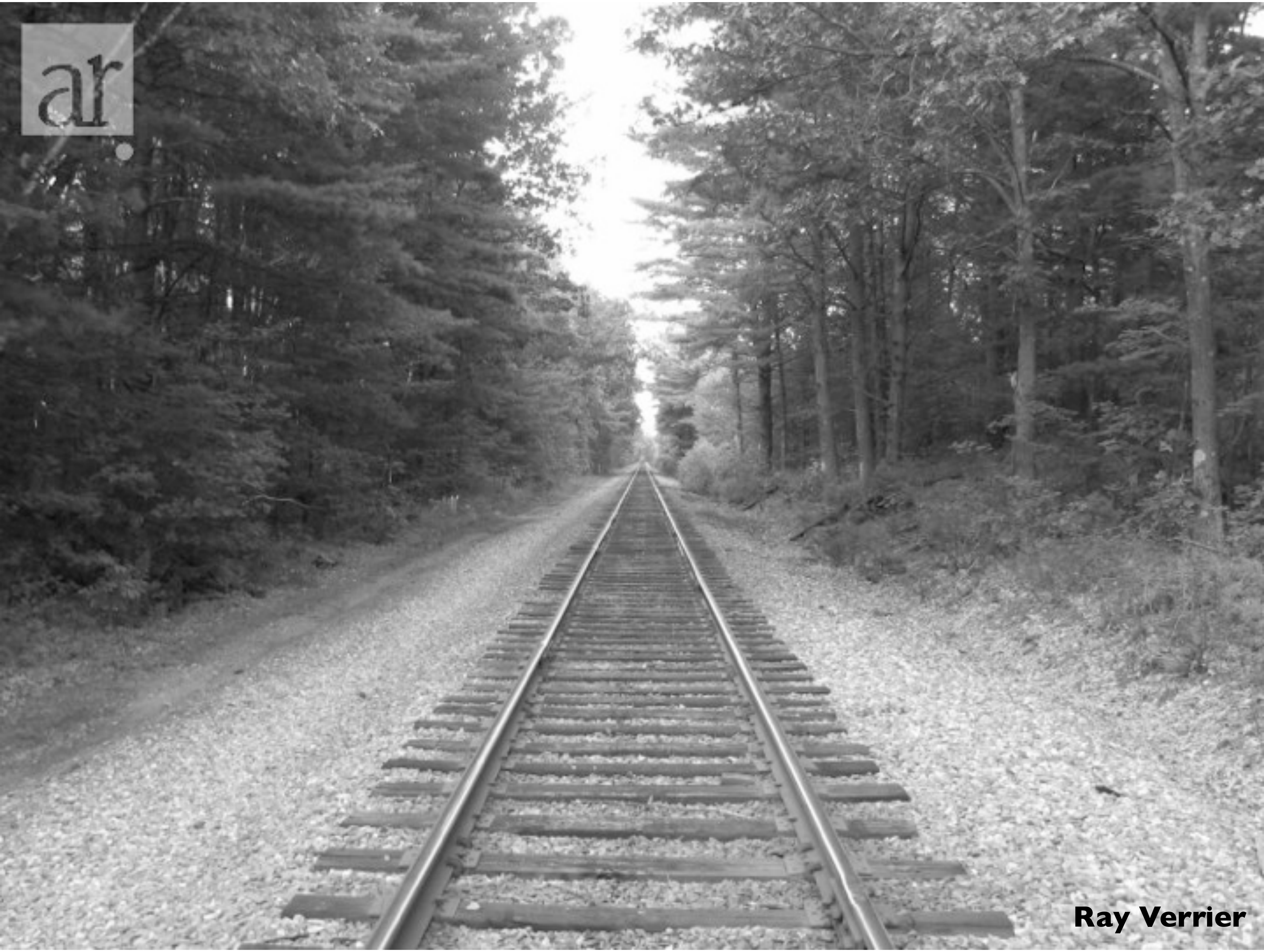
$$\begin{bmatrix} x_s \\ y_s \\ z_c \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{n_x}{2} & 0 & 0 & \frac{n_x-1}{2} \\ 0 & \frac{n_y}{2} & 0 & \frac{n_y-1}{2} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{2}{r-l} & 0 & 0 & -\frac{r+l}{r-l} \\ 0 & \frac{2}{t-b} & 0 & -\frac{t+b}{t-b} \\ 0 & 0 & \frac{2}{n-f} & -\frac{n+f}{n-f} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{u} & \mathbf{v} & \mathbf{w} & \mathbf{e} \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} \mathbf{M}_{\mathrm{m}} \begin{bmatrix} x_o \\ y_o \\ z_o \\ 1 \end{bmatrix}$$

Perspective transformation chain

- Transform into world coords (modeling transform, M_m)
- Transform into eye coords (camera xf., $M_{\text{cam}} = F_c^{-1}$)
- Perspective matrix, P
- Orthographic projection, M_{orth}
- Viewport transform, M_{vp}

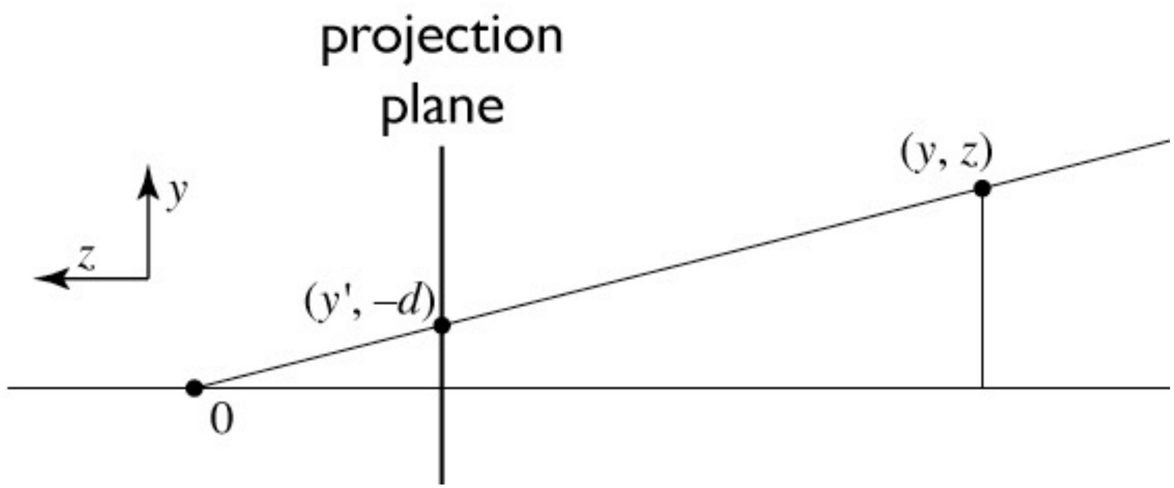
$$\mathbf{p}_s = \mathbf{M}_{\text{vp}}\mathbf{M}_{\text{orth}}\mathbf{P}\mathbf{M}_{\text{cam}}\mathbf{M}_{\text{m}}\mathbf{p}_o$$

$$\begin{bmatrix} x_s \\ y_s \\ z_c \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{n_x}{2} & 0 & 0 & \frac{n_x-1}{2} \\ 0 & \frac{n_y}{2} & 0 & \frac{n_y-1}{2} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{2}{r-l} & 0 & 0 & -\frac{r+l}{r-l} \\ 0 & \frac{2}{t-b} & 0 & -\frac{t+b}{t-b} \\ 0 & 0 & \frac{2}{n-f} & -\frac{n+f}{n-f} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} n & 0 & 0 & 0 \\ 0 & n & 0 & 0 \\ 0 & 0 & n+f & -fn \\ 0 & 0 & 1 & 0 \end{bmatrix} \mathbf{M}_{\text{cam}}\mathbf{M}_{\text{m}} \begin{bmatrix} x_o \\ y_o \\ z_o \\ 1 \end{bmatrix}$$



Ray Verrier

Perspective projection



similar triangles:

$$\frac{y'}{d} = \frac{y}{-z}$$

$$y' = -dy/z$$

Homogeneous coordinates revisited

- Perspective requires division
  - that is not part of affine transformations
  - in affine, parallel lines stay parallel
    - therefore not vanishing point
    - therefore no rays converging on viewpoint
- “True” purpose of homogeneous coords: projection

Homogeneous coordinates revisited

- Introduced *w* = 1 coordinate as a placeholder

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} \rightarrow \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

  – used as a convenience for unifying translation with linear

- Can also allow arbitrary *w*

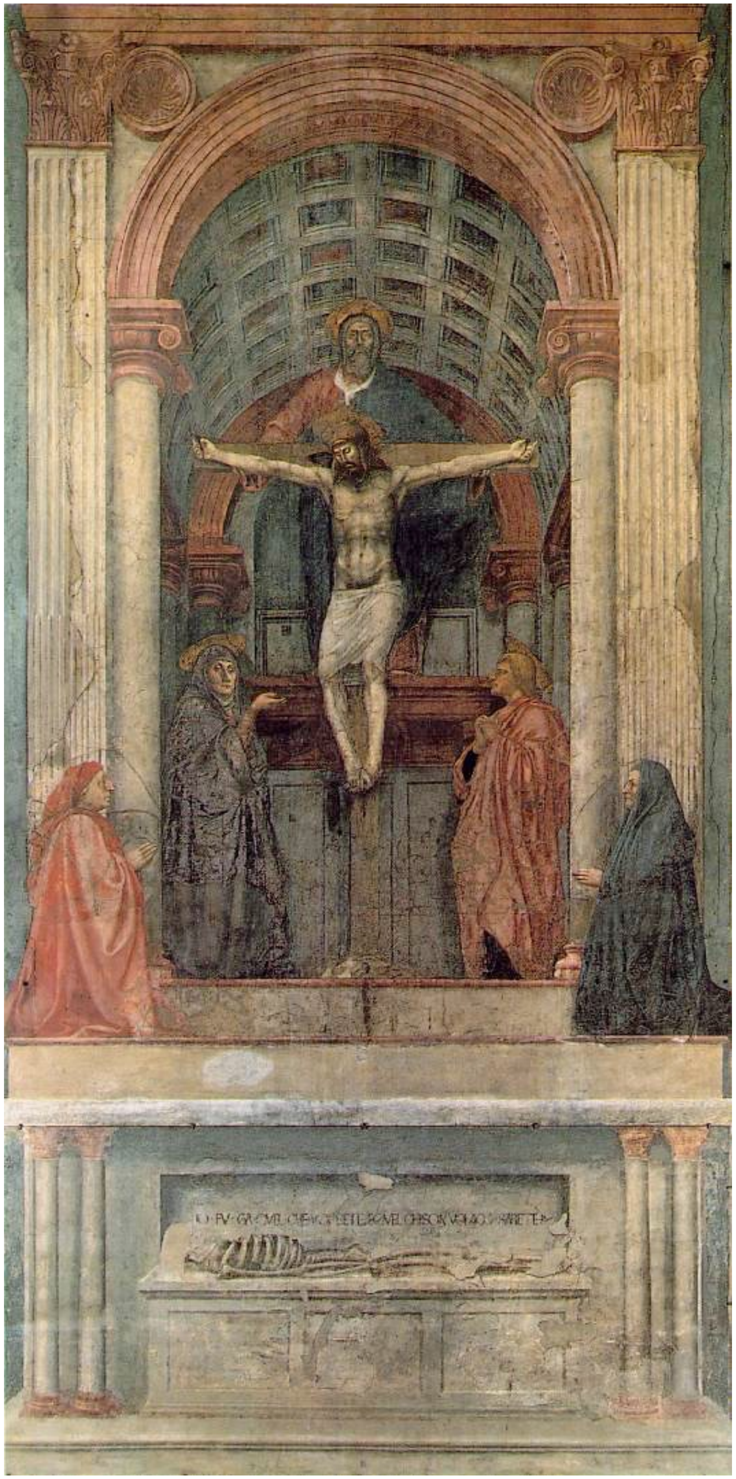
$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \sim \begin{bmatrix} wx \\ wy \\ wz \\ w \end{bmatrix}$$

- http://www.tomdalling.com/blog/modern-opengl/explaining-homogenous-coordinates-and-projective-geometry/

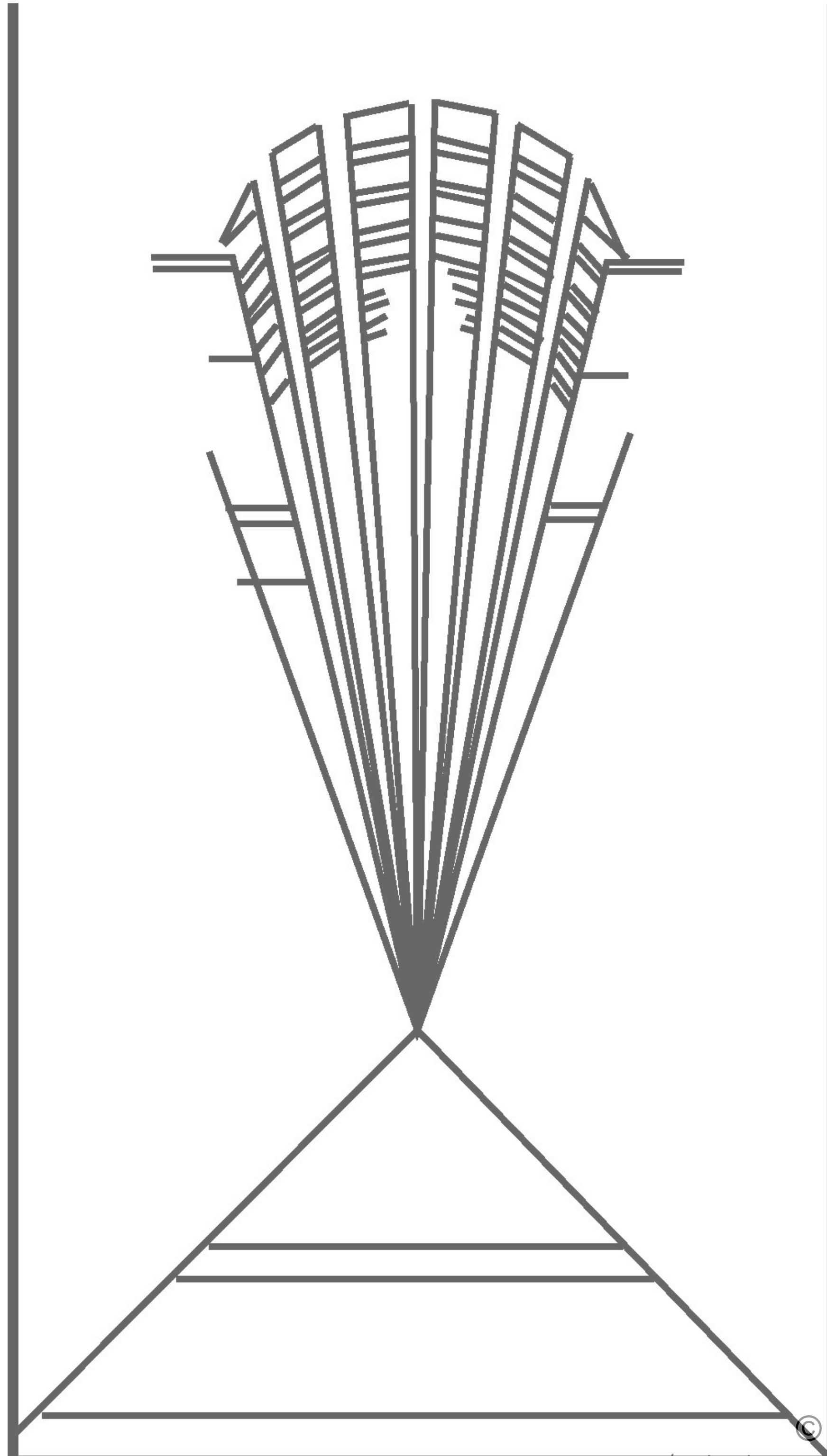
Implications of *w*

- All scalar multiples of a 4-vector are equivalent
- When *w* is not zero, can divide by *w*
  - therefore these points represent "normal" affine points
- When *w* is zero, it's a point at infinity, a.k.a. a direction
  - this is the point where parallel lines intersect
  - can also think of it as the vanishing point

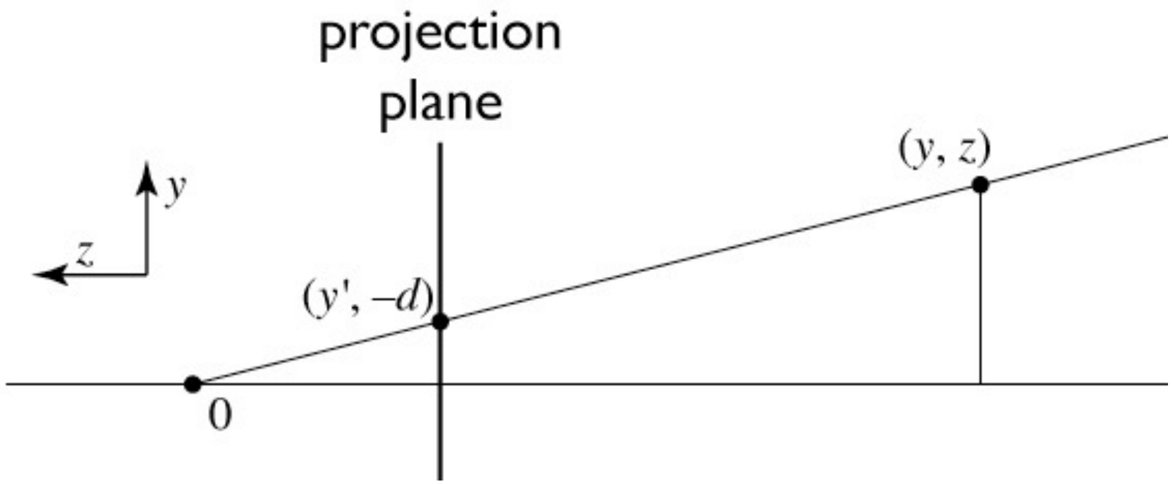
$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \sim \begin{bmatrix} wx \\ wy \\ wz \\ w \end{bmatrix}$$



Masaccio, Trinity, Florence



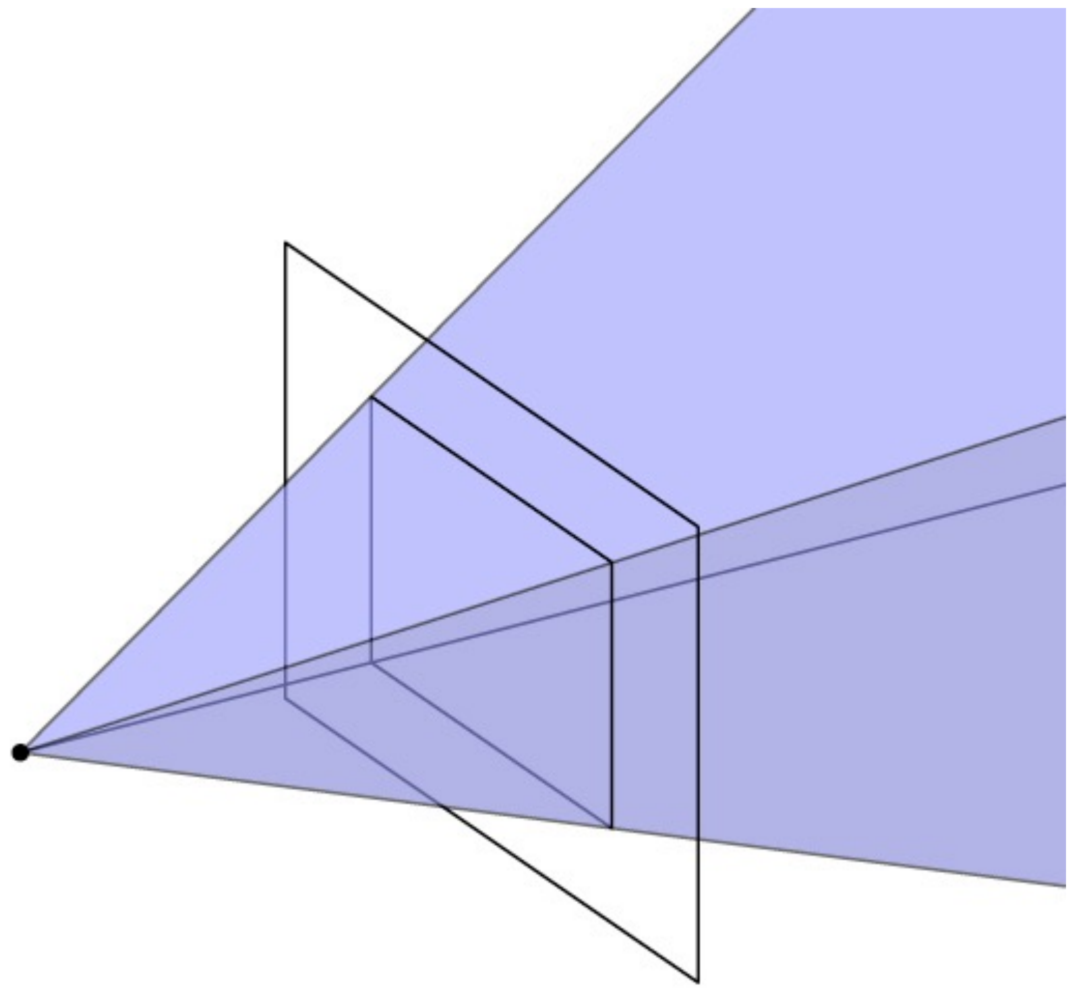
Perspective projection



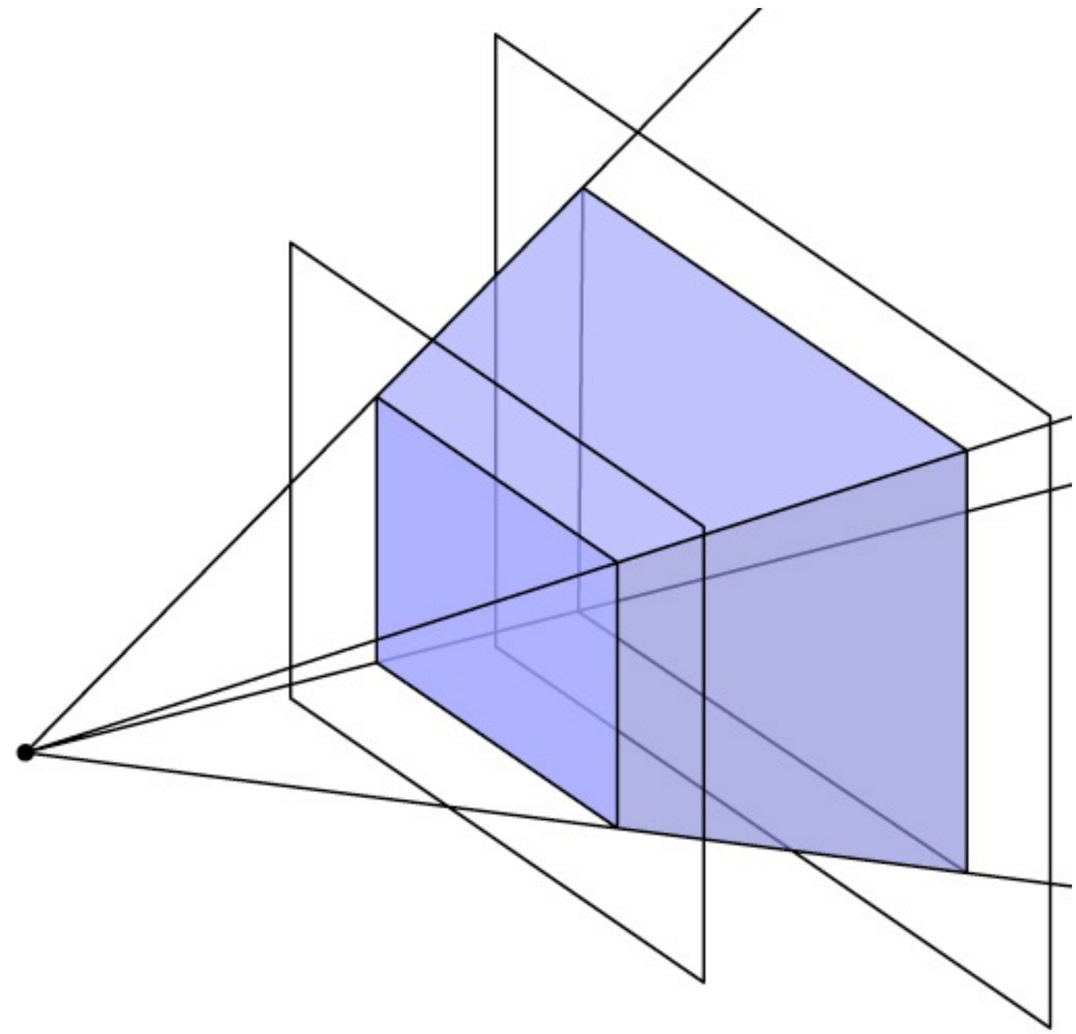
to implement perspective, just move z to w:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} -dx/z \\ -dy/z \\ 1 \end{bmatrix} \sim \begin{bmatrix} dx \\ dy \\ -z \end{bmatrix} = \begin{bmatrix} d & 0 & 0 & 0 \\ 0 & d & 0 & 0 \\ 0 & 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

View volume: perspective



View volume: perspective (clipped)



Carrying depth through perspective

- Perspective has a varying denominator—can't preserve depth!
- Compromise: preserve depth on near and far planes

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} \sim \begin{bmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{z} \\ -z \end{bmatrix} = \begin{bmatrix} d & 0 & 0 & 0 \\ 0 & d & 0 & 0 \\ 0 & 0 & a & b \\ 0 & 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

  – that is, choose a and b so that *z'*(*n*) = *n* and *z'*(*f*) = *f*.

$$\tilde{z}(z) = az + b$$

$$z'(z) = \frac{\tilde{z}}{-z} = \frac{az + b}{-z}$$

$$\text{want } z'(n) = n \text{ and } z'(f) = f$$

$$\text{result: } a = -(n + f) \text{ and } b = nf \text{ (try it)}$$

Official perspective matrix

- Use near plane distance as the projection distance
  - i.e., $d = -n$
- Scale by –1 to have fewer minus signs
  - scaling the matrix does not change the projective transformation

$$\mathbf{P} = \begin{bmatrix} n & 0 & 0 & 0 \\ 0 & n & 0 & 0 \\ 0 & 0 & n+f & -fn \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Perspective projection matrix

- Product of perspective matrix with orth. projection matrix

$$\mathbf{M}_{\text{per}} = \mathbf{M}_{\text{orth}}\mathbf{P}$$

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

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

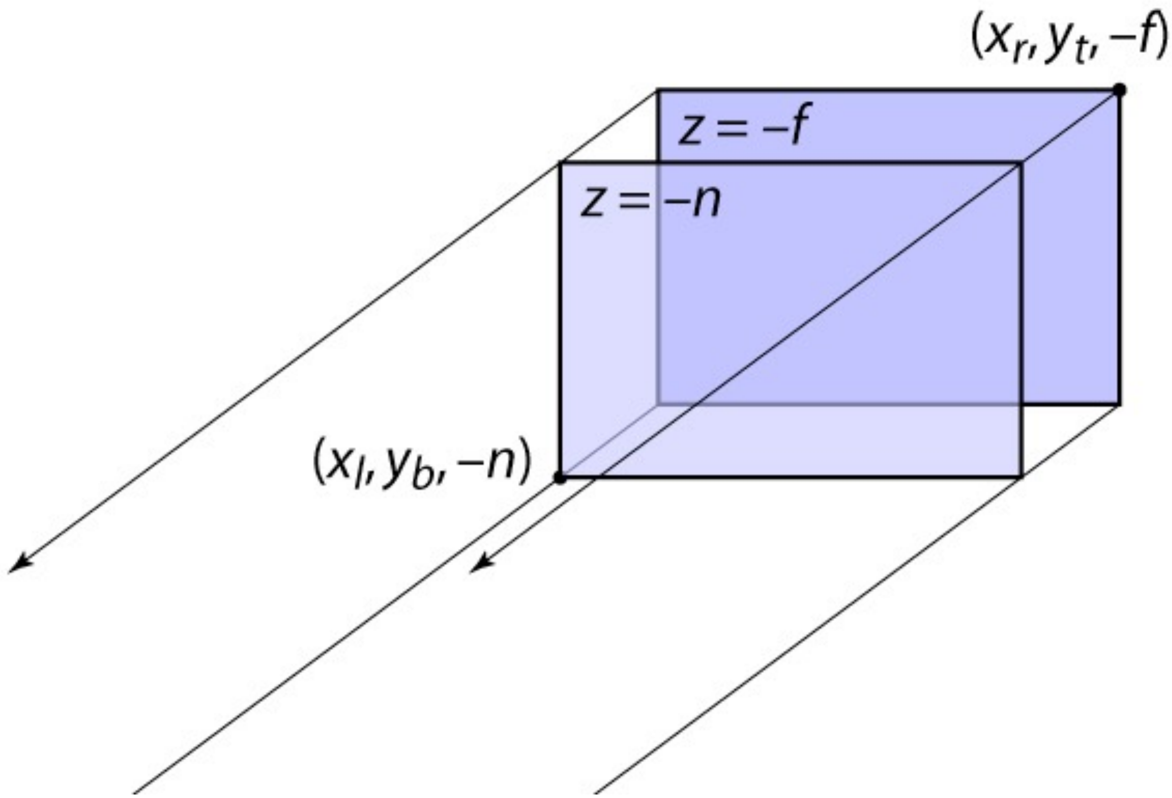
Perspective transformation chain

- Transform into world coords (modeling transform, M_m)
- Transform into eye coords (camera xf., $M_{\text{cam}} = F_c^{-1}$)
- Perspective matrix, P
- Orthographic projection, M_{orth}
- Viewport transform, M_{vp}

$$\mathbf{p}_s = \mathbf{M}_{\text{vp}} \mathbf{M}_{\text{orth}} \mathbf{P} \mathbf{M}_{\text{cam}} \mathbf{M}_{\text{m}} \mathbf{p}_o$$

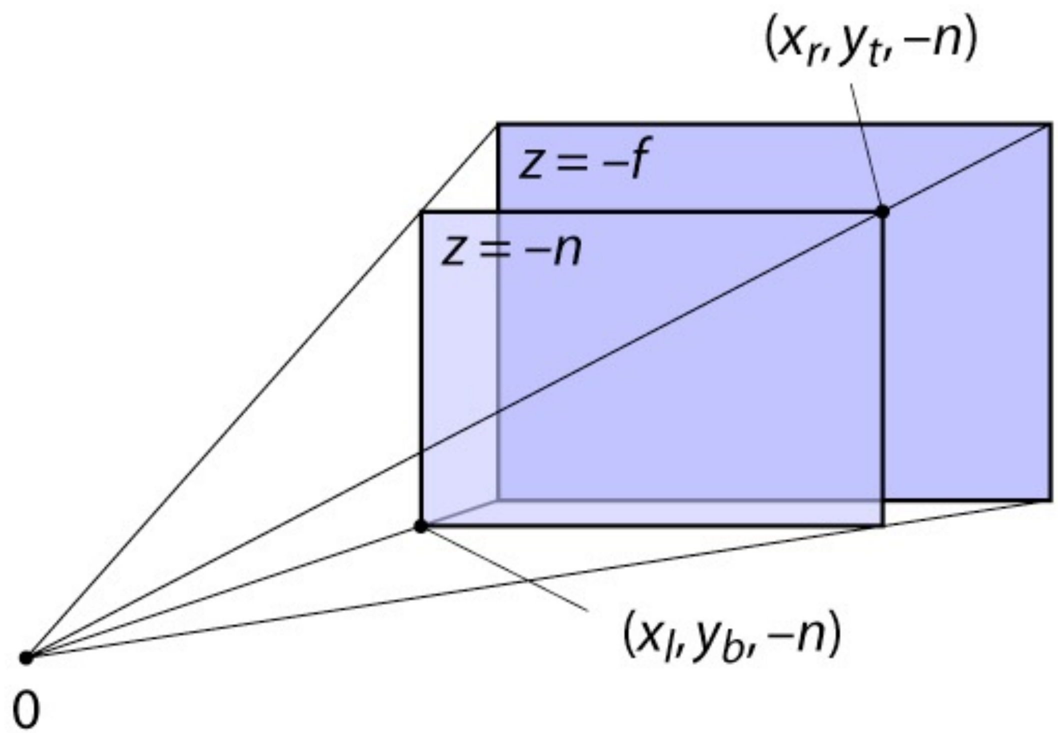
$$\begin{bmatrix} x_s \\ y_s \\ z_c \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{n_x}{2} & 0 & 0 & \frac{n_x - 1}{2} \\ 0 & \frac{n_y}{2} & 0 & \frac{n_y - 1}{2} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{2}{r-l} & 0 & 0 & -\frac{r+l}{r-l} \\ 0 & \frac{2}{t-b} & 0 & -\frac{t+b}{t-b} \\ 0 & 0 & \frac{2}{n-f} & -\frac{n+f}{n-f} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} n & 0 & 0 & 0 \\ 0 & n & 0 & 0 \\ 0 & 0 & n+f & -fn \\ 0 & 0 & 1 & 0 \end{bmatrix} \mathbf{M}_{\text{cam}} \mathbf{M}_{\text{m}} \begin{bmatrix} x_o \\ y_o \\ z_o \\ 1 \end{bmatrix}$$

OpenGL view frustum: orthographic



Note OpenGL puts the near and far planes at $-n$ and $-f$
so that the user can give positive numbers

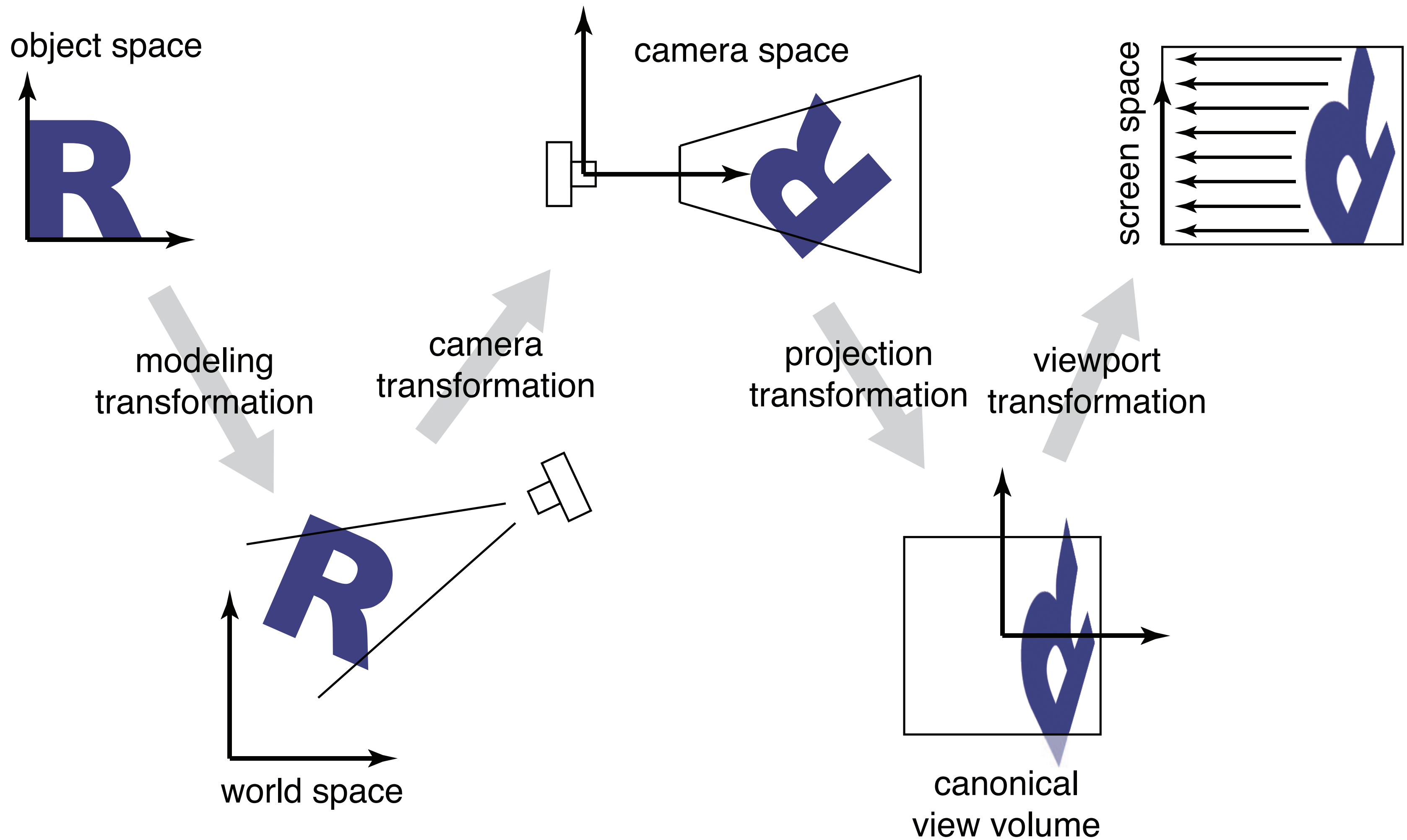
OpenGL view frustum: perspective



Note OpenGL puts the near and far planes at $-n$ and $-f$
so that the user can give positive numbers

Pipeline of transformations

- Standard sequence of transforms



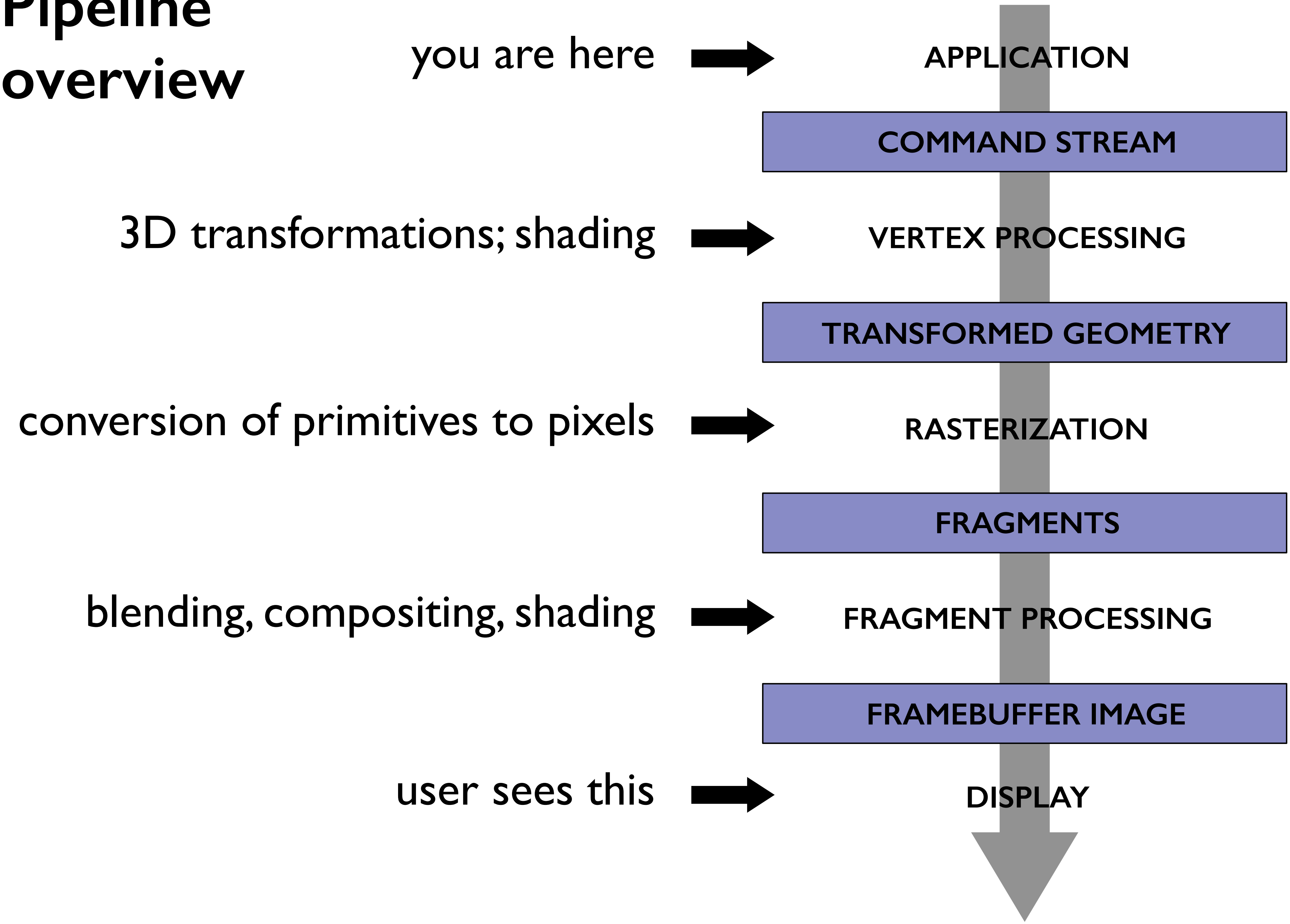
Demo

CS4620/5620

The graphics pipeline

- The standard approach to object-order graphics
- Many versions exist
  - software, e.g. Pixar's REYES architecture
    - many options for quality and flexibility
  - hardware, e.g. graphics cards in PCs
    - amazing performance: millions of triangles per frame
- We'll focus on an abstract version of hardware pipeline

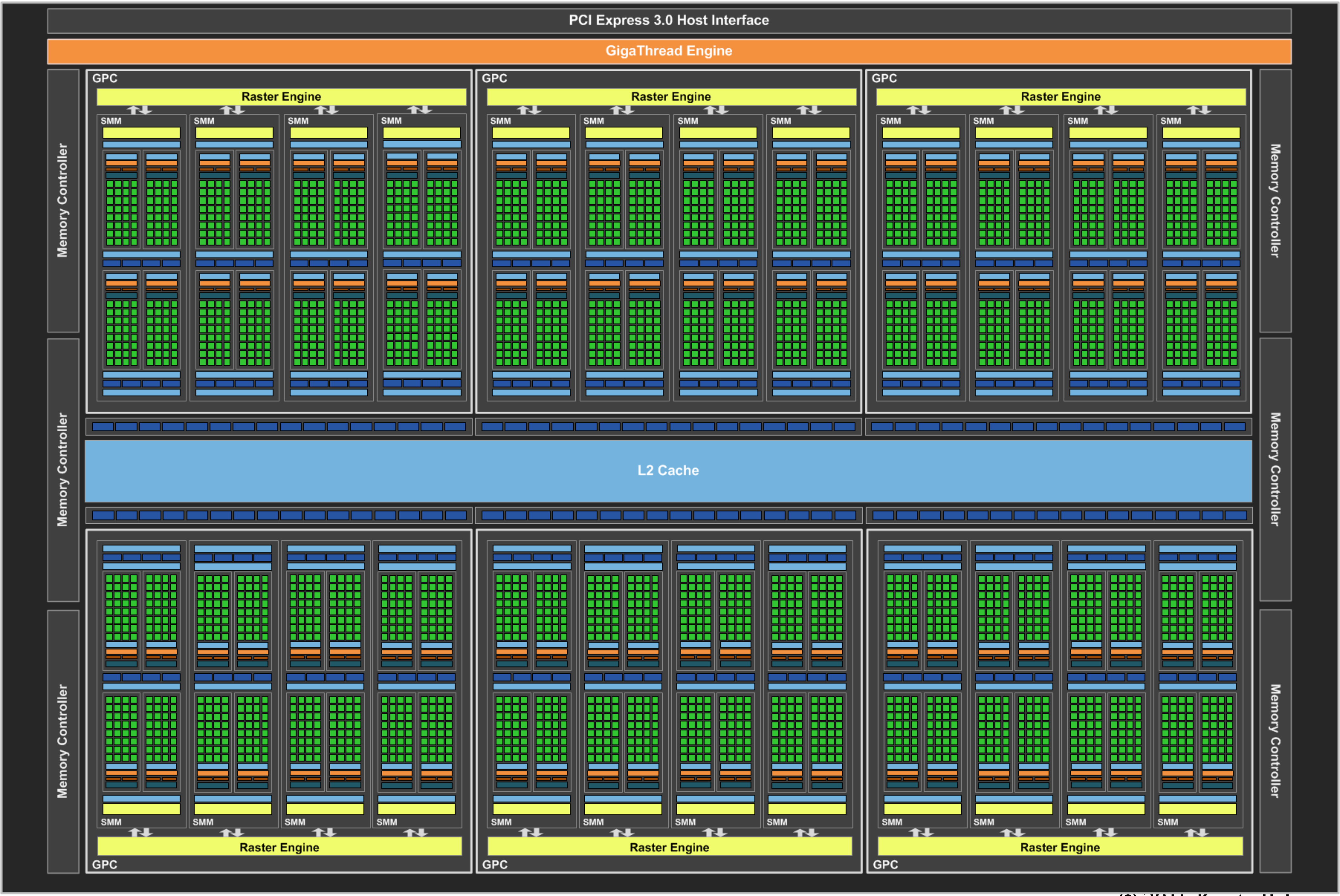
Pipeline overview



The graphics pipeline

- "Pipeline" because of the many stages
  - very parallelizable
  - leads to remarkable performance of graphics cards (many times the flops of the CPU at ~1/3 the clock speed)
  - gigaflops (10 to the 9th power), teraflop (12th power), petaflops (15th power)

- GeForce Titax X, 1 GHz, 3072 CUDA cores


PCI Express 3.0 Host Interface
GigaThread Engine
GPC
Raster Engine
SMM
Memory Controller
L2 Cache

s

Top 10 ranking [edit]

Top 10 positions of the 45th TOP500 on June 2015

Rank	Rmax Rpeak (PFLOPS)	Name	Computer design Processor type, interconnect	Vendor	Site Country, year
1	33.863 54.902	*Tianhe-2*	**NUDT** Xeon E5–2692 + Xeon Phi 31S1P, TH Express-2	NUDT	National Supercomputing Center in Guangzhou China, 2013
2	17.590 27.113	*Titan*	**Cray XK7** Opteron 6274 + Tesla K20X, Cray Gemini Interconnect	Cray Inc.	Oak Ridge National Laboratory United States, 2012
3	17.173 20.133	*Sequoia*	**Blue Gene/Q** PowerPC A2, Custom	IBM	Lawrence Livermore National Laboratory United States, 2013
4	10.510 11.280	*K computer*	**RIKEN** SPARC64 VIIIfx, Tofu	Fujitsu	RIKEN Japan, 2011
5	8.586 10.066	*Mira*	**Blue Gene/Q** PowerPC A2, Custom	IBM	Argonne National Laboratory United States, 2013
6	6.271 7.779	*Piz Daint*	**Cray XC30** Xeon E5–2670 + Tesla K20X, Aries	Cray Inc.	Swiss National Supercomputing Centre Switzerland, 2013
7	5.537 7.235	*Shaheen II*	**Cray XC40** Xeon E5–2698v3, Aries	Cray Inc.	King Abdullah University of Science and Technology Saudi Arabia, 2015
8	5.168 8.520	*Stampede*	**PowerEdge C8220** Xeon E5–2680 + Xeon Phi, Infiniband	Dell	Texas Advanced Computing Center United States, 2013
9	5.008 5.872	*JUQUEEN*	**Blue Gene/Q** PowerPC A2, Custom	IBM	Forschungszentrum Jülich Germany, 2013
10	4.293 5.033	*Vulcan*	**Blue Gene/Q** PowerPC A2, Custom	IBM	Lawrence Livermore National Laboratory United States, 2013

Legend:

Supercomputers

- Tianhe-2: 32,000 Xeon + 48,000 Xeon Phi
  - 33 PFlops

- Current supercompute
  - IBM Sequoia (petascale) Blue Gene (16 petaflops)

Titan (supercomputer)

Active	Became operational October 29, 2012
Sponsors	US DOE and NOAA (<10%)
Operators	Cray Inc.
Location	Oak Ridge National Laboratory
Architecture	18,688 AMD Opteron 6274 16-core CPUs 18,688 Nvidia Tesla K20X GPUs
Power	8.2 MW
Operating system	Cray Linux Environment
Space	404 m^2 (4352 ft^2)
Memory	693.5 TiB (584 TiB CPU and 109.5 TiB GPU)
Storage	40 PB, 1.4 TB/s IO Lustre filesystem
Speed	17.59 petaFLOPS (LINPACK) 27 petaFLOPS theoretical peak
Cost	$97 million
Ranking	TOP500: #2, June 2013[1]
Purpose	Scientific research
Legacy	Ranked 1 on TOP500 when built. First GPU based supercomputer to perform over 10 petaFLOPS
Web site	www.olcf.ornl.gov/titan/

