Projective Transformations

University of Victoria | Computer Science

Viewing Transformation

object space

camera space

screen space

model

camera

projection

viewport

world space

canonical
view volume

University of Victoria | Computer Science

Orthographic Projection

camera space

projection

canonical

view volume

y

z

x

(l,b,n)

(r,t,f)

$$\mathbf{M}_{orth} = \begin{bmatrix} \frac{2}{r-l} & 0 & 0 & -\frac{r+l}{r-l} \\ 0 & \frac{2}{t-b} & 0 & -\frac{t+b}{t-b} \\ 0 & 0 & \frac{2}{n-f} & -\frac{n+f}{n-f} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$
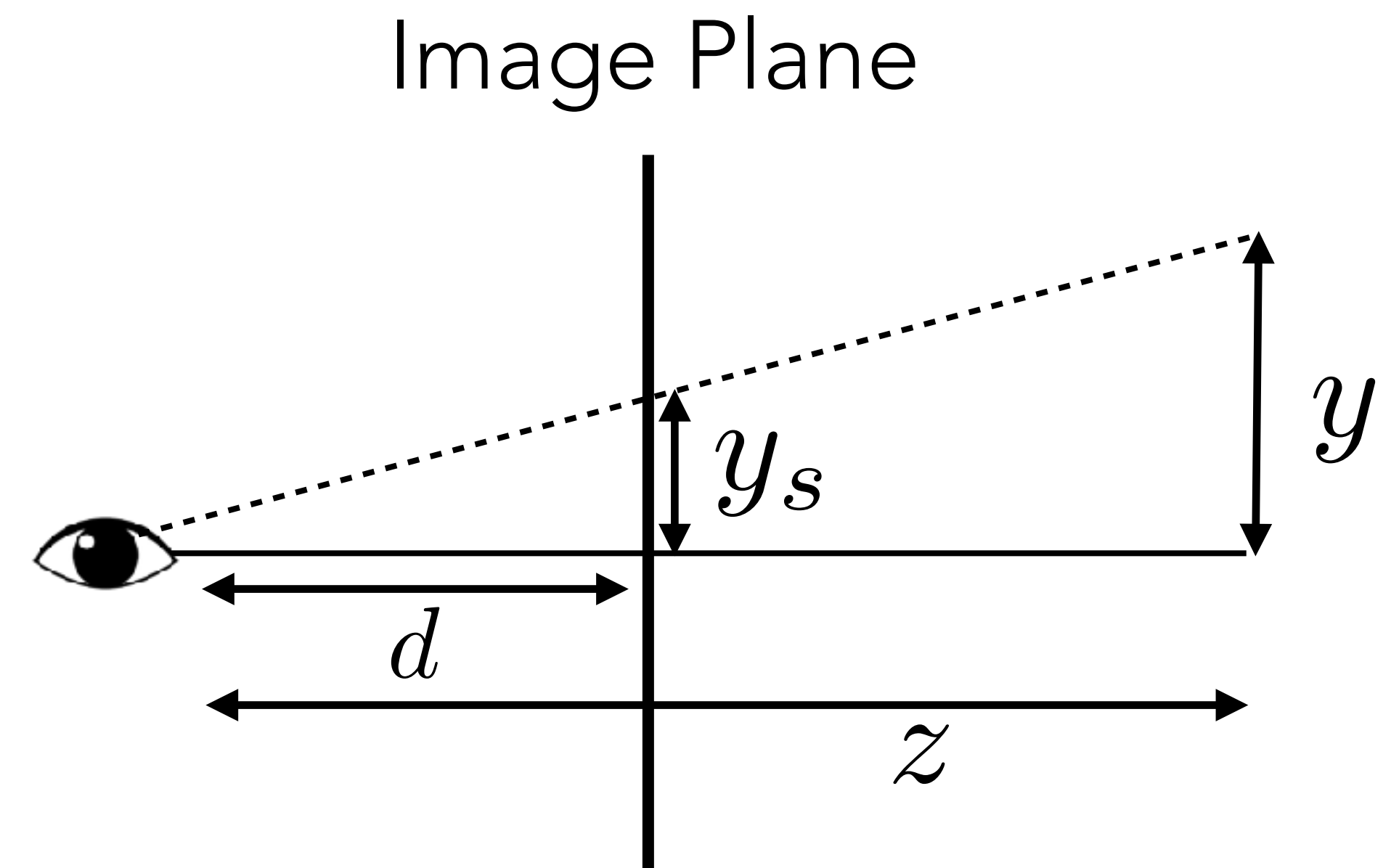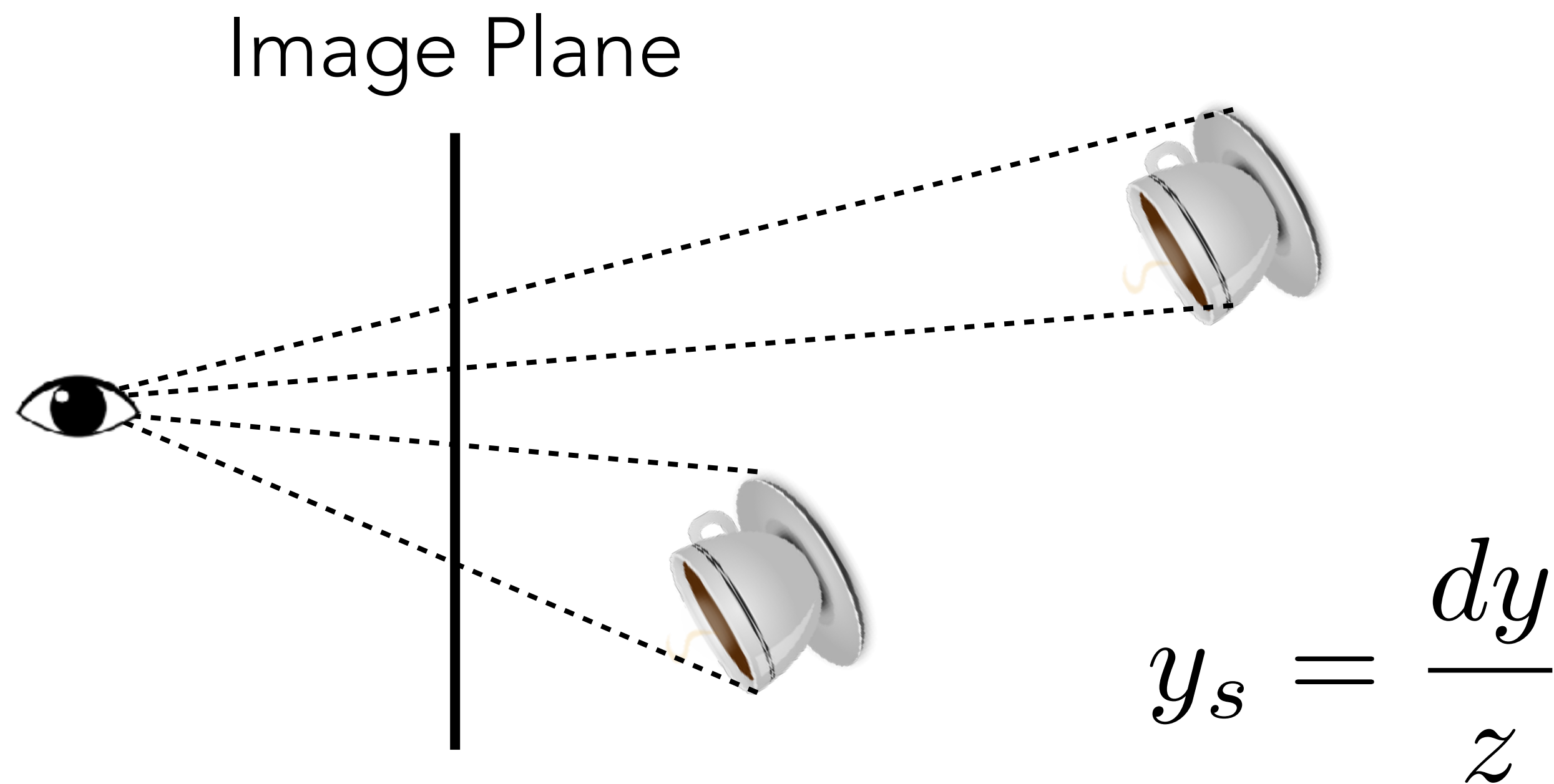
University of Victoria | Computer Science
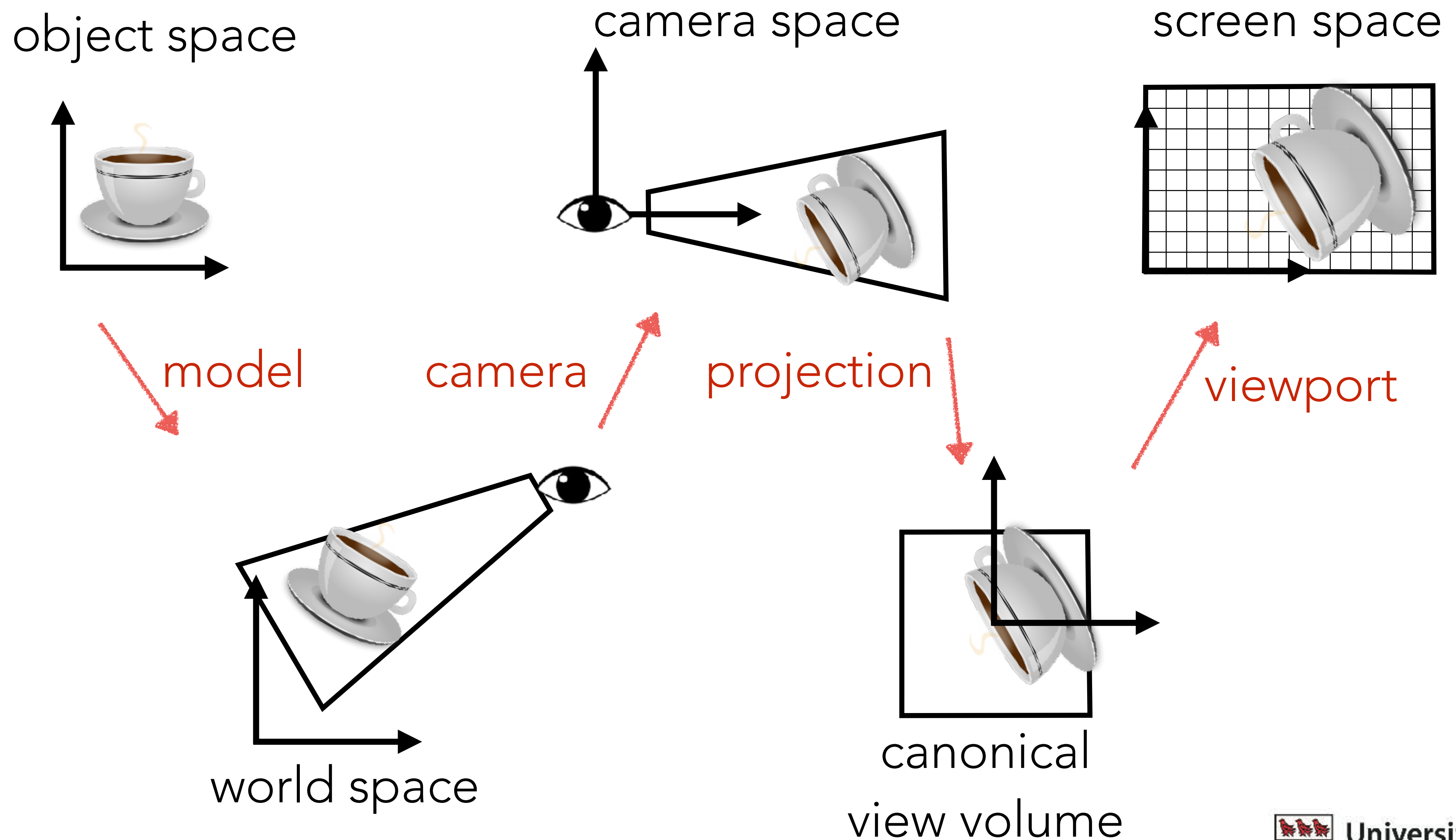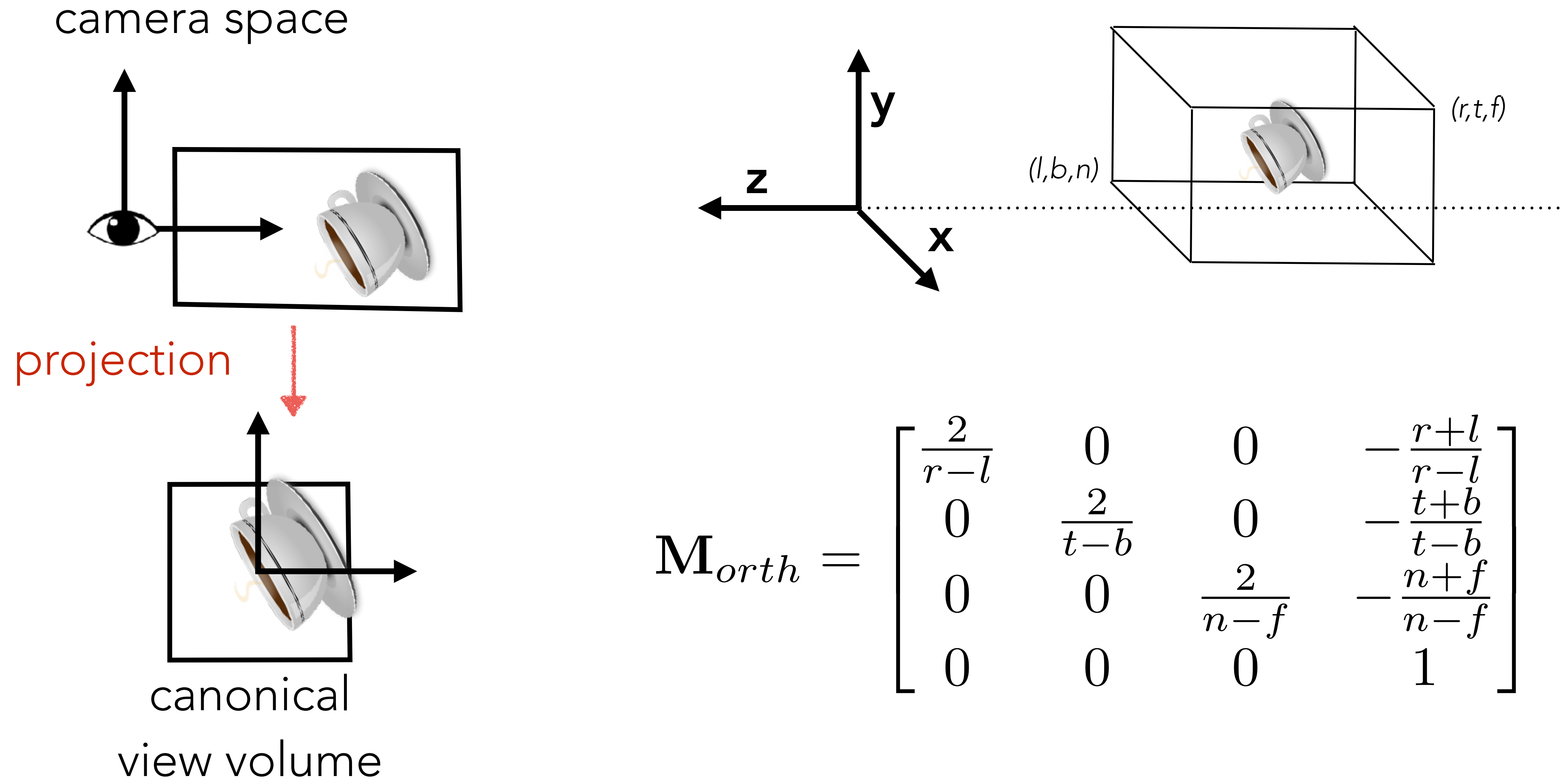
Perspective Projection

- In Orthographic projection, the size of the objects does not change with distance

- In Perspective projection, the objects that are far away look smaller

Image Plane

Image Plane

$$y_s = \frac{dy}{z}$$

y_s

y

d

z

University of Victoria | Computer Science

Divisions in Matrix Form

- We would like to reuse the matrix machinery that we built in the previous lectures

- How do we encode divisions?

$$y_s = \frac{dy}{z}$$

- We extend homogeneous coordinates

Image Plane

y_s

y

d

z

University of Victoria | Computer Science
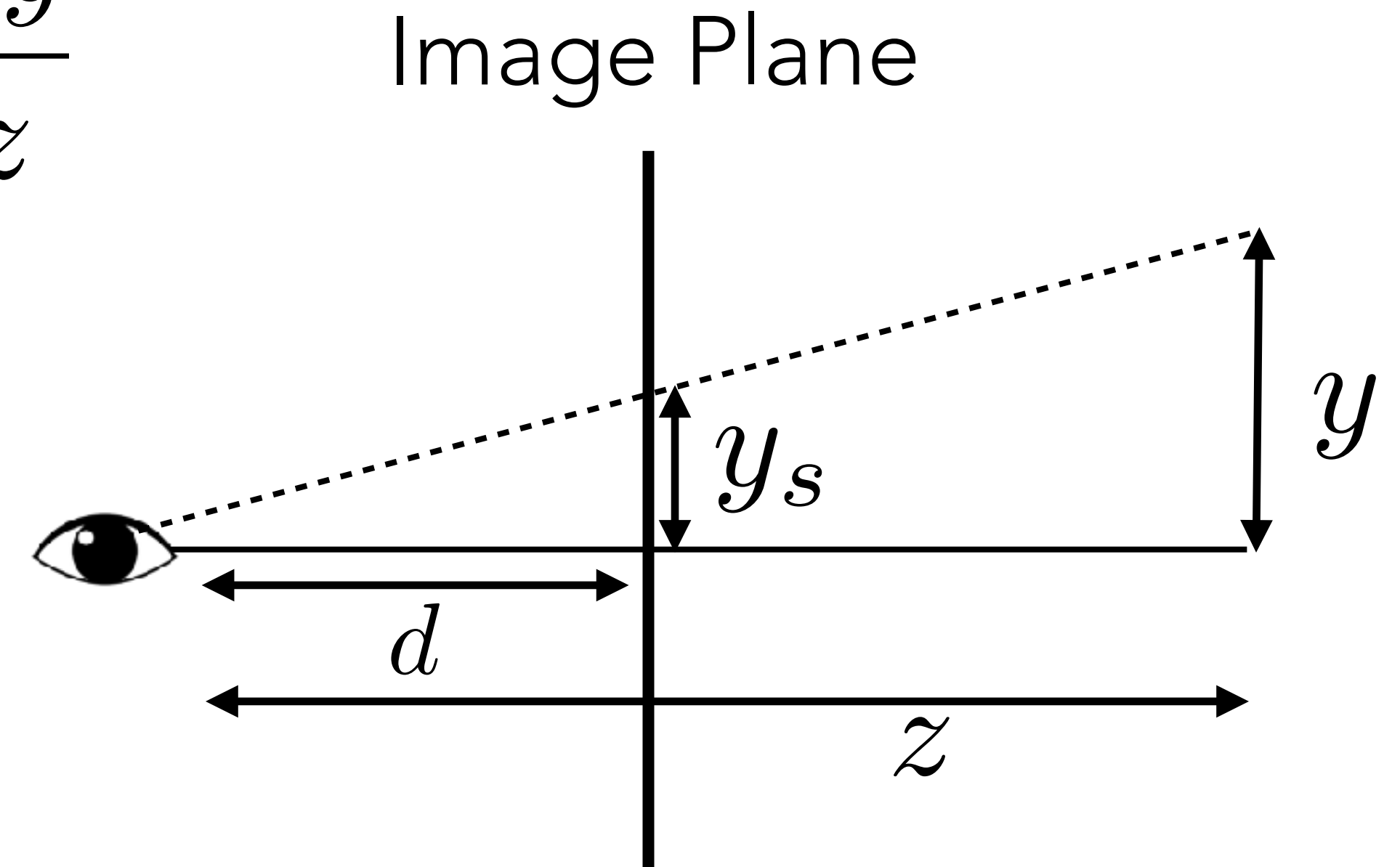
Until now…

- What do we have left?

$$\begin{pmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} a_1 x + b_1 y + c_1 \\ a_2 x + b_2 y + c_2 \\ 1 \end{pmatrix}$$

- We can use the last row of the transformation:

$$\begin{pmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ e & f & g \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} a_1 x + b_1 y + c_1 \\ a_2 x + b_2 y + c_2 \\ ex + fy + g \end{pmatrix} \sim \begin{pmatrix} \frac{a_1 x + b_1 y + c_1}{ex + fy + g} \\ \frac{a_2 x + b_2 y + c_2}{ex + fy + g} \\ 1 \end{pmatrix}$$

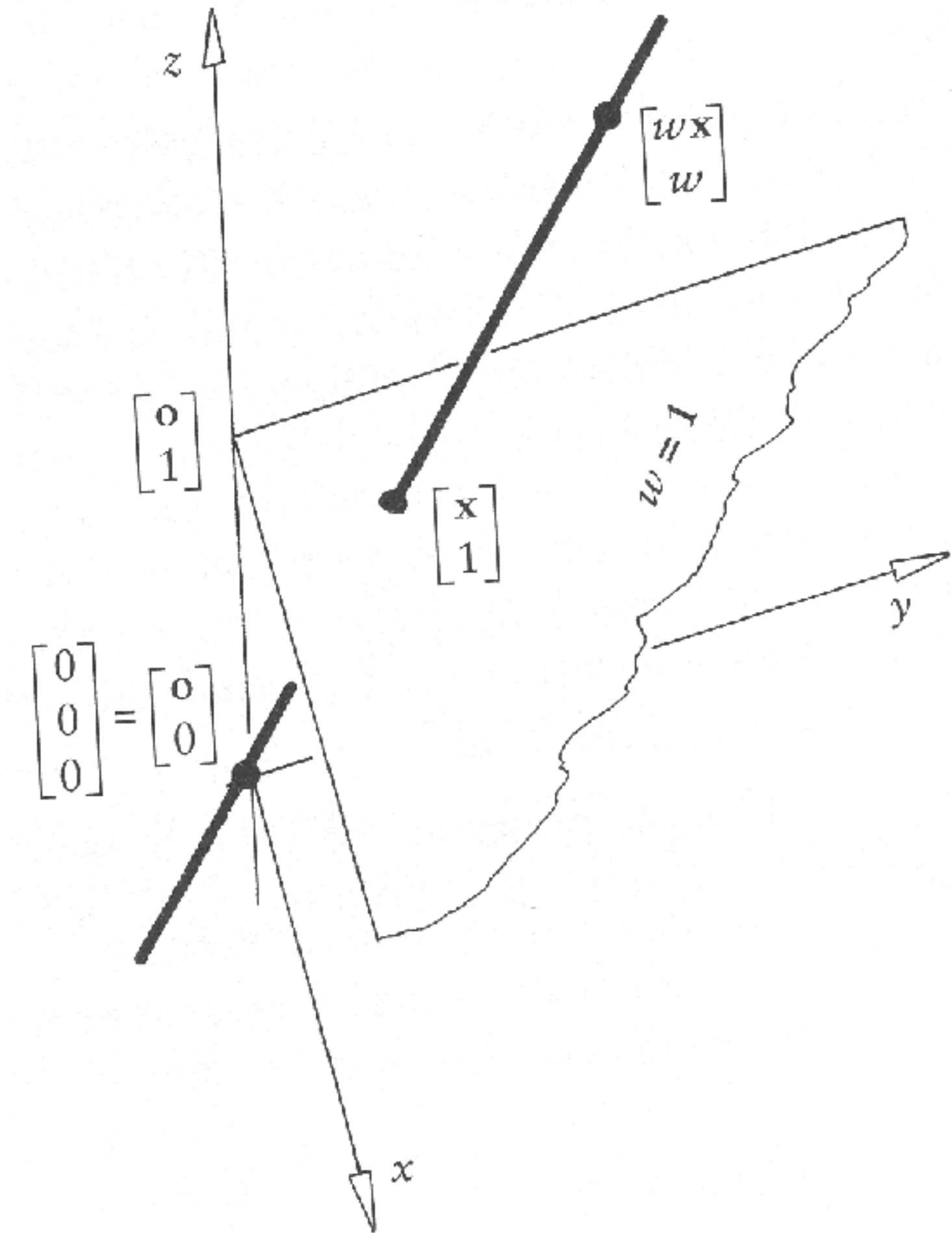University of Victoria | Computer Science

Intuition

- Purely algebraic:

$$\begin{pmatrix} x \\ y \\ w \end{pmatrix} \sim \begin{pmatrix} x/w \\ y/w \\ 1 \end{pmatrix}$$

- Or as a projection, where each line is identified by a point on the plane z=1

- Note that in this case, you can think of it as a transformation in a space with one more dimension

Projective Transformation

- A transformation of this form is called a *projective transformation* (or a homography)

- The points are represented in *homogeneous coordinates*

$$\begin{pmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ e & f & g \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} a_1 x + b_1 y + c_1 \\ a_2 x + b_2 y + c_2 \\ ex + fy + g \end{pmatrix} \sim \begin{pmatrix} \frac{a_1 x + b_1 y + c_1}{ex + fy + g} \\ \frac{a_2 x + b_2 y + c_2}{ex + fy + g} \\ 1 \end{pmatrix}$$

University of Victoria | Computer Science

Example

$$\mathbf{M} = \begin{pmatrix} 2 & 0 & -1 \\ 0 & 3 & 0 \\ 0 & 2/3 & 1/3 \end{pmatrix}$$

- It transforms a square into a quadrilateral — note that straight lines are preserved, but parallel lines are not!

- Note that you can use homogeneous coordinates for as many transformations as you want, only when you need the cartesian representation you have to normalize
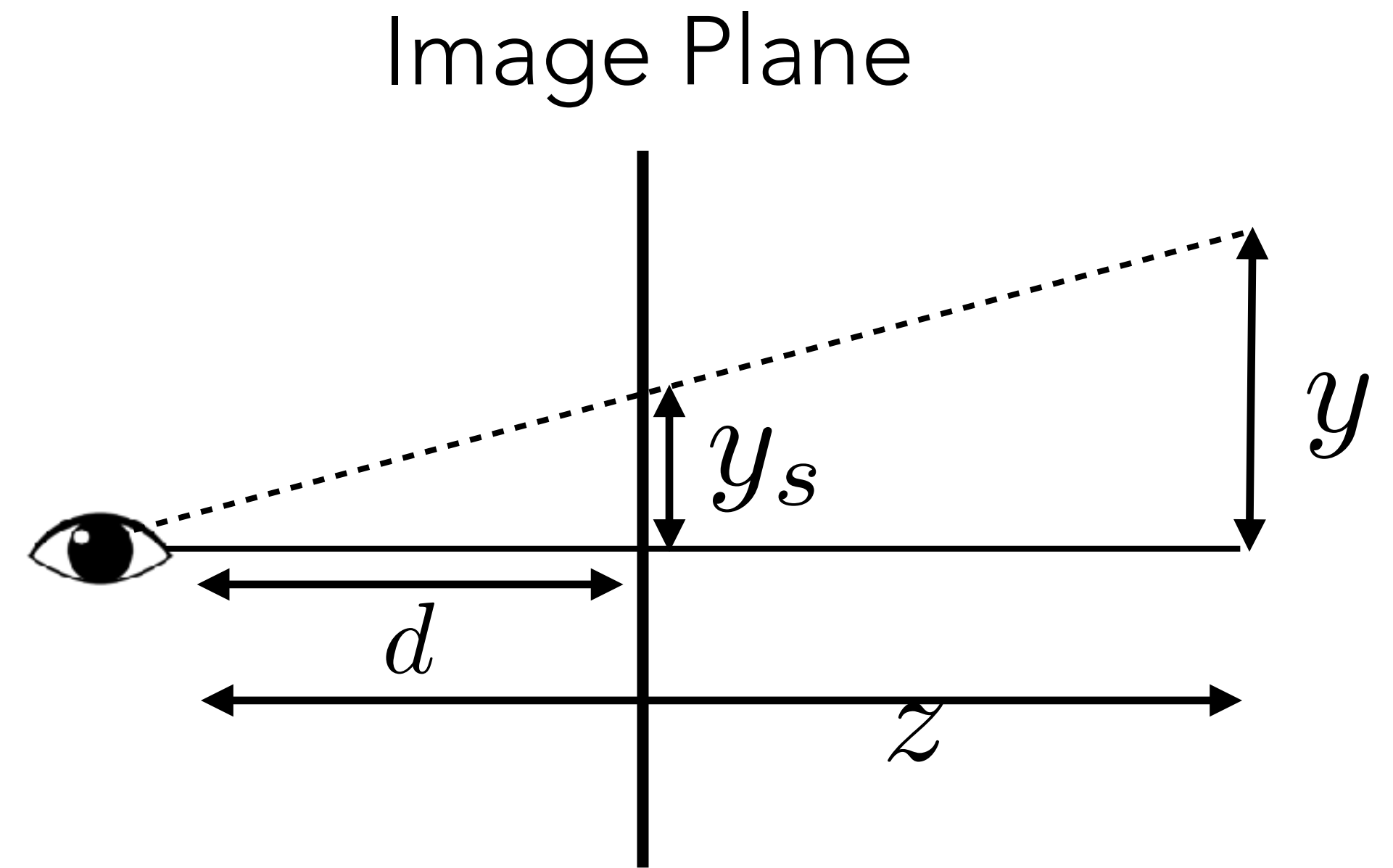
1,1

0,0

1,3

0,0   3,0
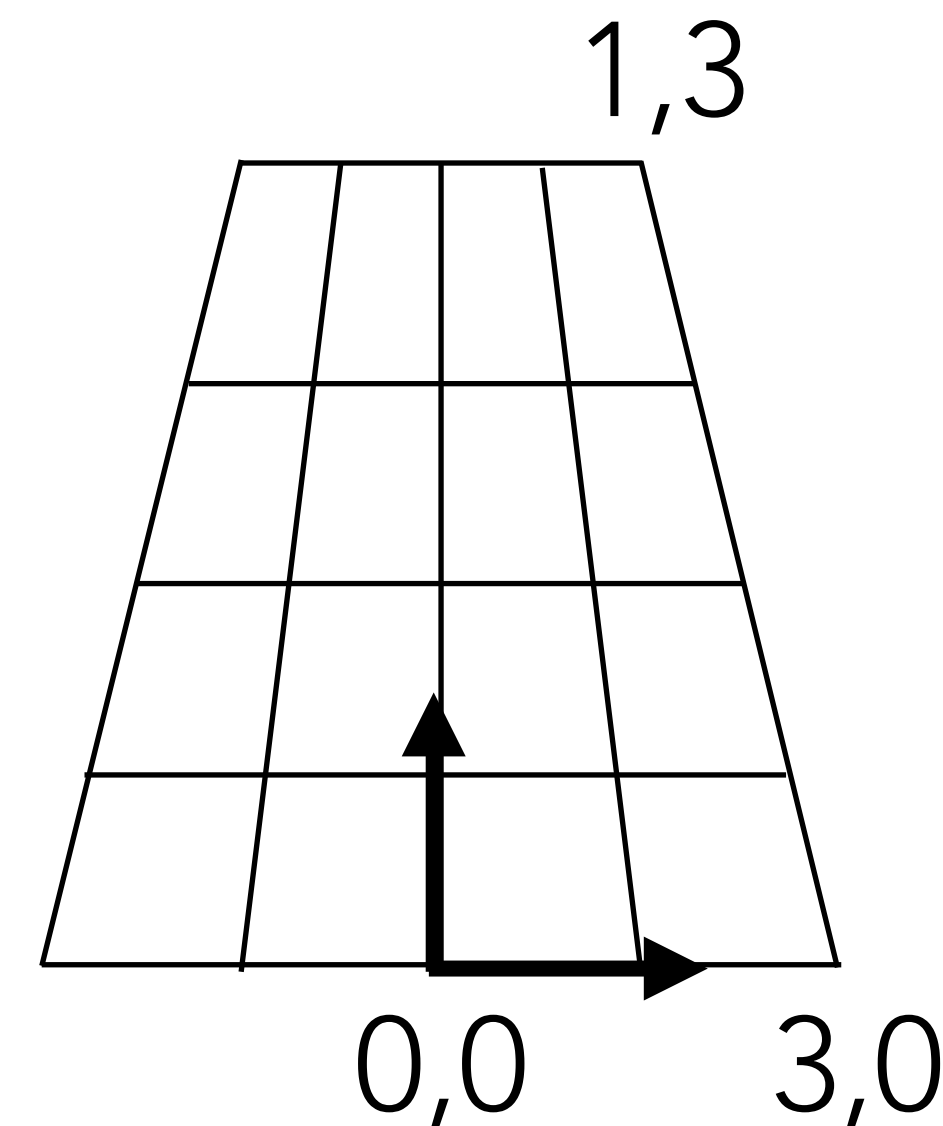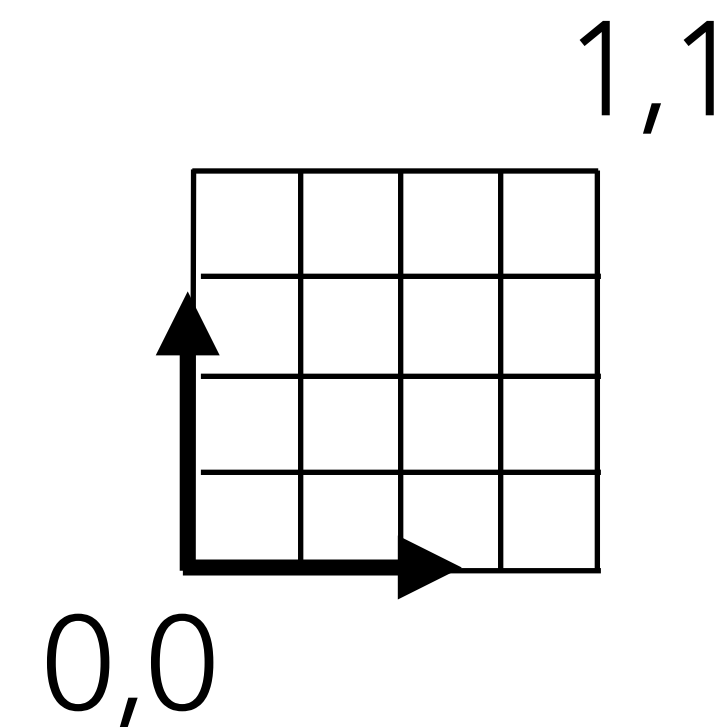
Perspective Projection

- Perspective projection is easily implementable using this machinery
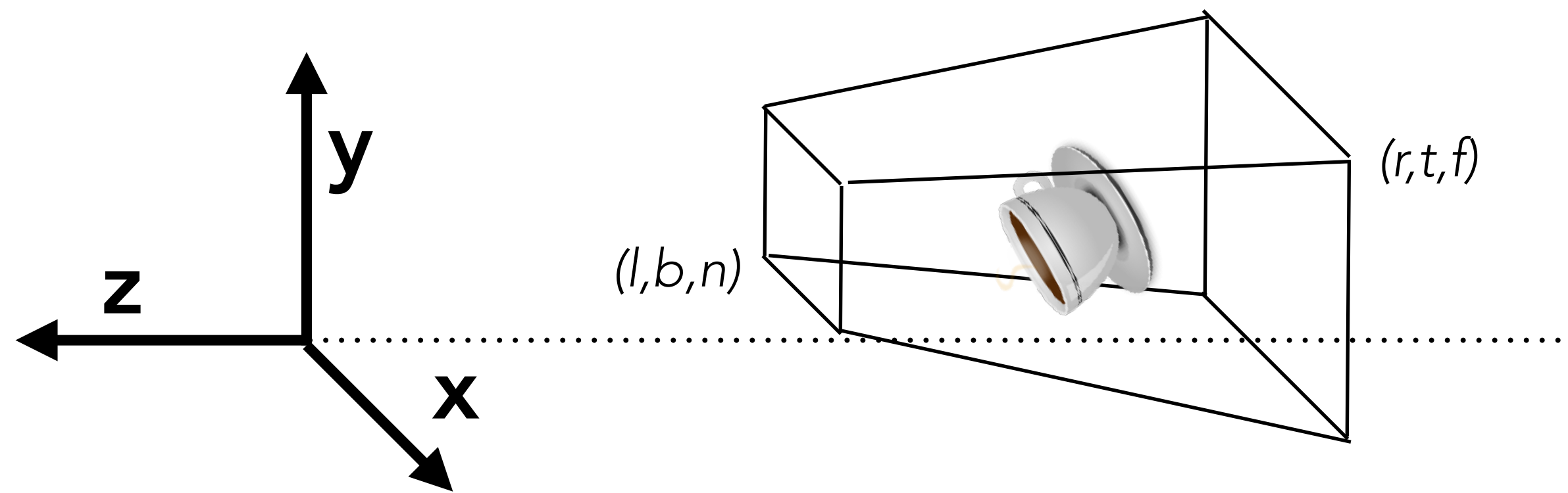
$$y_s = \frac{dy}{z}$$

$$\begin{pmatrix} y_s \\ 1 \end{pmatrix} \sim \begin{pmatrix} d & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} y \\ z \\ 1 \end{pmatrix}$$

Image Plane

y_s

y

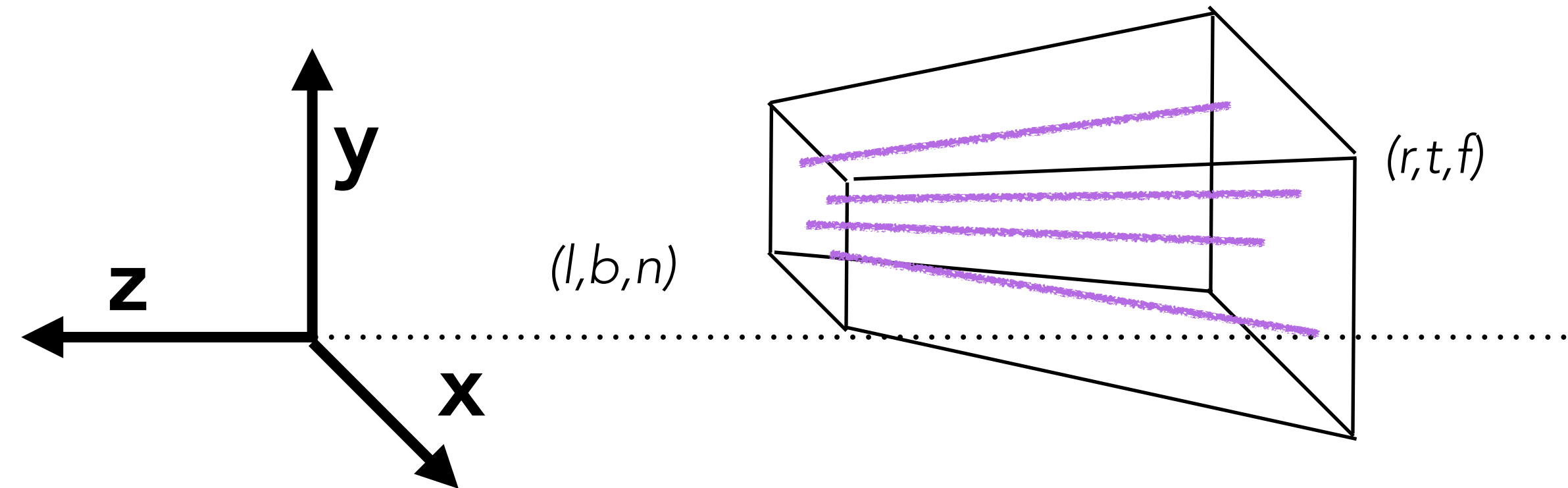d

z

University of Victoria | Computer Science

Perspective Projection

- We will use the same conventions that we used for orthographic:

  - Camera at the origin, pointing negative z

  - We scale x, y and "bring along" the z

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

Effect on the points

y

z

x

(l,b,n)

(r,t,f)

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

$$\mathbf{P} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} nx \\ ny \\ (n+f)z - fn \\ z \end{pmatrix} \sim \begin{pmatrix} \frac{nx}{z} \\ \frac{ny}{z} \\ n+f-\frac{fn}{z} \\ 1 \end{pmatrix}$$

University of Victoria | Computer Science

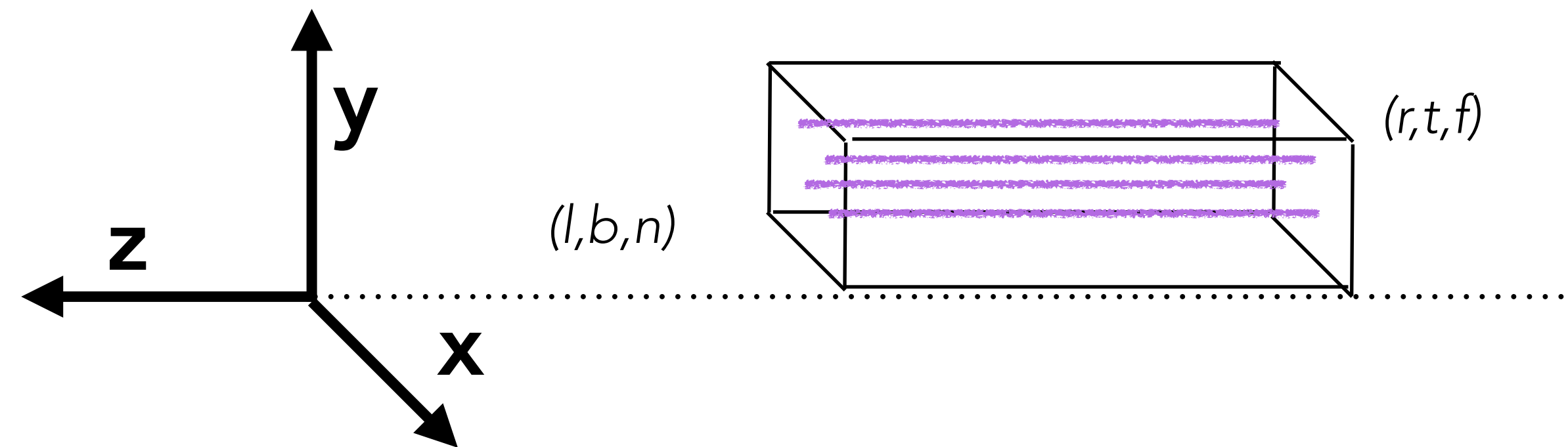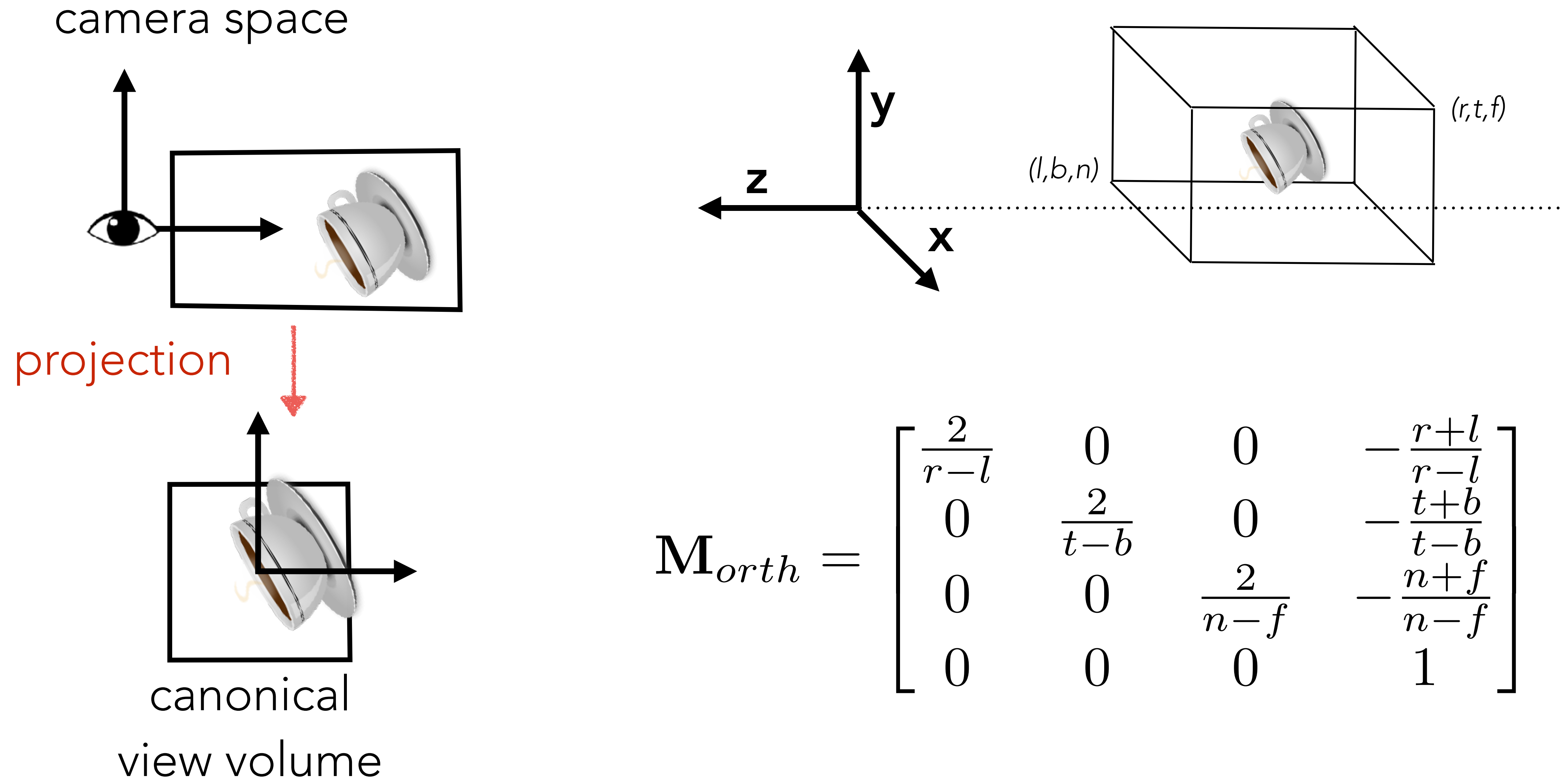# Effect on the points

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

$$\mathbf{P} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} nx \\ ny \\ (n+f)z - fn \\ z \end{pmatrix} \sim \begin{pmatrix} \frac{nx}{z} \\ \frac{ny}{z} \\ n+f-\frac{fn}{z} \\ 1 \end{pmatrix}$$

Orthographic Projection

camera space

projection

canonical

view volume

y

z

x

(r,t,f)

(l,b,n)

$$\mathbf{M}_{orth} = \begin{bmatrix} \dfrac{2}{r-l} & 0 & 0 & -\dfrac{r+l}{r-l} \\ 0 & \dfrac{2}{t-b} & 0 & -\dfrac{t+b}{t-b} \\ 0 & 0 & \dfrac{2}{n-f} & -\dfrac{n+f}{n-f} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$
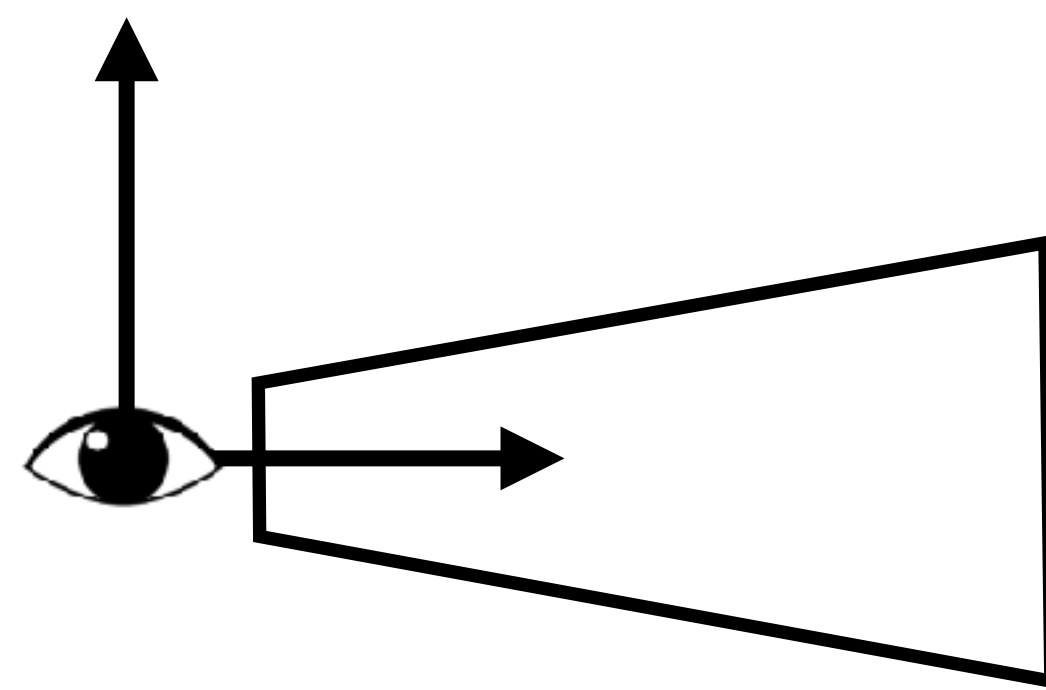
University of Victoria | Computer Science
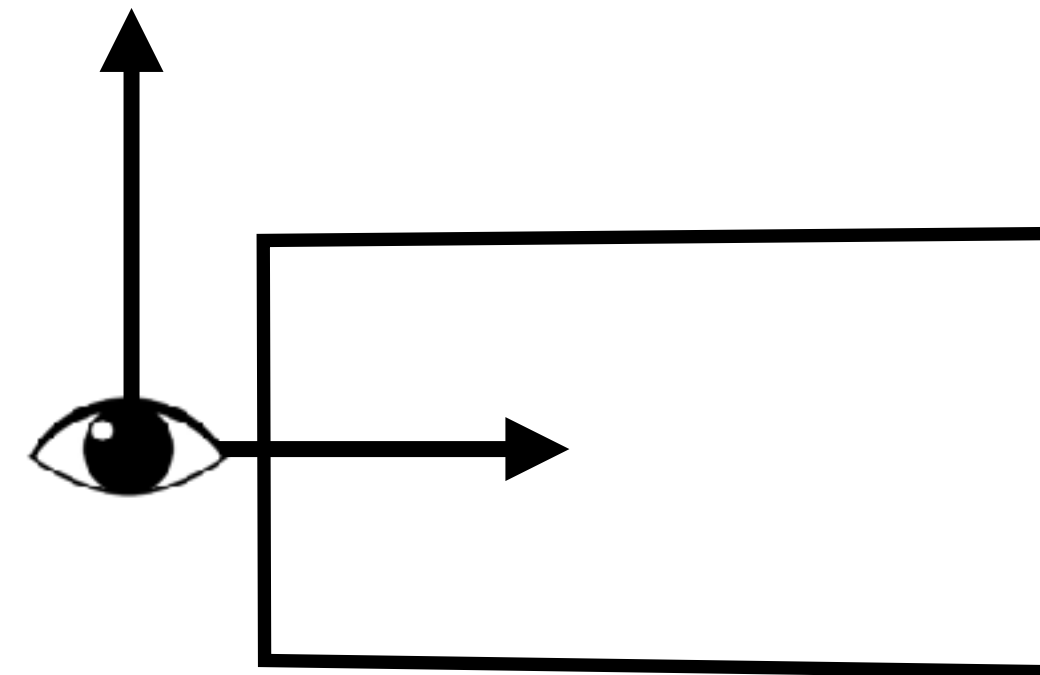
Complete Perspective Transformation

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

$$\mathbf{M}_{orth} = \begin{bmatrix} \frac{2}{r-l} & 0 & 0 & -\frac{r+l}{r-l} \\ 0 & \frac{2}{t-b} & 0 & -\frac{t+b}{t-b} \\ 0 & 0 & \frac{2}{n-f} & -\frac{n+f}{n-f} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$
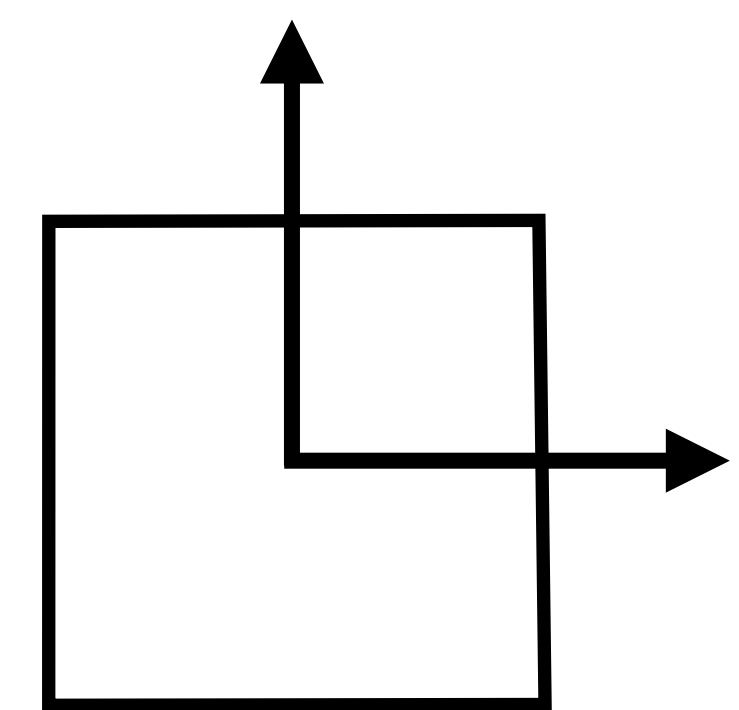
canonical

camera space

view volume

$\mathbf{P}$

$\mathbf{M_{orth}}$

University of Victoria | Computer Science

Parameters?

- How to set the parameters of the transformation?

- If we look at the center of the center of the window then the barycenter of the front back should be at (0,0,f)

- If we want no distortion on the image we need to keep a fixed aspect ratio:

  - width/height = r/t (width and height are the size in pixels of the final image)

- There is only one degree of freedom left, the field of view angle θ:

  - $tan\dfrac{\theta}{2} = \dfrac{t}{|n|}$

- The parameters can thus by found by fixing n and θ. You can then compute t and consequently all the other parameters needed to construct the transformation
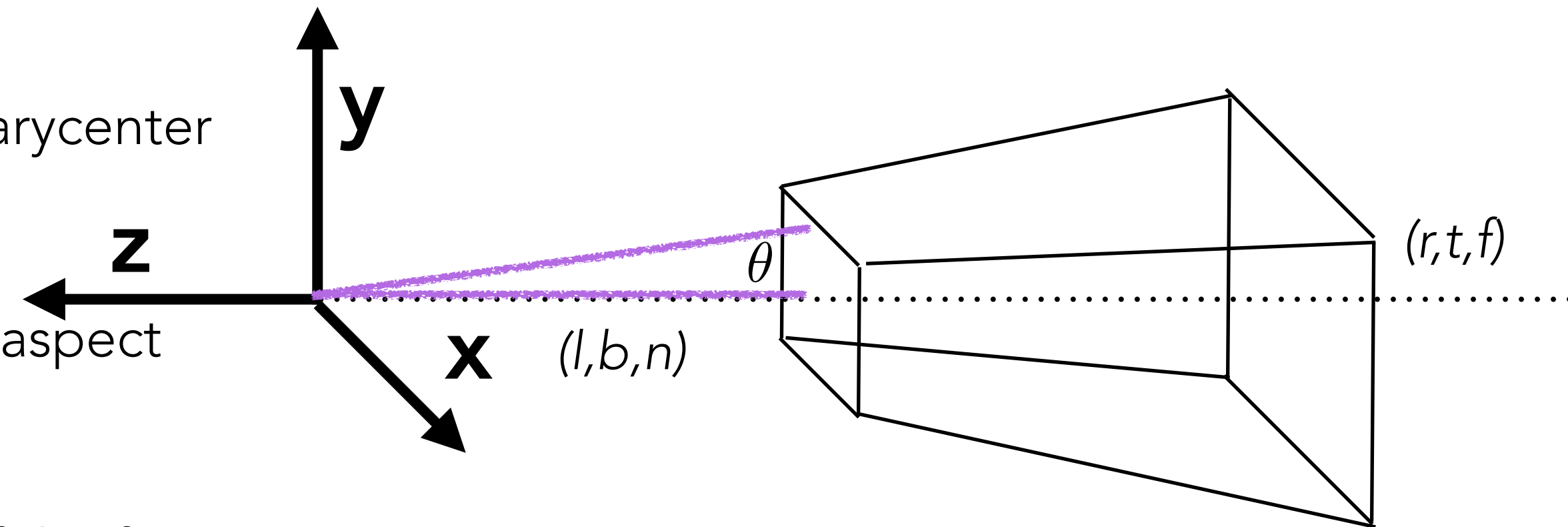
References

Fundamentals of Computer Graphics, Fourth Edition
4th Edition **by Steve Marschner, Peter Shirley**

Chapter 7

University of Victoria | Computer Science