Today's Topics

6. Transformations in 3D

7. 3D Viewing

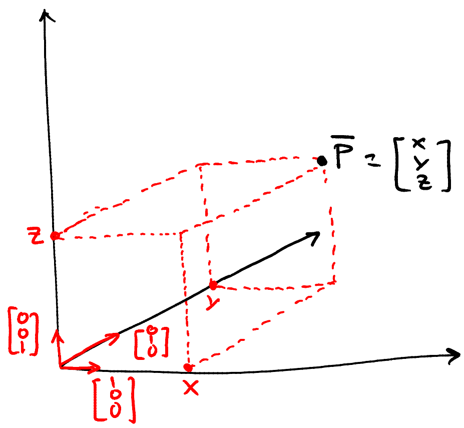
Next week: There will be only ONE tutorial, in the regular lecture room (BA1170)

Topic 6:

3D Transformations

- Homogeneous coordinates in 3D
- Homogeneous 3D transformations
- Affine transformations & rotations in 3D

Representing Points by Euclidean 3D Coords



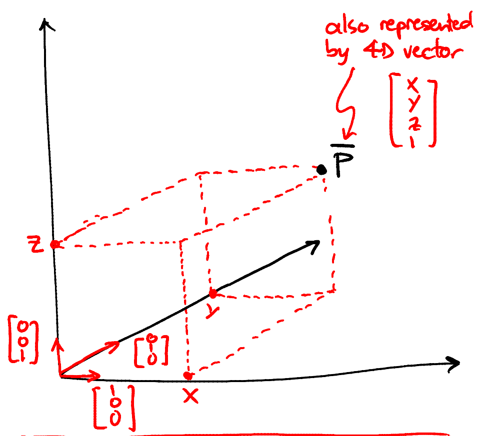
- "Standard" (Euclidean) representation of a point $\bar{p}$:

$$\bar{p} = \underbrace{x\begin{bmatrix}1\\0\\0\end{bmatrix} + y\begin{bmatrix}0\\1\\0\end{bmatrix} + z\begin{bmatrix}0\\0\\1\end{bmatrix}}_{\text{basis vectors}} = \underbrace{\begin{bmatrix}x\\y\\z\end{bmatrix}}_{\text{Euclidean coordinates}}$$

basis vectors

Euclidean coordinates

Euclidean Coords ⇒ Homogeneous Coords



- "Standard" (Euclidean) representation of a point $\bar{p}$:

$$\bar{p} = x\begin{bmatrix}1\\0\\0\end{bmatrix} + y\begin{bmatrix}0\\1\\0\end{bmatrix} + z\begin{bmatrix}0\\0\\1\end{bmatrix} = \begin{bmatrix}x\\y\\z\end{bmatrix}$$

- Homogeneous (a.k.a. Projective) representation of $\bar{p}$

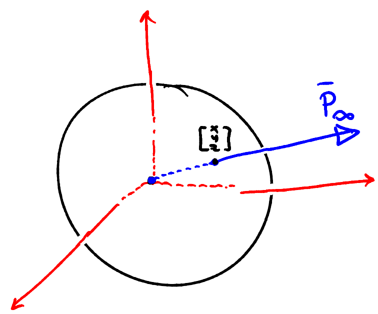
Converting from homogeneous to Euclidean 3D Coords

$$\begin{bmatrix}x\\y\\z\\w\end{bmatrix} \longrightarrow \begin{bmatrix}x/w\\y/w\\z/w\end{bmatrix}$$

3D coordinates → homogeneous 3D coordinates

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

Points at ∞ in Homogeneous Coordinates

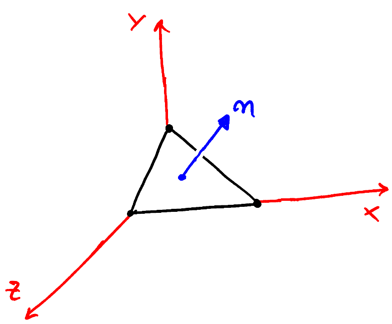


- A point at infinity does not represent a physical location on the plane
- It represents a direction

$$\bar{P}_\infty = \begin{bmatrix} x \\ y \\ z \\ 0 \end{bmatrix}$$ i.e. point at ∞ in direction of 3D vector $\begin{bmatrix} x \\ y \\ z \end{bmatrix}$

Points at infinity have their last coordinate equal to zero

Plane Equation in Homogeneous Coordinates



- The equation of a plane

$$ax + by + cz + d = 0$$

plane parameters

- In homogeneous coordinates

$$\begin{bmatrix} a & b & c & d \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = 0$$

first 3 coordinates represent direction of the plane's normal, n

or $\bar{\ell}^T \cdot \bar{p} = 0$

vector holding plane parameters

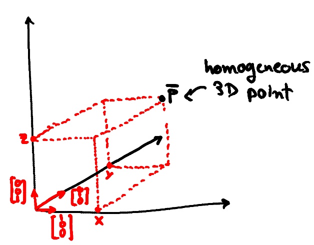
vector holding homogeneous coordinates of a point

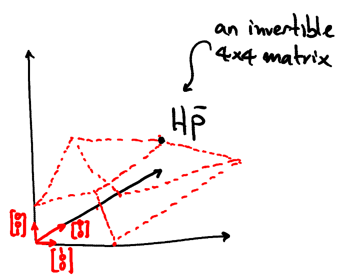
Topic 6:

3D Transformations

- Homogeneous coordinates in 3D
- Homogeneous 3D transformations
- Affine transformations & rotations in 3D

General Linear 3D Transformations



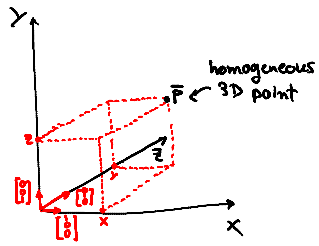


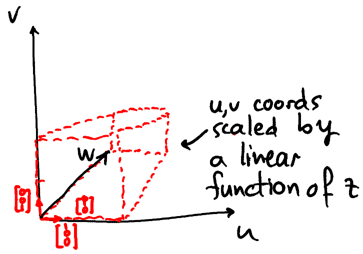
- The matrix H represents a very general set of transformations

General linear (preserve planes)

- Affine (preserve parallelism)
  - Arbitrary shearing
  - General scaling
  - Conformal (preserve angles)
    - Uniform scaling
    - Rigid (preserve lengths)
      - Translation
      - Rotation

General Linear 3D Transformations





- The matrix H represents a very general set of transformations
- Example: z-dependent tapering

non-linear in Euclidean coords

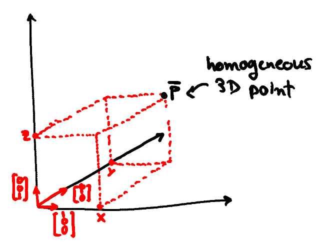
$$(x,y,z) \rightarrow \left(\alpha(z)x,\ \alpha(z)y,\ \alpha(z)z\right)$$

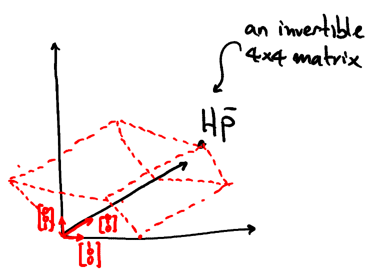
where $\alpha(z) = (\alpha_0 + \alpha_1 z)^{-1}$

$$\begin{bmatrix} u \\ v \\ w \\ q \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & \alpha_1 & \alpha_0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

⇒ linear in homogeneous coords!

Affine Transformations in 3D





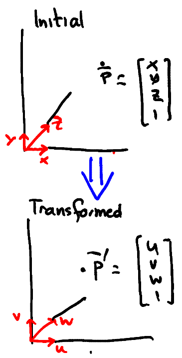
- The matrix H represents a very general set of transformations

General linear (preserve planes)

Affine (preserve parallelism)

The matrix H now takes a more restricted form!

Affine Transformations: Basic Properties



General form of matrix H

arbitrary 3x3 matrix

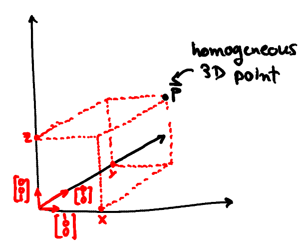
3x1 vector

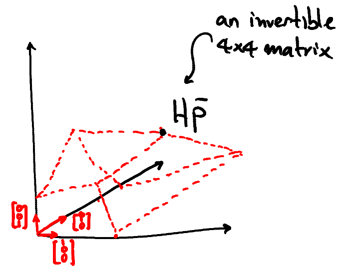
$$H = \left[\begin{array}{c|c} A & \bar{t} \\ \hline 000 & 1 \end{array}\right]$$

3. Affine transforms preserve the value of the last homogeneous coordinate $\Longleftrightarrow$ points at ∞ before the transform remain at ∞ afterwards

$$\left[\begin{array}{c|c} A & \bar{t} \\ \hline 000 & 1 \end{array}\right] \begin{bmatrix} x \\ y \\ z \\ \hline 1 \end{bmatrix} = \begin{bmatrix} A\begin{bmatrix} x \\ y \\ z \end{bmatrix} + \bar{t} \\ [0\ 0\ 0]\begin{bmatrix} x \\ y \\ z \end{bmatrix} + 1 \end{bmatrix} = \begin{bmatrix} A\begin{bmatrix} x \\ y \\ z \end{bmatrix} + \bar{t} \\ 1 \end{bmatrix}$$

From Affine to Rigid Transformations





Affine: $\left[\begin{array}{c|c} A & \bar{t} \\ \hline 000 & 1 \end{array}\right]$

General linear (preserve lines)

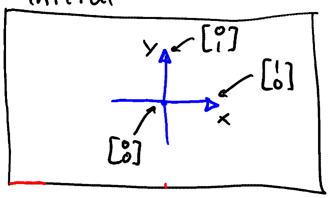
Affine (preserve parallelism)
- Arbitrary shearing
- General scaling

Conformal (preserve angles)
- Uniform scaling
- Reflection

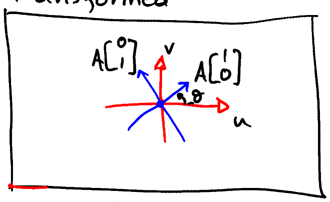
- Translation
- Rotation

Rigid Transformations: Rotations in 3D

Initial

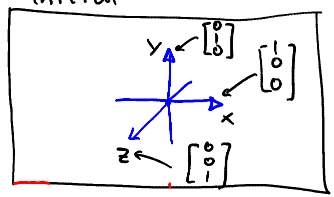


Transformed

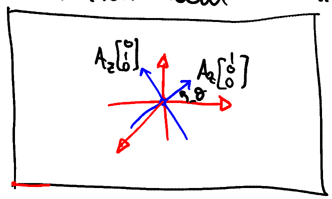


$$\left[\begin{array}{c:c} A_{2\times2} & \begin{matrix}0\\0\end{matrix} \\ \hdashline 0\ \ 0 & 1\end{array}\right] \qquad A = \begin{bmatrix}\cos\theta & -\sin\theta\\ \sin\theta & \cos\theta\end{bmatrix}$$

Initial



Rotation about z-axis

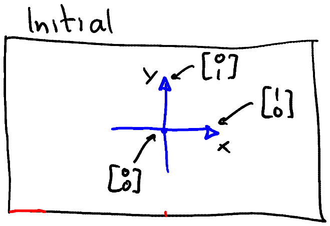


does not affect z-words.

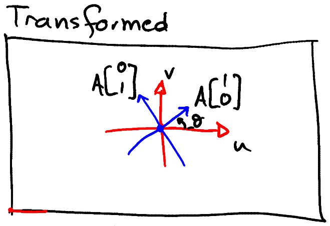
$$\left[\begin{array}{c:c} A_z & \begin{matrix}0\\0\\0\end{matrix} \\ \hdashline 0\,0\,0 & 1\end{array}\right] \qquad A_z = \begin{bmatrix}\cos\theta & -\sin\theta & 0\\ \sin\theta & \cos\theta & 0\\ 0 & 0 & 1\end{bmatrix}$$

Elementary Rotations in 3D

Initial



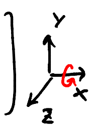
Transformed



$$\left[\begin{array}{c:c} A_{2\times2} & \begin{matrix}0\\0\end{matrix} \\ \hdashline 0\ \ 0 & 1\end{array}\right] \qquad A = \begin{bmatrix}\cos\theta & -\sin\theta\\ \sin\theta & \cos\theta\end{bmatrix}$$

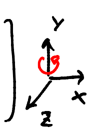
Rotation by θ about x axis

$$H_x^\theta = \left[\begin{array}{c:c} A_x & \begin{matrix}0\\0\\0\end{matrix} \\ \hdashline 0\,0\,0 & 1\end{array}\right] \qquad A_x = \begin{bmatrix}1 & 0 & 0\\ 0 & \cos\theta & -\sin\theta\\ 0 & \sin\theta & \cos\theta\end{bmatrix}$$



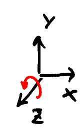
Rotation by θ about y axis:

$$H_y^\theta = \left[\begin{array}{c:c} A_y & \begin{matrix}0\\0\\0\end{matrix} \\ \hdashline 0\,0\,0 & 1\end{array}\right] \qquad A_y = \begin{bmatrix}\cos\theta & 0 & \sin\theta\\ 0 & 1 & 0\\ -\sin\theta & 0 & \cos\theta\end{bmatrix}$$



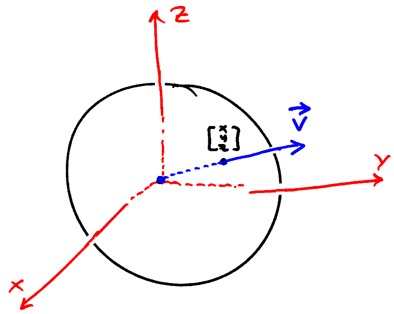
Rotation by θ about z axis:

$$H_z^\theta = \left[\begin{array}{c:c} A_z & \begin{matrix}0\\0\\0\end{matrix} \\ \hdashline 0\,0\,0 & 1\end{array}\right] \qquad A_z = \begin{bmatrix}\cos\theta & -\sin\theta & 0\\ \sin\theta & \cos\theta & 0\\ 0 & 0 & 1\end{bmatrix}$$



Rotation About Arbitrary Vector?

Question: How do we define A when it is a rotation about an arbitrary vector $\vec{v}$?



Ans: Express it as a composition of the three elementary matrices A_x, A_y, A_z

Rotation by ϑ about x axis

$$H_x^{\vartheta} = \left[\begin{array}{c:c} A_x & \begin{matrix}0\\0\\0\end{matrix} \\ \hdashline 0\,0\,0 & 1 \end{array}\right] \quad A_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\vartheta & -\sin\vartheta \\ 0 & \sin\vartheta & \cos\vartheta \end{bmatrix}$$

Rotation by ϑ about y axis:

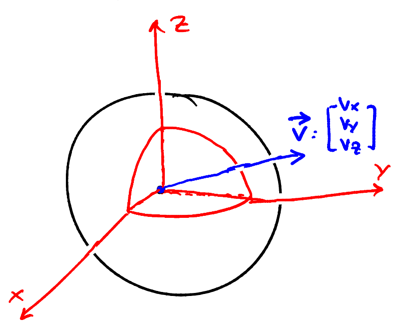
$$H_y^{\vartheta} = \left[\begin{array}{c:c} A_y & \begin{matrix}0\\0\\0\end{matrix} \\ \hdashline 0\,0\,0 & 1 \end{array}\right] \quad A_y = \begin{bmatrix} \cos\vartheta & 0 & \sin\vartheta \\ 0 & 1 & 0 \\ -\sin\vartheta & 0 & \cos\vartheta \end{bmatrix}$$

Rotation by ϑ about z axis:

$$H_z^{\vartheta} = \left[\begin{array}{c:c} A_z & \begin{matrix}0\\0\\0\end{matrix} \\ \hdashline 0\,0\,0 & 1 \end{array}\right] \quad A_z = \begin{bmatrix} \cos\vartheta & -\sin\vartheta & 0 \\ \sin\vartheta & \cos\vartheta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Rotation About Arbitrary Vector: Construction

Question: How do we define A when it is a rotation of ϕ about an arbitrary vector $\vec{v}$?



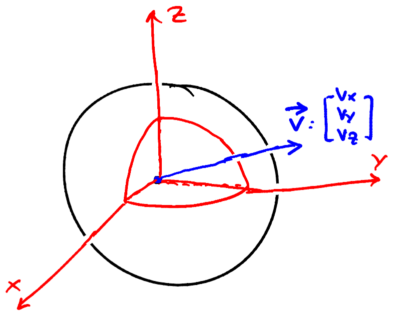
Ans: Express it as a composition of the three elementary matrices A_x, A_y, A_z

Basic idea: Since we know how to do rotations about z, we will do the following:

* Align $\vec{v}$ with the $\vec{z}$ axis "temporarily" (using axis-aligned rotations)
* Rotate about $\vec{v}$ using A_z
* Undo the temporary alignment

Rotation About Arbitrary Vector: Construction

Question: How do we define A when it is a rotation of ϕ about an arbitrary vector $\vec{v}$?



Ans: Express it as a composition of the three elementary matrices A_x, A_y, A_z

Step 1: Convert $\vec{v}$ to unit length

$$\vec{u} = \frac{\vec{v}}{\sqrt{v_x^2 + v_y^2 + v_z^2}}$$

Step 2: Compute the spherical coordinates of $\vec{u}$

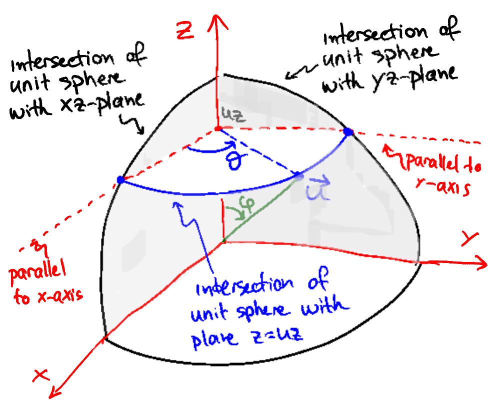
i.e. coordinates of $\vec{u}$ in the parameterization of a sphere

Aside: Spherical Coordinates of a Unit Vector

The cross-section of plane $z = u_z$

$$(\sin\varphi\cos\vartheta,\ \sin\varphi\sin\vartheta,\ u_z)$$

radius of circular cross-section



Step 1: Convert $\vec{v}$ to unit length

$$\vec{u} = \frac{\vec{v}}{\sqrt{v_x^2 + v_y^2 + v_z^2}}$$

Step 2: Compute the spherical coordinates of $\vec{u}$

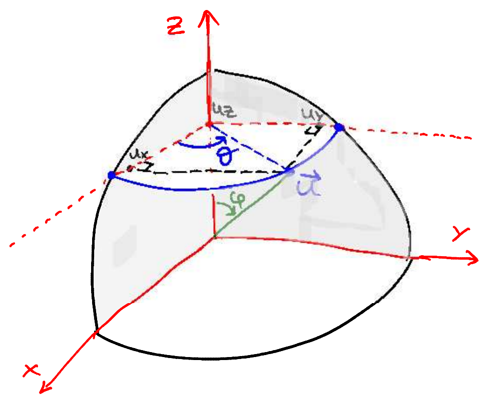
i.e. coordinates of $\vec{u}$ in the parameterization of a sphere

Aside: Spherical Coordinates of a Unit Vector

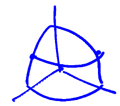
The cross-section of plane $z = u_z$

$(\sin\varphi \cos\vartheta, \sin\varphi \sin\vartheta, u_z)$

$\Rightarrow \frac{u_y}{u_x} = \tan\vartheta \Rightarrow \boxed{\vartheta = \arctan\left(\frac{u_y}{u_x}\right)}$



Step 1: Convert $\vec{v}$ to unit length



$$\vec{u} = \frac{\vec{v}}{\sqrt{v_x^2 + v_y^2 + v_z^2}}$$

Step 2: Compute the spherical coordinates of $\vec{u}$

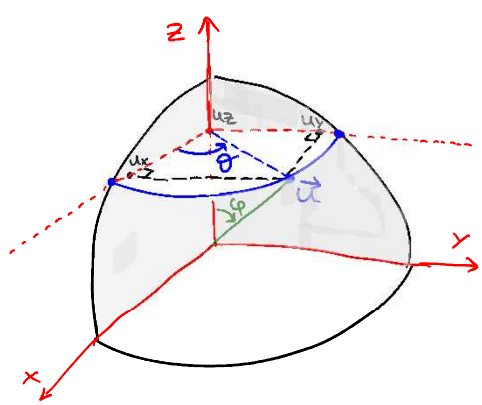
i.e. coordinates of $\vec{u}$ in the parameterization of a sphere

Aside: Spherical Coordinates of a Unit Vector

The cross-section of plane $z = u_z$

$(\sin\varphi \cos\vartheta, \sin\varphi \sin\vartheta, u_z)$

$u_z = \cos\varphi \Rightarrow \boxed{\varphi = \arccos(u_z)}$



Step 1: Convert $\vec{v}$ to unit length

$$\vec{u} = \frac{\vec{v}}{\sqrt{v_x^2 + v_y^2 + v_z^2}}$$

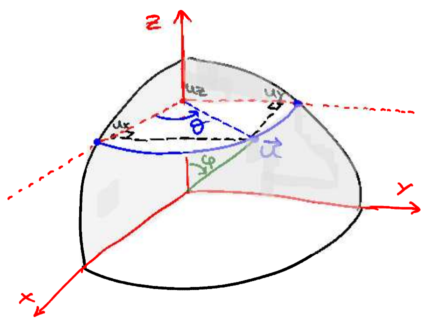
Step 2: Compute the spherical coordinates of $\vec{u}$

i.e. coordinates of $\vec{u}$ in the parameterization of a sphere

Rotation About Arbitrary Vector: Construction

Step 3: Align z axis with vector $\vec{u}$:

- Rotate by $-\vartheta$ about z
- Rotate by $-\varphi$ about y



Step 4: Rotate by desired angle ψ about z

Step 1: Convert $\vec{v}$ to unit length

$$\vec{u} = \frac{\vec{v}}{\sqrt{v_x^2 + v_y^2 + v_z^2}}$$

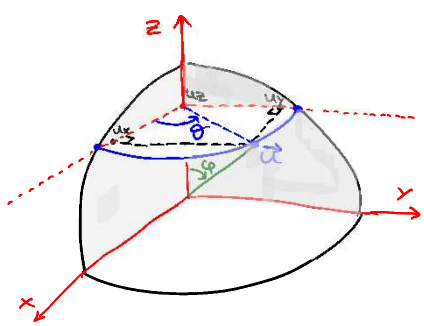
Step 2: Compute the spherical coordinates of $\vec{u}$

i.e. coordinates of $\vec{u}$ in the parameterization of a sphere

Rotation About Arbitrary Vector: Construction

Step 3: Align z axis with vector $\vec{u}$:

- Rotate by $-\vartheta$ about z
- Rotate by $-\varphi$ about y



Step 4: Rotate by desired angle ψ about z

Step 5: Move z-axis back to its original position

- Rotate by φ about y
- Rotate by ϑ about z

Final transform:

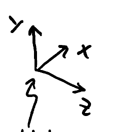
$$\bar{p}' = H_z^{\vartheta} H_y^{\varphi} H_z^{\psi} H_y^{-\varphi} H_z^{-\vartheta} \bar{p}$$

Topic 7:

3D Viewing

- Orthographic projection
- The world-to-camera transformation
- Perspective projection
- The transformation chain for 3D viewing

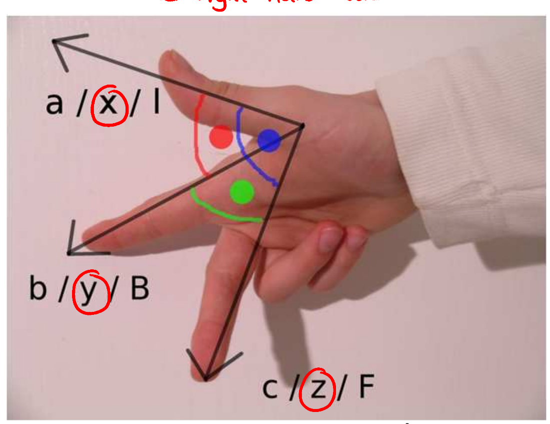
The Camera-Centered Coordinate System



right-handed coordinate system

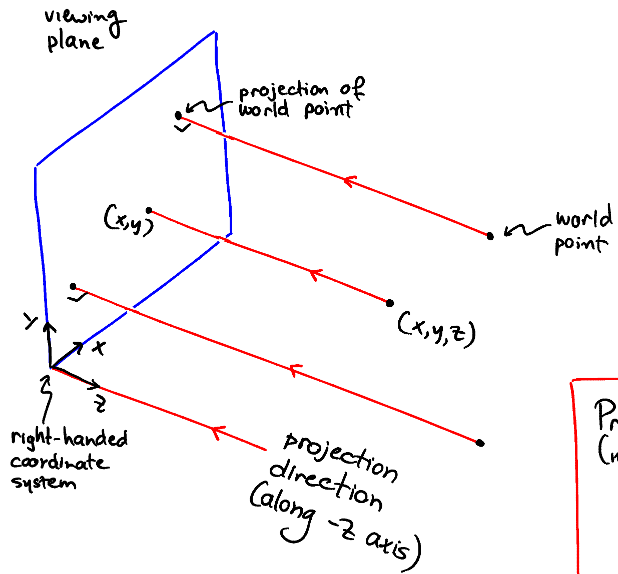
world point (x,y,z)

The right-hand rule



rechte-hand-regel.jpg (wikipedia.com)

Orthographic Projection



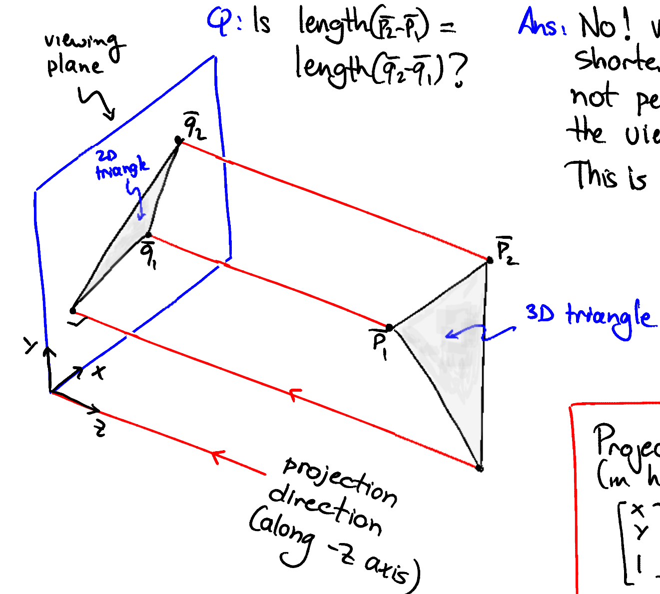
Projection equation (in homogeneous coords)

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

Orthographic Projection (cont.)

Q: Is $\text{length}(\bar{P}_2-\bar{P}_1) = \text{length}(\bar{q}_2-\bar{q}_1)$?

Ans: No! vector $P_2'-P_1'$ is shorter when P_2-P_1 is not perpendicular to the viewing direction. This is called foreshortening

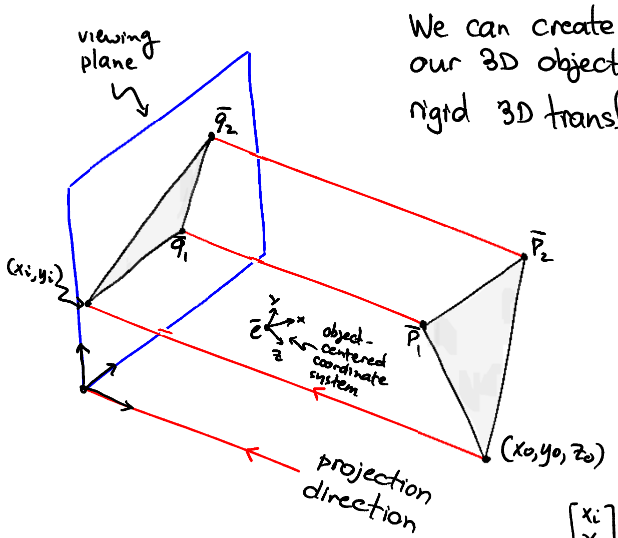


Projection equation (in homogeneous coords)

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

The Object-to-Camera Transformation

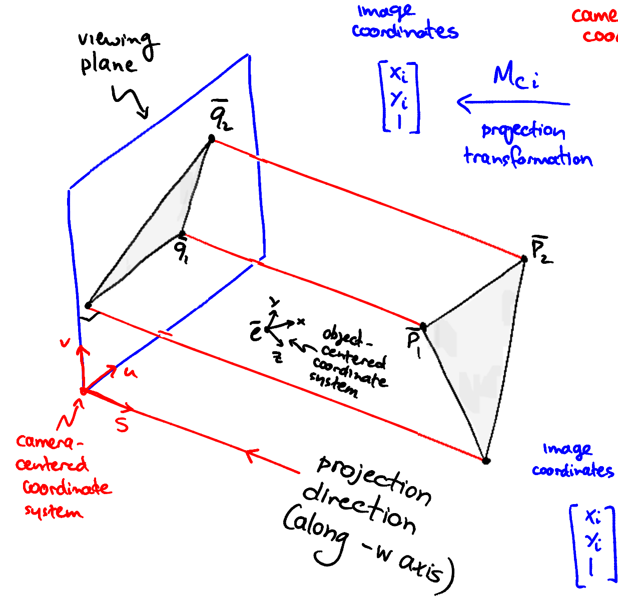
We can create different views of our 3D object by applying a rigid 3D transformation M_{oc} to it



$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \underbrace{\begin{bmatrix} R_{oc} & \bar{e} \\ 000 & 1 \end{bmatrix}}_{M_{oc}} \begin{bmatrix} x_o \\ y_o \\ z_o \\ 1 \end{bmatrix}$$

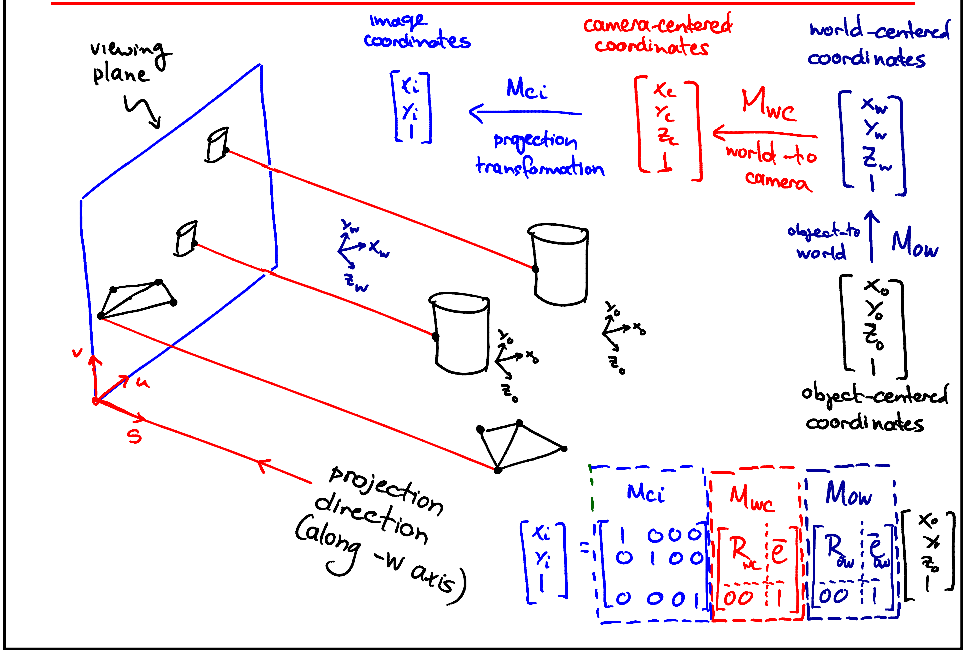
Main Transformations Used in 3D Viewing

$$\underset{\text{image coordinates}}{\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}} \xleftarrow[\text{projection transformation}]{M_{ci}} \underset{\text{camera-centered coordinates}}{\begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix}} \xleftarrow[\text{object-to camera transformation}]{M_{oc}} \underset{\text{object-centered coordinates}}{\begin{bmatrix} X_o \\ Y_o \\ Z_o \\ 1 \end{bmatrix}}$$

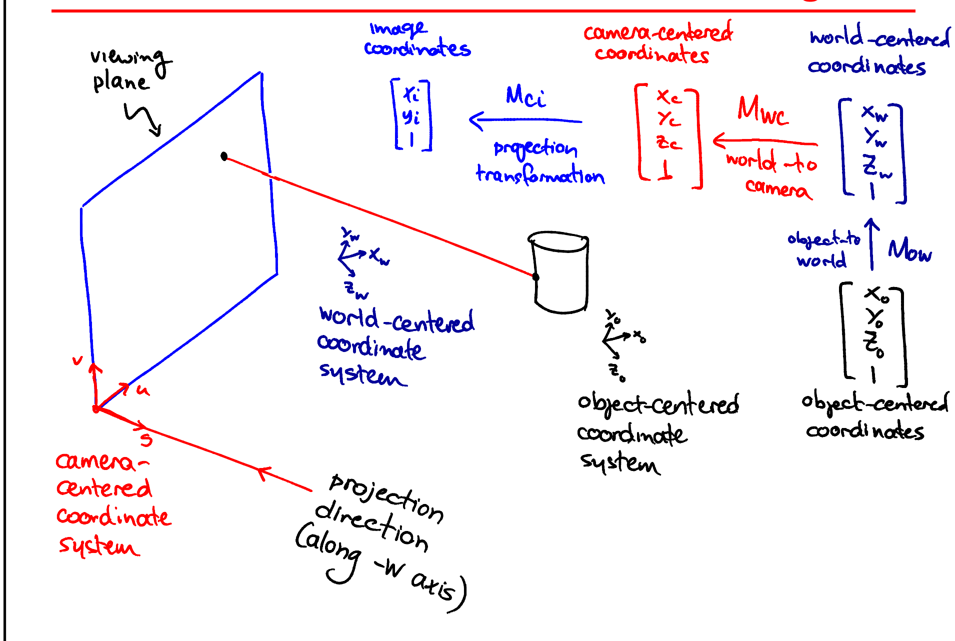


$$\underset{\text{image coordinates}}{\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}} = \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{M_{ci}} \underbrace{\begin{bmatrix} R_{oc} & \bar{e}_{oc} \\ 00 & 1 \end{bmatrix}}_{M_{oc}} \underset{\text{object coordinates}}{\begin{bmatrix} x_o \\ y_o \\ z_o \\ 1 \end{bmatrix}}$$

Main Transformations Used in 3D Viewing



The Transformation Chain for 3D Viewing



Transformation Chain for 3D Viewing (partial)

Object-to-world transformation (M_{ow})

$$\begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} = \begin{bmatrix} R_{ow} & e_{ow} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_o \\ y_o \\ z_o \\ 1 \end{bmatrix}$$

a 4x4 matrix that maps object-centered coords to world-centered coords

World-to-camera transformation (M_{wc})

$$\begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} = \begin{bmatrix} R_{wc} & e_{wc} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}$$

a 4x4 matrix that maps world-centered coords to camera-centered coords

Camera-to-image transformation (M_{ci}) (a.k.a. projection)

$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix}$$

a 4x3 matrix that maps camera-centered 3D coords to 2D image coords

Transformation Chain for 3D Viewing (partial)

Object-to-world transformation (M_{ow})

$$\begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} = \begin{bmatrix} R_{ow} & e_{ow} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_o \\ y_o \\ z_o \\ 1 \end{bmatrix}$$

a 4x4 matrix that maps object-centered coords to world-centered coords

World-to-camera transformation (M_{wc})

$$\begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} = \begin{bmatrix} R_{wc} & e_{wc} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}$$

a 4x4 matrix that maps world-centered coords to camera-centered coords

Camera-to-image transformation (M_{ci}) (a.k.a. projection)

$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix}$$

a 4x3 matrix that maps camera-centered 3D coords to 2D image coords

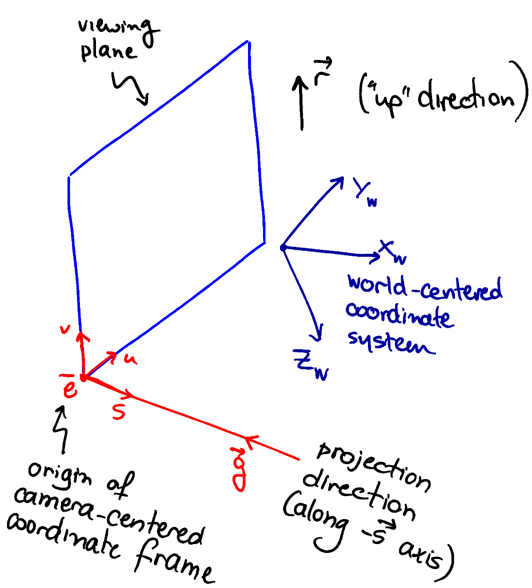
Q: How do we define M_{wc}?

Topic 7:

3D Viewing

- Orthographic projection
- The world-to-camera transformation
- Perspective projection
- The transformation chain for 3D viewing

Computing the World-to-Camera Transform



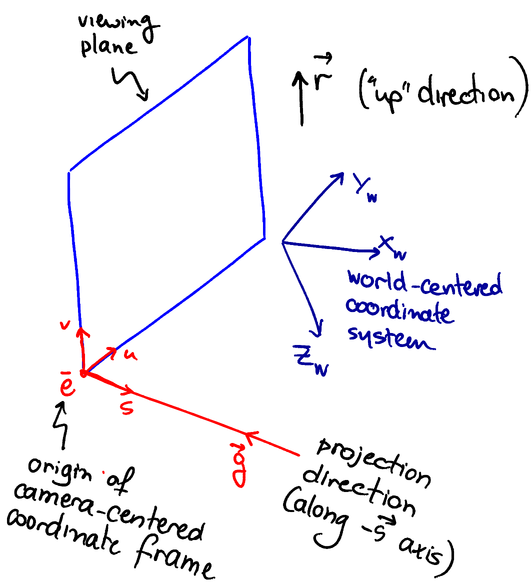
$$\begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} \xleftarrow[\text{world-to camera}]{M_{wc}} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}$$

Goal: given ① camera origin $\bar{e}$ and ② a projection direction unit vector $\vec{g}$, compute M_{wc}

1. Let $\vec{r}$ be the "up" vector in world coords
2. Set $\vec{s} = -\vec{g}$
3. Define 2 perpendicular unit vectors on view plane

$$\boxed{\begin{aligned} \vec{u} &= (\vec{r} \times \vec{s}) / \|\vec{r} \times \vec{s}\| \\ \vec{v} &= (\vec{s} \times \vec{u}) / \|\vec{s} \times \vec{u}\| \end{aligned}}$$

Computing the World-to-Camera Transform



$$\begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} \xleftarrow[\text{world-to camera}]{M_{wc}} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}$$

1. Let $\vec{r}$ be the "up" vector in world coords
2. Set $\vec{s} = -\vec{g}$
3. Define 2 perpendicular unit vectors on view plane

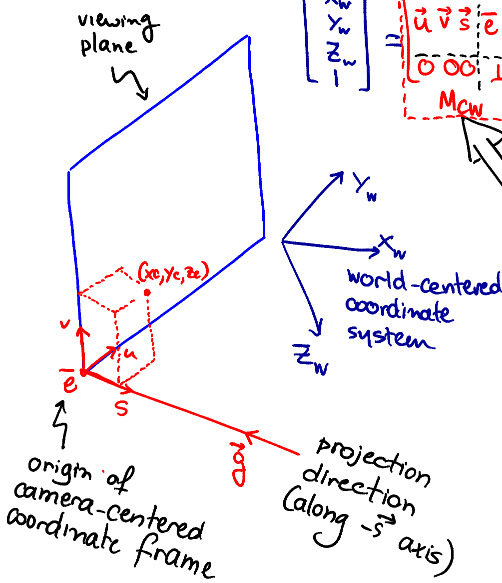
$$\boxed{\begin{aligned} \vec{u} &= \vec{r} \times \vec{s} \,/\, \|\vec{r} \times \vec{s}\| \\ \vec{v} &= \vec{s} \times \vec{u} \,/\, \|\vec{s} \times \vec{u}\| \end{aligned}}$$

4. Use vectors $\vec{u}, \vec{v}, \vec{s}$ as the right-handed camera coord system

First Compute Camera-to-World Transform...

$$\begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} \vec{u} & \vec{v} & \vec{s} & \bar{e} \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{M_{cw}} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix}$$

camera-to-world transformation



Camera coords $\xrightarrow{\text{camera-to world}}$ world coords

$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$ becomes $\begin{bmatrix} \bar{e} \\ 1 \end{bmatrix}$

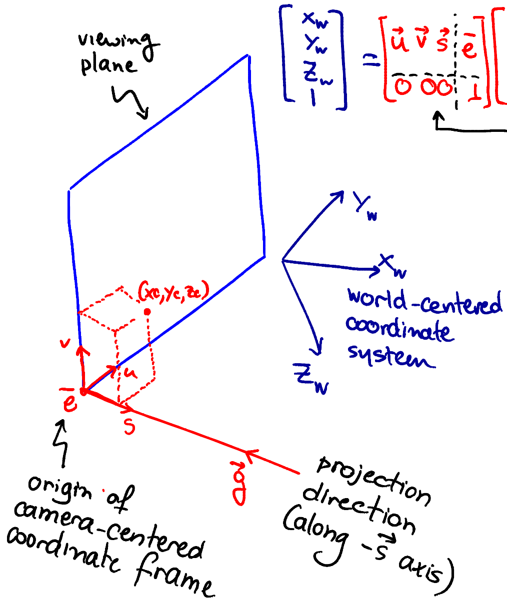
$\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$ becomes $\begin{bmatrix} \vec{u} \\ 0 \end{bmatrix}$

$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$ becomes $\begin{bmatrix} \vec{v} \\ 0 \end{bmatrix}$

$\begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$ becomes $\begin{bmatrix} \vec{s} \\ 0 \end{bmatrix}$

$\begin{bmatrix} x_c \\ y_c \\ z_c \\ 0 \end{bmatrix}$ becomes $\begin{bmatrix} x_c\vec{u} + y_c\vec{v} + z_c\vec{v} \\ 0 \end{bmatrix}$

... then Compute its Inverse



$$\begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} = \left[\begin{array}{ccc|c} \vec{u} & \vec{v} & \vec{s} & \bar{e} \\ \hline 0 & 0 & 0 & 1 \end{array}\right] \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix}$$

- World-to-camera trans. = inverse of this matrix
- Since $[\vec{u}\ \vec{v}\ \vec{w}]$ orthonormal, (i.e. $\vec{u}, \vec{v}, \vec{g}$ unit-length & perpendicular)

$$[\vec{u}\ \vec{v}\ \vec{s}]^{-1} = [\vec{u}\ \vec{v}\ \vec{s}]^T$$

- So M_{wc} is given by

$$M_{wc} = \left[\begin{array}{c|c} [\vec{u}\ \vec{v}\ \vec{s}]^T & -[\vec{u}\ \vec{v}\ \vec{s}]^T \bar{e} \\ \hline 0\ 0\ 0 & 1 \end{array}\right]$$

Transformation Chain for 3D Viewing (partial)

Object-to-world transformation (M_{ow})

$$\begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} = \begin{bmatrix} R_{ow} & e_{ow} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_o \\ y_o \\ z_o \\ 1 \end{bmatrix}$$

a 4x4 matrix that maps object-centered coords to world-centered coords

World-to-camera transformation (M_{wc})

$$\begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} = \begin{bmatrix} R_{wc} & e_{wc} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}$$

a 4x4 matrix that maps world-centered coords to camera-centered coords

Camera-to-image transformation (M_{ci}) (a.k.a. projection)

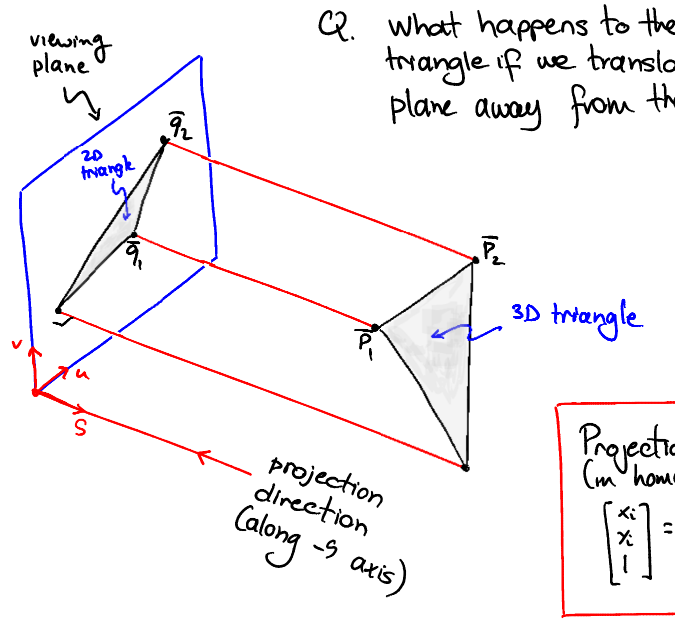
$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix}$$

a 4x3 matrix that maps camera-centered 3D coords to 2D image coords

Q: Does this matrix accurately represent how objects are projected in real photos?

How Accurate is Orthographic Projection?

Q. What happens to the projected triangle if we translate the viewing plane away from the object?

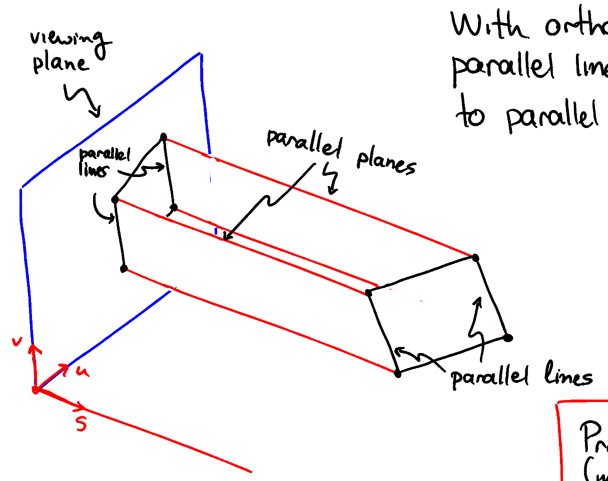


Projection equation (in homogeneous coords)

$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix}$$

How Accurate is Orthographic Projection?

With orthographic projection, parallel lines in 3D project to parallel lines in the image



Projection equation (in homogeneous coords)

$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix}$$

Transformation Chain for 3D Viewing (partial)

Object-to-world transformation (M_{ow})

$$\begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} = \begin{bmatrix} R_{ow} & e_{ow} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_o \\ y_o \\ z_o \\ 1 \end{bmatrix}$$

a 4x4 matrix that maps object-centered coords to world-centered coords

World-to-camera transformation (M_{wc})

$$\begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} = \begin{bmatrix} R_{wc} & e_{wc} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}$$

a 4x4 matrix that maps world-centered coords to camera-centered coords

Camera-to-image transformation (M_{ci}) (a.k.a. projection)

$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = \begin{bmatrix} ? \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix}$$

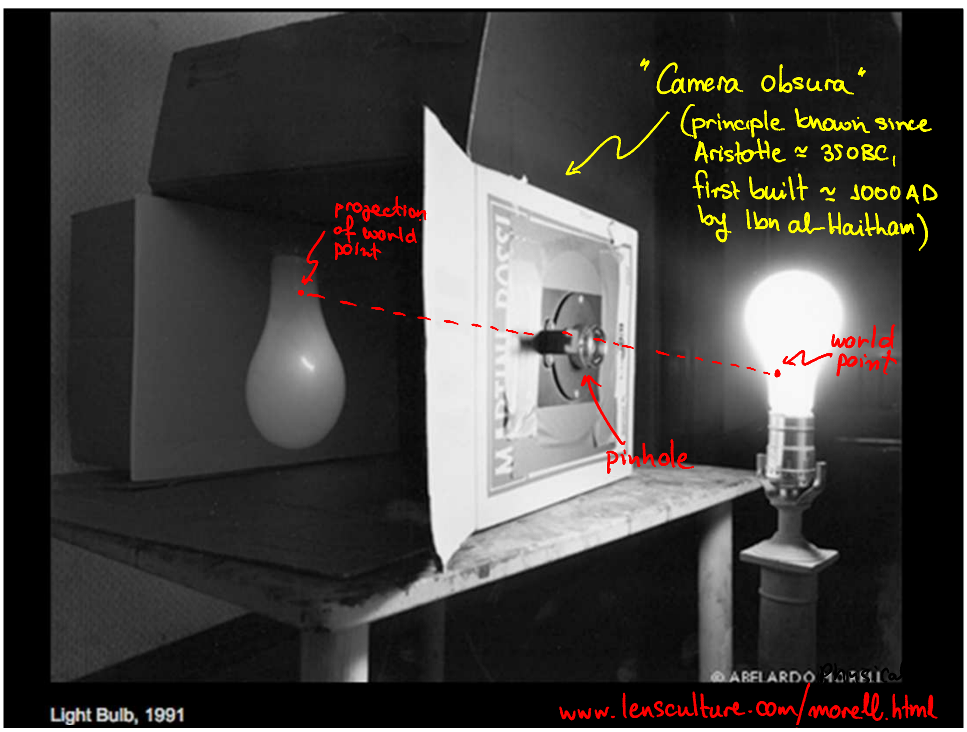
a 4x3 matrix that maps camera-centered 3D coords to 2D image coords

We need to define a more accurate 4x3 matrix M_{ci}

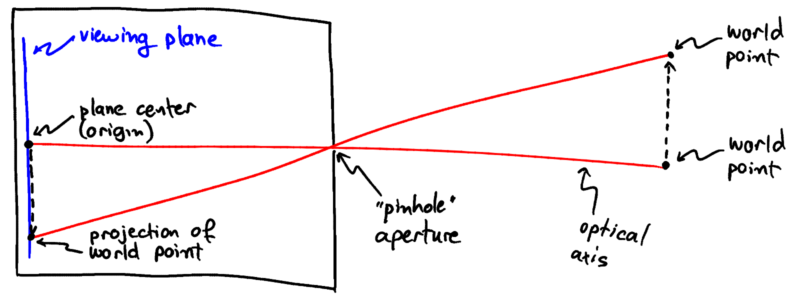
Topic 7:

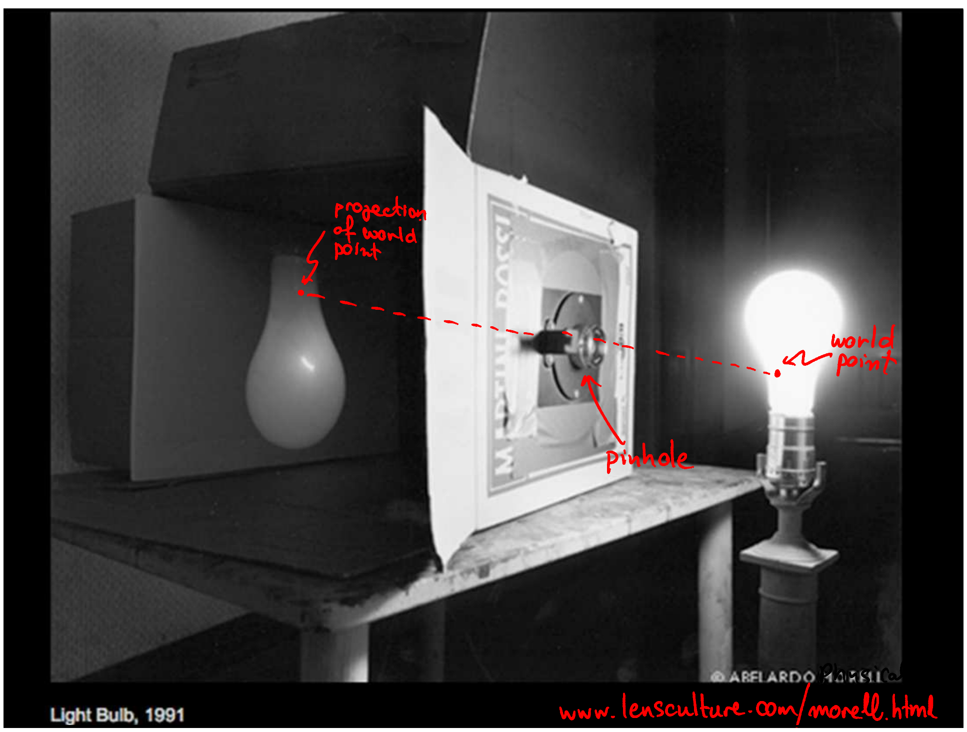
3D Viewing

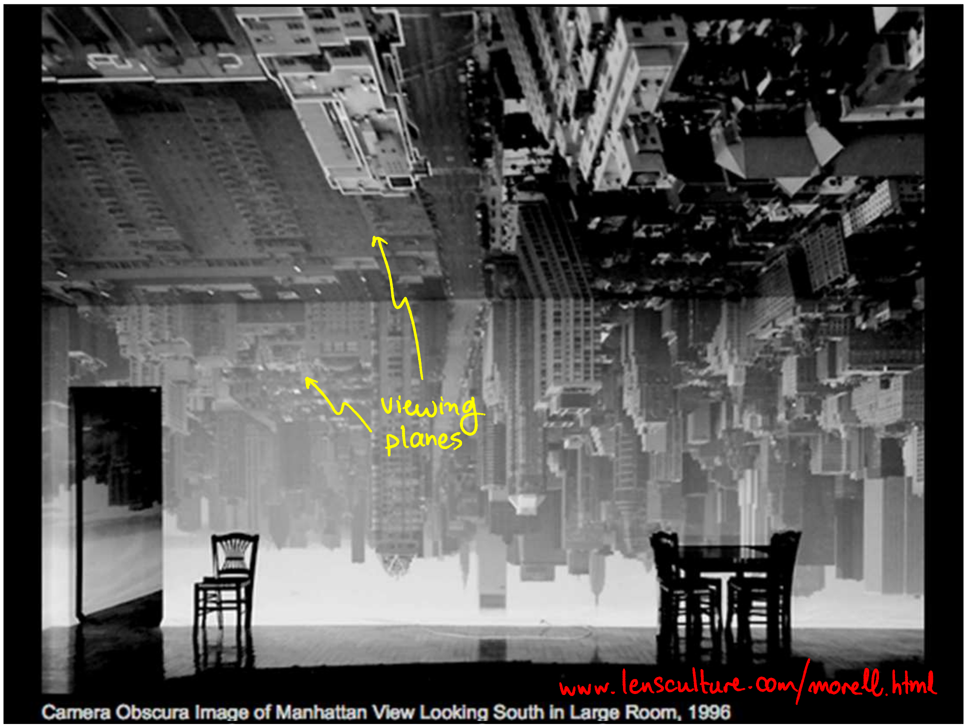
- Orthographic projection
- The world-to-camera transformation
- **Perspective projection**
- The transformation chain for 3D viewing

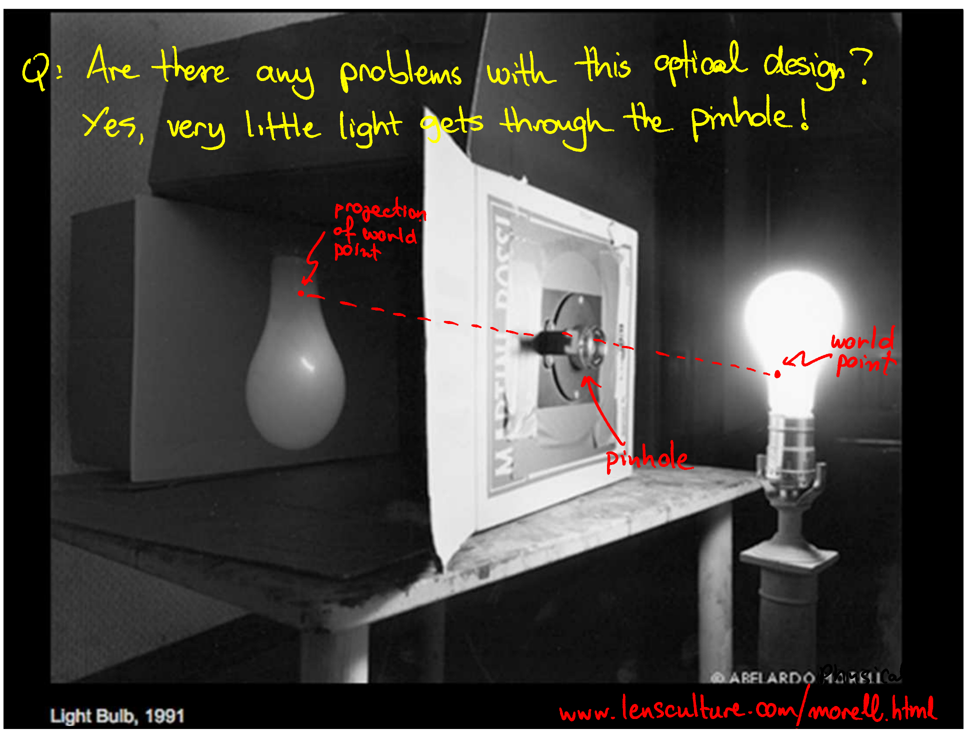


The Pinhole Camera

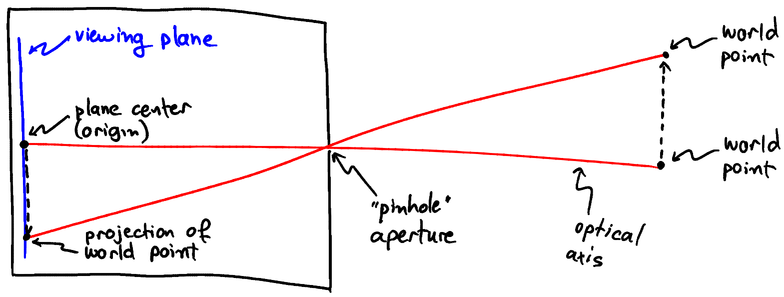




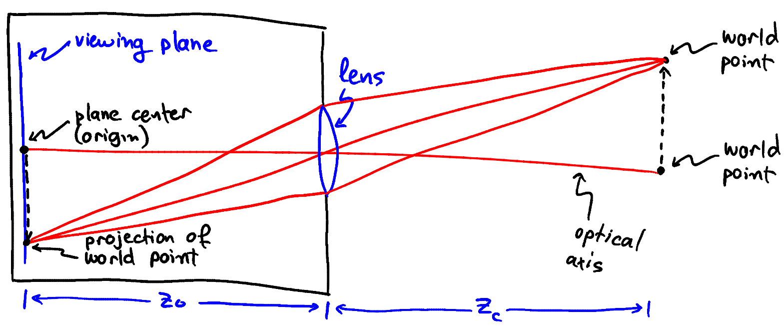

viewing planes
www.lensculture.com/morell.html
Camera Obscura Image of Manhattan View Looking South in Large Room, 1996


Q: Are there any problems with this optical design?
Yes, very little light gets through the pinhole!
projection of world point
world point
pinhole
© ABELARDO MORELL
Light Bulb, 1991
www.lensculture.com/morell.html

The Pinhole Camera



Simple Lens-Based Camera & Thin-Lens Law

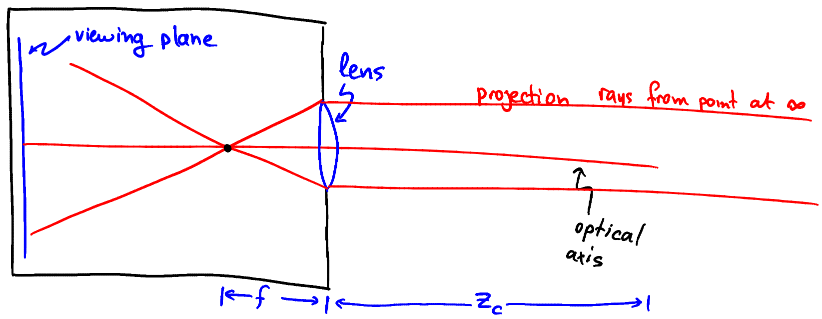


When $z_c \to \infty$, $z_0 \to f$
So f is the distance at which rays converge for a world point at ∞

Thin-lens law:

$$\boxed{\frac{1}{z_0} + \frac{1}{z_c} = \frac{1}{f}}$$

Simple Lens-Based Camera & Thin-Lens Law



When $z_c \to \infty$, $z_0 \to f$
So f is the distance at which rays converge for a world point at ∞

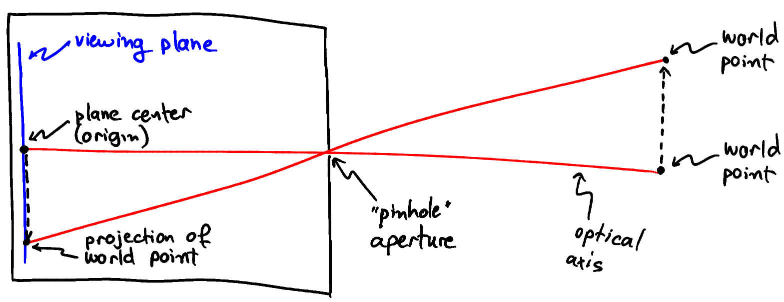
Thin-lens law:

$$\boxed{\frac{1}{z_0} + \frac{1}{z_c} = \frac{1}{f}}$$

When objects are very far, projection is well-approximated by orthographic projection

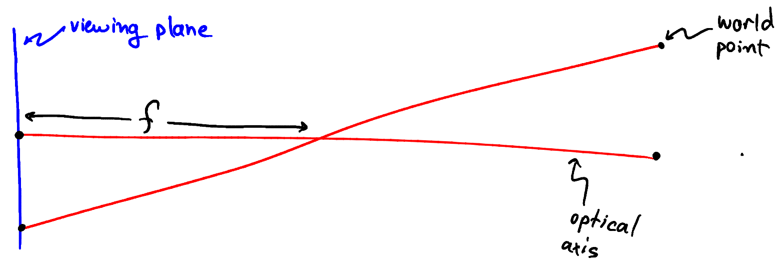
The Pinhole Camera

We will only consider the idealized pinhole model here (a.k.a. perspective projection)



The Pinhole Camera: Basic Geometry

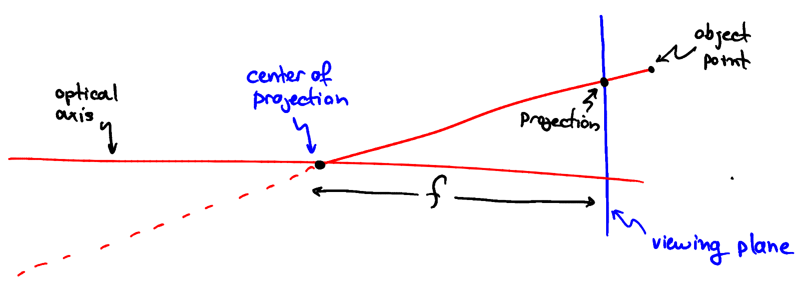
We will only consider the idealized pinhole model here
(a.k.a. perspective projection)



Simplification #1: Take plane-to-pinhole distance = f

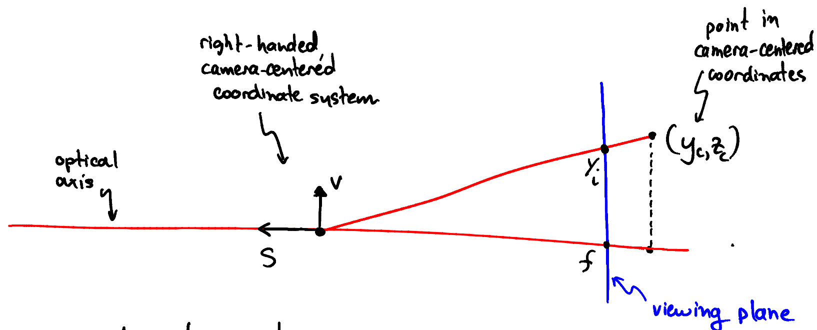
The Pinhole Camera: Basic Geometry in 2D

We will only consider the idealized pinhole model here
(a.k.a. perspective projection)



Simplification #1: Take plane-to-pinhole distance = f
Simplification #2: "Undo" image reversal by placing viewing plane in front of pinhole

The Perspective Projection Equation in 2D

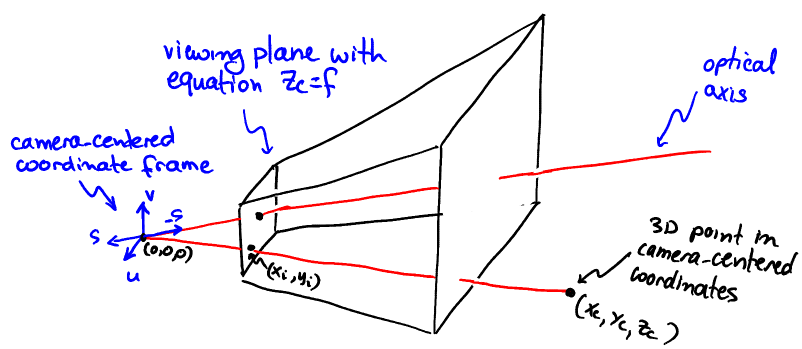


From similar triangles,

$$\frac{y_c}{z_c} = \frac{Y_i}{f} \Rightarrow \boxed{Y_i = \frac{f}{z_c} Y_c}$$

The perspective projection equation

The Perspective Projection Equations in 3D



From similar triangles,

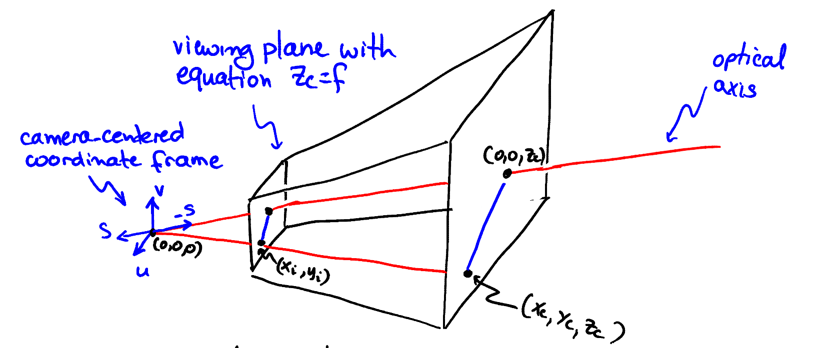
$$\frac{Y_c}{Z_c} = \frac{Y_i}{f}$$

analogously, for x_i

$$\frac{X_c}{Z_c} = \frac{X_i}{f} \Rightarrow \boxed{\begin{aligned} X_i &= \frac{f}{Z_c} X_c \\ Y_i &= \frac{f}{Z_c} Y_c \end{aligned}}$$

The perspective projection equations

The Perspective Projection Equations in 3D



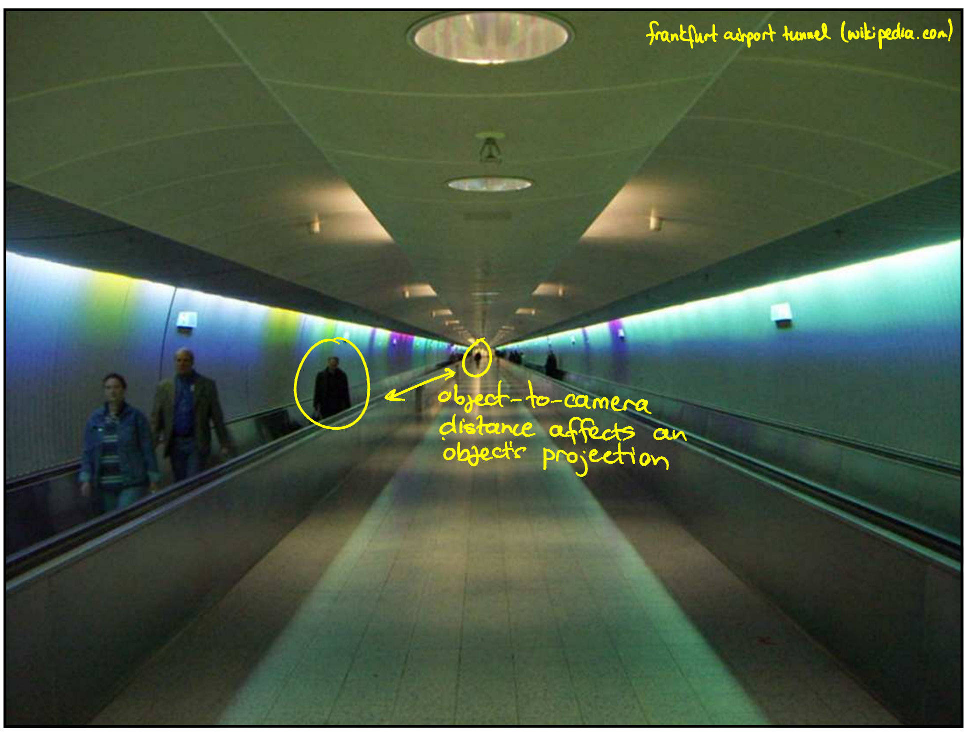
From similar triangles,

$$\frac{Y_c}{Z_c} = \frac{Y_i}{f}$$

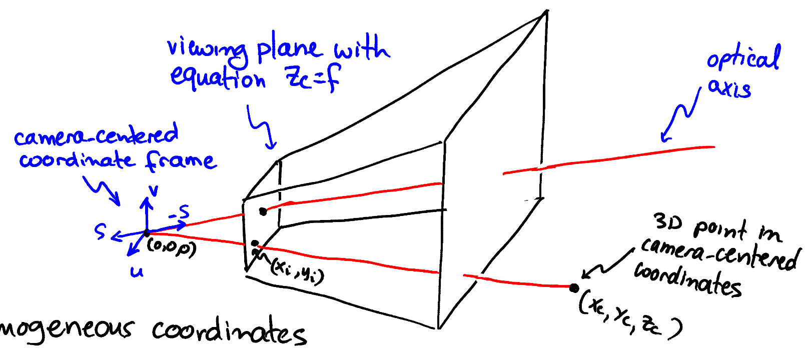
analogously, for x_i

$$\frac{X_c}{Z_c} = \frac{X_i}{f} \quad \Rightarrow \quad \boxed{X_i = \frac{f}{Z_c} X_c \qquad Y_i = \frac{f}{Z_c} Y_c}$$

As objects move farther away (i.e. $|Z_c|$ increases) $\Rightarrow$ their projection gets smaller and smaller



Perspective Projection & Homogeneous Coords



In homogeneous coordinates

$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{f x_c}{z_c} \\ \frac{f y_c}{z_c} \\ 1 \end{bmatrix} \cong \begin{bmatrix} x_c \\ y_c \\ \frac{z_c}{f} \end{bmatrix} \cong \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{1}{f} & 0 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix}$$

$$\boxed{x_i = \frac{f}{z_c} x_c \qquad y_i = \frac{f}{z_c} y_c}$$

Transformation Chain for 3D Viewing (partial)

Object-to-world transformation (M_{ow})

$$\begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} = \begin{bmatrix} R_{ow} & e_{ow} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_o \\ y_o \\ z_o \\ 1 \end{bmatrix}$$

a 4x4 matrix that maps object-centered coords to world-centered coords

World-to-camera transformation (M_{wc})

$$\begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} = \begin{bmatrix} R_{wc} & e_{wc} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}$$

a 4x4 matrix that maps world-centered coords to camera-centered coords

Camera-to-image transformation (M_{ci}) (a.k.a. projection)

$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{1}{f} & 0 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix}$$

a 4x3 matrix that maps camera-centered 3D coords to 2D image coords

models perspective projection

The Canonical View Volume Transform

We can re-write the projection equation as a 4D linear transformation followed by orthographic projection:

$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} \cong \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & \frac{1}{f} & 0 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix}$$

In homogeneous coordinates:

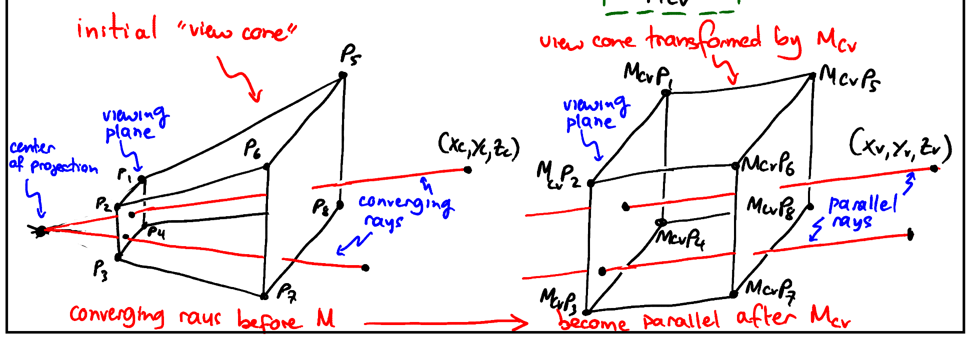
$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{f x_c}{z_c} \\ \frac{f y_c}{z_c} \\ 1 \end{bmatrix} \cong \begin{bmatrix} x_c \\ y_c \\ \frac{z_c}{f} \end{bmatrix} \cong \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{1}{f} & 0 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix}$$

$$\boxed{\begin{aligned} x_i &= \frac{f}{z_c} x_c \\ y_i &= \frac{f}{z_c} y_c \end{aligned}}$$

The Canonical View Volume Transform

This adds yet another 3D linear transformation M_{cv} in the "transformation chain" that allows the projection to always be modeled as orthographic

$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} \cong \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_v \\ y_v \\ z_v \\ 1 \end{bmatrix} \qquad \begin{bmatrix} x_v \\ y_v \\ z_v \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & \frac{1}{f} & 0 \end{bmatrix}}_{M_{cv}} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix}$$



Transformation Chain for 3D Viewing (complete)

Object-to-world transformation (M_{ow})

$$\begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} = \begin{bmatrix} R_{ow} & e_{ow} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_o \\ y_o \\ z_o \\ 1 \end{bmatrix}$$

a 4x4 matrix that maps object-centered coords to world-centered coords

World-to-camera transformation (M_{wc})

$$\begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} = \begin{bmatrix} R_{wc} & e_{wc} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}$$

a 4x4 matrix that maps world-centered coords to camera-centered coords

Camera-to-canonical view transformation (M_{cv})

$$\begin{bmatrix} x_v \\ y_v \\ z_v \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/f & 0 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix}$$

a 4x4 matrix that maps camera-centered coords to canonical view coords

Canonical view-to image transformation (M_{vi}) (a.k.a. projection)

$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} \cong \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_v \\ y_v \\ z_v \\ 1 \end{bmatrix}$$

a 4x3 matrix that maps canonical view 3D coords to 2D image coords

Orthographic projection matrix