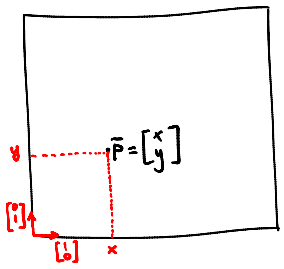
Topic 3:

2D Transformations

- Homogeneous coordinates
- Homogeneous 2D transformations
- Affine transformations & restrictions

Representing Points by Euclidean 2D Coords



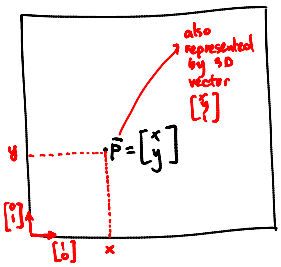
- "Standard" (Euclidean) representation of a point $\bar{p}$:

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

basis vectors

Euclidean coordinates

Euclidean Coords ⇒ Homogeneous Coords



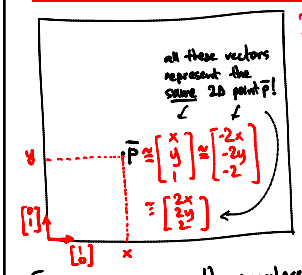
- "Standard" (Euclidean) representation of a point p:

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

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

pixel coordinates	homogeneous 2D coordinates
$\begin{bmatrix}x\\y\end{bmatrix}$	$\longrightarrow \begin{bmatrix}x\\y\\1\end{bmatrix}$

2D Homogeneous Coordinates: Definition



Definition:
Homogeneous representation of $\bar{p}$

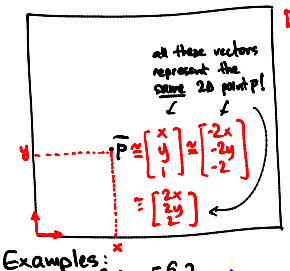
p represented by any 3D vector $\begin{bmatrix}\lambda x\\ \lambda y\\ \lambda\end{bmatrix}$ with $\lambda \neq 0$

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

- For any $\lambda \neq 0$, the numbers $\lambda x, \lambda y, \lambda$ are called the homogeneous coordinates of point $\bar{p}$

pixel coordinates	homogeneous 2D coordinates
$\begin{bmatrix}x\\y\end{bmatrix}$	$\longrightarrow \begin{bmatrix}\lambda x\\ \lambda y\\ \lambda\cdot 1\end{bmatrix}$ $\lambda\neq 0$

2D Homogeneous Coordinates: Equality



Definition (Homogeneous Equality)

Two vectors of homogeneous coords $\bar{v}_1=\begin{bmatrix}x\\y\\w\end{bmatrix}$ and $\bar{v}_2=\begin{bmatrix}x'\\y'\\w'\end{bmatrix}$ are called equal if they represent the same 2D point:

$\bar{v}_1 \cong \bar{v}_2$ ($\cong$ denotes homog. equality)

$\Leftrightarrow$

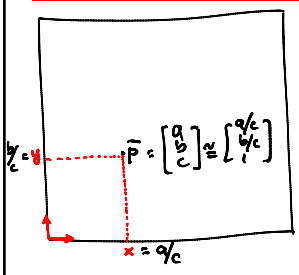
there is a $\lambda \neq 0$ such that

$$\begin{bmatrix}x\\y\\w\end{bmatrix} = \lambda\begin{bmatrix}x'\\y'\\w'\end{bmatrix}$$

Examples:

- Is $\begin{bmatrix}3\\9\\6\end{bmatrix}\cong\begin{bmatrix}6\\18\\12\end{bmatrix}$? yes (take $\lambda=2$)
- Is $\begin{bmatrix}0\\0\\1\end{bmatrix}\cong\begin{bmatrix}0\\0\\20\end{bmatrix}$? yes (take $\lambda=20$)
- Is $\begin{bmatrix}1\\0\\2\end{bmatrix}\cong\begin{bmatrix}-2\\0\\4\end{bmatrix}$? no!

Homogeneous Coords ⇒ Euclidean Coords



Converting from homogeneous to Euclidean coordinates:

$\begin{bmatrix}a\\b\\c\end{bmatrix}\cong\begin{bmatrix}a/c\\b/c\\1\end{bmatrix}$ represent the same 2D point

$\Leftrightarrow$ 2D coordinates are $\begin{bmatrix}a/c\\b/c\end{bmatrix}$

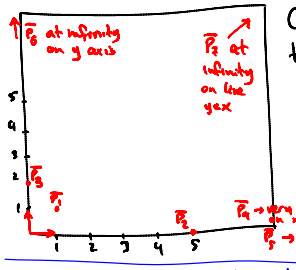
$\bar{v}_1 \cong \bar{v}_2$

$\Leftrightarrow$

there is a $\lambda \neq 0$ such that

$$\begin{bmatrix}x\\y\\w\end{bmatrix} = \lambda\begin{bmatrix}x'\\y'\\w'\end{bmatrix}$$

Homogeneous Coords ⇒ Euclidean Coords



Converting from homogeneous to Euclidean coordinates:

$\begin{bmatrix} a \\ b \\ c \end{bmatrix} \sim \begin{bmatrix} a/c \\ b/c \\ 1 \end{bmatrix}$ represent the same 2D point

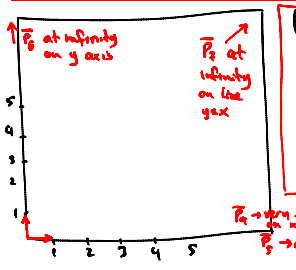
$\Leftrightarrow$ 2D coordinates are $\begin{bmatrix} a/c \\ b/c \end{bmatrix}$

Practice exercise: Plot positions of the following points

$\bar{P}_6 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$ $\bar{P}_7 = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}$

$\bar{P}_1 = \begin{bmatrix} 3 \\ 2 \\ 2 \end{bmatrix}$ $\bar{P}_2 = \begin{bmatrix} 10 \\ 0 \\ 2 \end{bmatrix}$ $\bar{P}_3 = \begin{bmatrix} 0 \\ 8 \\ 4 \end{bmatrix}$ $\bar{P}_4 = \begin{bmatrix} 1 \\ 0 \\ 0.0001 \end{bmatrix}$ $\bar{P}_5 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$

Points at ∞ in Homogeneous Coordinates



Useful property #1:
Even points infinitely far away have a finite representation in homogeneous coords!

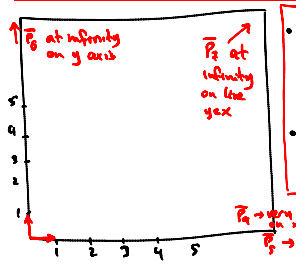
leads to very stable geometric computations

Points at infinity have their last coordinate equal to zero

$\bar{P}_6 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$ $\bar{P}_7 = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}$

$\bar{P}_4 = \begin{bmatrix} 1 \\ 0 \\ 0.0001 \end{bmatrix}$ $\bar{P}_5 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$

Points at ∞ in Homogeneous Coordinates



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

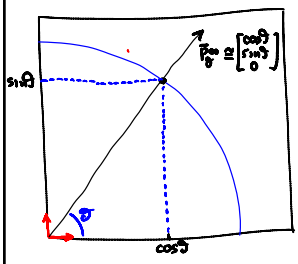
Points at infinity have their last coordinate equal to zero

$\bar{P}_6 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$ $\bar{P}_7 = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}$

first two coords represent directions in 2D

$\bar{P}_5 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$

Points at ∞ in Homogeneous Coordinates



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

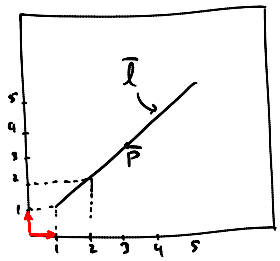
Points at infinity have their last coordinate equal to zero

$\bar{P}_6 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$ $\bar{P}_7 = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}$

first two coords represent directions in 2D

$\bar{P}_5 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$

Line Equations in Homogeneous Coordinates



- The equation of a line

$ax + by + c = 0$ (line parameters)

- In homogeneous coordinates

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

or $\bar{l}^T \cdot \bar{P} = 0$

vector holding line parameters

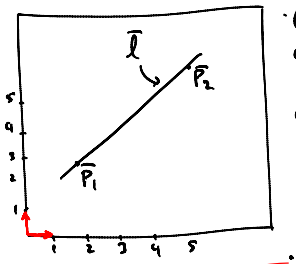
vector holding homogeneous coordinates of a point

Example: line $y = x$ in homogeneous coords:

$1 \cdot x - 1 \cdot y + 0 \cdot 1 = 0$

line parameters of $\bar{l}$: $\begin{bmatrix} 1 & -1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = 0$

The Line Passing Through 2 Points



- Calculating the parameters of a line through two points with homogeneous coordinates $\bar{P}_1, \bar{P}_2$

$\bar{l} = \bar{P}_1 \times \bar{P}_2$ (cross product of two 3D vectors)

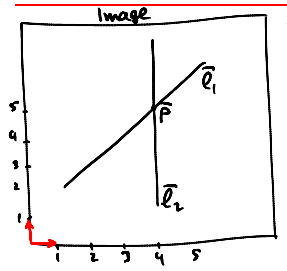
- $\bar{l}$ must satisfy $\bar{l}^T \cdot \bar{P}_1 = 0$, $\bar{l}^T \cdot \bar{P}_2 = 0$
- taken as 3D vectors, $\bar{l}$ is perpendicular to both $\bar{P}_1$ and $\bar{P}_2$ ⇒ it is along the cross product, $\bar{P}_1 \times \bar{P}_2$

- In homogeneous coordinates

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

or $\bar{l}^T \cdot \bar{P} = 0$

The Point of Intersection of Two Lines



- Calculating the homogeneous coordinates of the intersection of two lines $\bar{l}_1, \bar{l}_2$

$$\bar{p} = \bar{l}_1 \times \bar{l}_2$$

↑ cross product of two 3D vectors

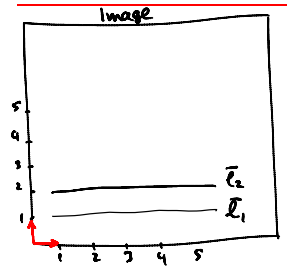
- In homogeneous coordinates

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

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

- p must satisfy $\bar{l}_1^T \bar{p} = 0$, $\bar{l}_2^T \bar{p} = 0$
- taken as 3D vectors, $\bar{p}$ is perpendicular to both $\bar{l}_1$ and $\bar{l}_2$ ⇒ it is along the cross product, $\bar{l}_1 \times \bar{l}_2$

Computing the Intersection of Parallel Lines



- Calculating the homogeneous coordinates of the intersection of two lines $\bar{l}_1, \bar{l}_2$

$$\bar{p} = \bar{l}_1 \times \bar{l}_2$$

↑ cross product of two 3D vectors

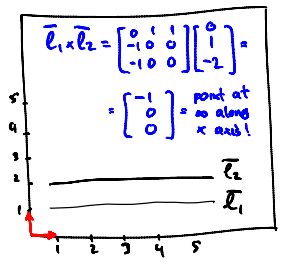
This calculation works even when $\bar{l}_1, \bar{l}_2$ are parallel!

(no floating point exceptions or divide-by-zero errors!)

Computing the Intersection of Parallel Lines

$$\bar{l}_1 \times \bar{l}_2 = \begin{bmatrix} 0 & 1 & 1 \\ -1 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ -2 \end{bmatrix} = \begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix}$$

point at ∞ along x axis!



- Calculating the homogeneous coordinates of the intersection of two lines $\bar{l}_1, \bar{l}_2$

$$\bar{p} = \bar{l}_1 \times \bar{l}_2$$

↑ cross product of two 3D vectors

- Line eq. of $\bar{l}_1$ is $y=1$. Also written as $0 \cdot x + 1 \cdot y - 1 = 0$. So $\bar{l}_1 = \begin{bmatrix} 0 \\ 1 \\ -1 \end{bmatrix}$
- Similarly $\bar{l}_2 = \begin{bmatrix} 0 \\ 1 \\ -2 \end{bmatrix}$

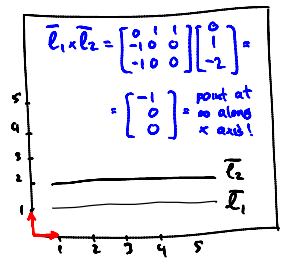
Aside (Calculating cross products): If $\bar{l}_2 = (a, b, c)$ then

$$\bar{l}_1 \times \bar{l}_2 = \begin{bmatrix} 0 & -c & b \\ c & 0 & -a \\ -b & a & 0 \end{bmatrix} \bar{l}_2$$

Computing the Intersection of Parallel Lines

$$\bar{l}_1 \times \bar{l}_2 = \begin{bmatrix} 0 & 1 & 1 \\ -1 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ -2 \end{bmatrix} = \begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix}$$

point at ∞ along x axis!



- Calculating the homogeneous coordinates of the intersection of two lines $\bar{l}_1, \bar{l}_2$

$$\bar{p} = \bar{l}_1 \times \bar{l}_2$$

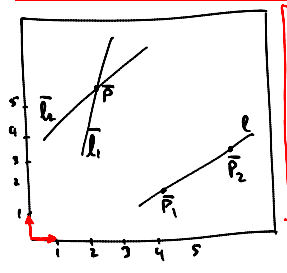
↑ cross product of two 3D vectors

- Line eq. of $\bar{l}_1$ is $y=1$. Also written as $0 \cdot x + 1 \cdot y - 1 = 0$. So $\bar{l}_1 = \begin{bmatrix} 0 \\ 1 \\ -1 \end{bmatrix}$
- Similarly $\bar{l}_2 = \begin{bmatrix} 0 \\ 1 \\ -2 \end{bmatrix}$

Aside (Calculating cross products):

$$\mathbf{a} \times \mathbf{b} = \det \begin{bmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \end{bmatrix}.$$

Lines from Points & Points from Lines



Useful property #2

- Very simple way of computing 2 intersecting lines
- Numerical stability even when result is at ∞

Line through 2 points

$$\bar{l} = \bar{p}_1 \times \bar{p}_2 = \begin{bmatrix} 0 & -p_1^z & p_1^y \\ p_1^z & 0 & -p_1^x \\ -p_1^y & p_1^x & 0 \end{bmatrix} \begin{bmatrix} p_2^x \\ p_2^y \\ p_2^z \end{bmatrix}$$

Intersection of 2 lines

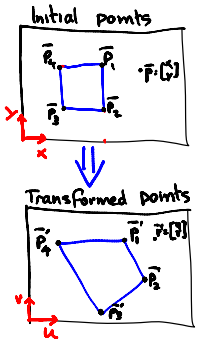
$$\bar{p} = \bar{l}_1 \times \bar{l}_2 = \begin{bmatrix} 0 & -l_1^z & l_1^y \\ l_1^z & 0 & -l_1^x \\ -l_1^y & l_1^x & 0 \end{bmatrix} \begin{bmatrix} l_2^x \\ l_2^y \\ l_2^z \end{bmatrix}$$

Topic 3:

2D Transformations

- Homogeneous coordinates
- Homogeneous 2D transformations
- Affine transformations & restrictions

2D Transformations



Definition:

A function $\bar{f}: \mathbb{R}^2 \to \mathbb{R}^2$

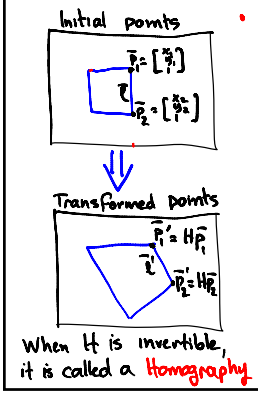
$\bar{p} \mapsto f(\bar{p})$

- Usually f is invertible.
- In this case it can be thought of as a change in coordinates

$\bar{p} = \begin{bmatrix} x \\ y \end{bmatrix} \xrightarrow{f} \bar{p}' = \begin{bmatrix} u \\ v \end{bmatrix}$, with $\bar{f}^{-1}$ mapping back

- **Applications:**
  - Compare objects with parts
  - Shape deformation
  - Animation

General Linear 2D Transformations



When H is invertible, it is called a **Homography**

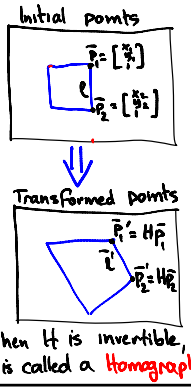
- **Definition (Linear 2D Transforms)**

A 2D transform is called linear if every 2D line $\bar{l}$ in the original image is transformed into a line $\bar{l}'$ in the warped image (i.e. the warp preserves all lines in the original photo)

- **Property (w/out proof)**

Every linear warp can be expressed as a 3x3 matrix H that transforms homogeneous image coordinates

General Linear 2D Transformations



When H is invertible, it is called a **Homography**

- So our focus will be on transformations $f()$ for which

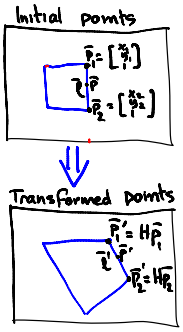
$$f(\bar{p}) = H\bar{p}$$

for some homography matrix H

- **Property (w/out proof)**

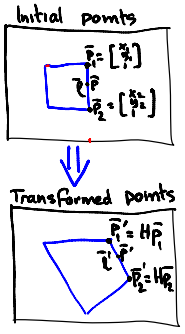
Every linear warp can be expressed as a 3x3 matrix H that transforms homogeneous image coordinates

Homographies: Basic properties



- Homographies transform lines to lines

Homographies: Basic properties



- Homographies transform lines to lines

Proof:

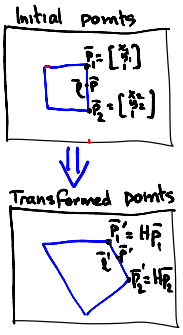
- All points on line $\bar{l}$ satisfy

$$\bar{l}^T \cdot \bar{p} = 0 \quad (*)$$

- The homography H will transform p to Hp.

Why can't we just multiply H into the line equation?

Homographies: Basic properties



- Homographies transform lines to lines

Proof:

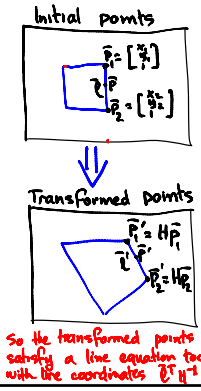
- All points on line $\bar{l}$ satisfy

$$\bar{l}^T \cdot \bar{p} = 0 \quad (*)$$

- The homography H will transform $\bar{p}$ to $H\bar{p}$. Therefore,

$$\bar{p}' \cong H\bar{p} \iff \bar{p} \cong H^{-1}\bar{p}'$$

Homographies: Basic properties



- Homographies transform lines to lines

Proof:

. All points on line ℓ satisfy

$$\bar{\ell}^T \cdot \bar{p} = 0 \qquad (*)$$

. The homography H will transform $\bar{p}$ to $H\bar{p}$. Therefore,

$$\bar{p}' \cong H\bar{p} \iff \bar{p} \cong H^{-1}\bar{p}'$$

. Combining with (*) we get

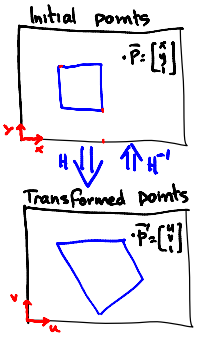
$$\bar{\ell}^T \cdot H^{-1}\bar{p}' = 0 \iff$$

$$(\bar{\ell}^T H^{-1}) \cdot \bar{p}' = 0$$

So the transformed points satisfy a line equation too, with line coordinates $\bar{\ell}^T H^{-1}$

Q.E.D.

Homographies: Basic properties



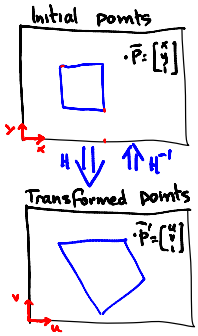
- Homographies transform lines to lines
- Scaling the homography matrix H does not affect the transformation

$$(\lambda \cdot H)\bar{p} = H \cdot (\lambda \bar{p}) \cong H\bar{p}$$

- H is easy to go back & forth between the original & transformed points

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \cong H \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \qquad \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \cong H^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

Homographies: Basic properties



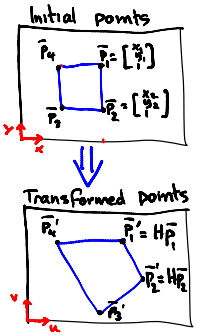
- Homographies are associative

$$H_2\left(H_1\bar{p}\right) = \left(H_2 H_1\right)\bar{p}$$

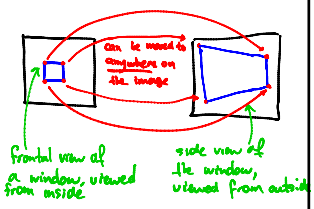
- Homographies are <u>not</u> commutative in general

$$H_2\left(H_1\bar{p}\right) \neq H_1\left(H_2\bar{p}\right)$$

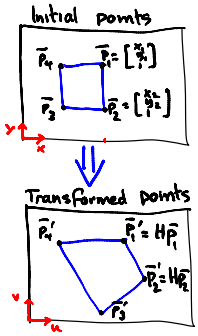
Homographies: Geometric Intuition



Linear warps correspond to every possible distortion of a square created by moving its vertices to arbitrary locations

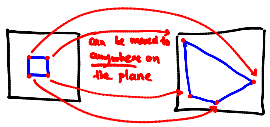


Homographies from Point Correspondences

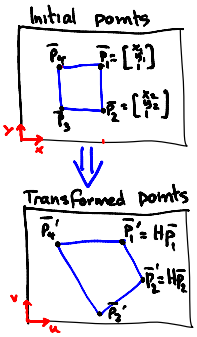


- Intuition:

If we have a correspondence between 4 points in the two images, we can compute H

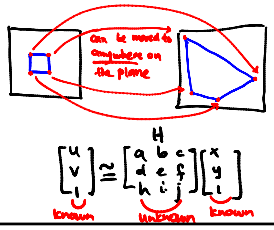


Homographies from Point Correspondences

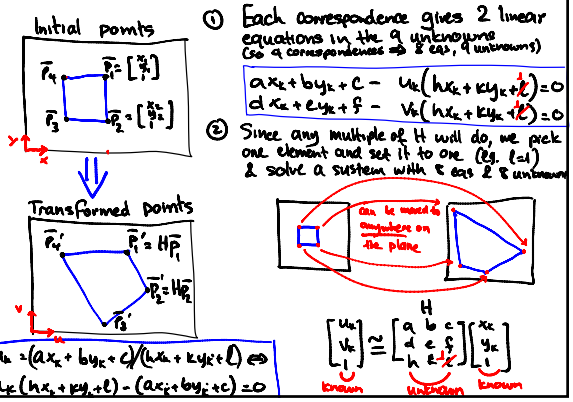


- Intuition:

If we have a correspondence between 4 points in the two images, we can compute H:



Homographies from Point Correspondences

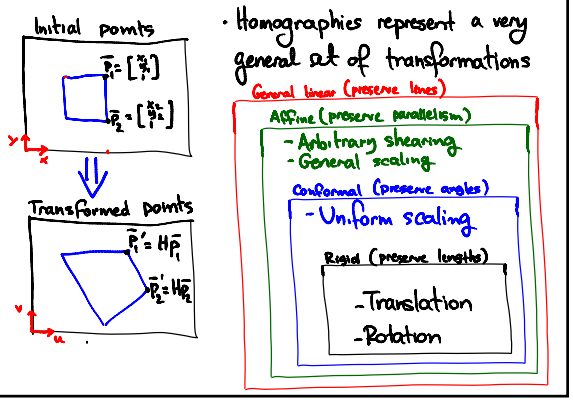


Topic 3:

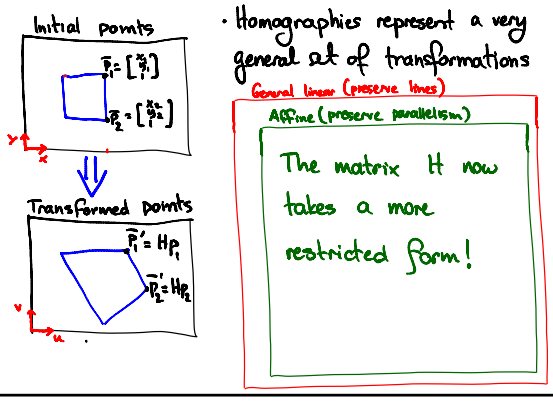
2D Transformations

- Homogeneous coordinates
- Homogeneous 2D transformations
- Affine transformations & restrictions

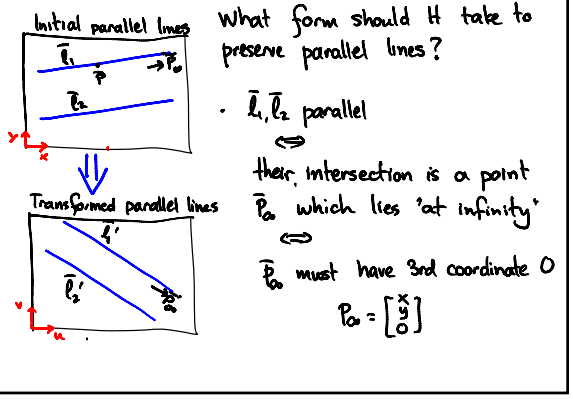
General Linear 2D Transformations



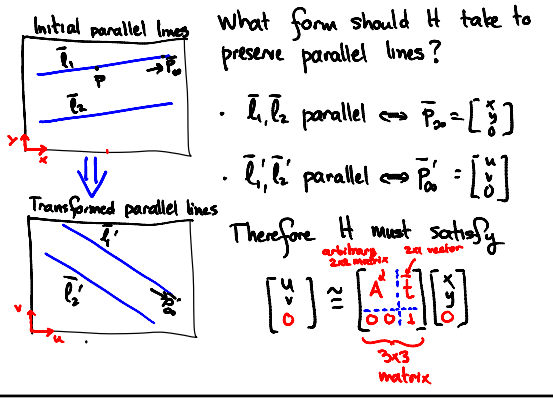
Affine Transformations



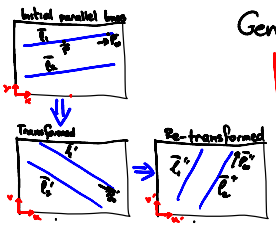
Affine Transformations: General Matrix Form



Affine Transformations: General Matrix Form



Affine Transformations: Basic Properties



General form of matrix H

$$H = \begin{bmatrix} A & \bar{t} \\ 0\ 0 & 1 \end{bmatrix}$$

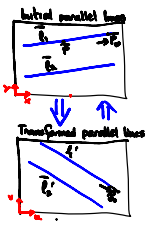
(arbitrary 2x2 matrix; 2x1 vector)

1. Affine transforms are closed under composition:

this is an affine transformation!

$$\begin{bmatrix} A_2 & \bar{t}_2 \\ 0\ 0 & 1 \end{bmatrix}\begin{bmatrix} A_1 & \bar{t}_1 \\ 0\ 0 & 1 \end{bmatrix} = \begin{bmatrix} A_2A_1 + \bar{t}_2[0\ 0] & A_2\bar{t}_1 + \bar{t}_2 1 \\ [0\ 0]A_1 + 1\cdot[0\ 0] & [0\ 0]\bar{t}_1 + 1\cdot 1 \end{bmatrix} = \begin{bmatrix} A_2A_1 & A_2\bar{t}_1 + \bar{t}_2 \\ 0\ 0 & 1 \end{bmatrix}$$

Affine Transformations: Basic Properties



General form of matrix H

$$H = \begin{bmatrix} A & \bar{t} \\ 0\ 0 & 1 \end{bmatrix}$$

(arbitrary 2x2 matrix; 2x1 vector)

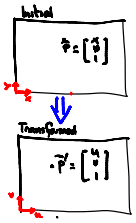
1. Affine transforms are closed under composition:

If H_1, H_2 are affine transform matrices, so is $H_1 \cdot H_2$.

2. The inverse H^{-1} of an affine transform H is affine

Proof: By definition, H^{-1} will map $\bar{P}_\infty$ to $\bar{P}_\infty$, Since it preserves points at infinity, its matrix must have the above form. QED.

Affine Transformations: Basic Properties



General form of matrix H

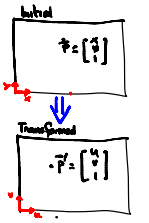
$$H = \begin{bmatrix} A & \bar{t} \\ 0\ 0 & 1 \end{bmatrix}$$

(arbitrary 2x2 matrix; 2x1 vector)

3. Affine transforms preserve the value of the last homogeneous coordinate

$$\begin{bmatrix} A & \bar{t} \\ 0\ 0 & 1 \end{bmatrix}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} A\begin{bmatrix} x \\ y \end{bmatrix} + \bar{t} \\ [0\ 0]\begin{bmatrix} x \\ y \end{bmatrix} + 1 \end{bmatrix} = \begin{bmatrix} A\begin{bmatrix} x \\ y \end{bmatrix} + \bar{t} \\ 1 \end{bmatrix}$$

Affine-Transforming 2D Points



General form of matrix H

$$H = \begin{bmatrix} A & \bar{t} \\ 0\ 0 & 1 \end{bmatrix}$$

(arbitrary 2x2 matrix; 2x1 vector)

3. Affine transforms preserve the value of the last homogeneous coordinate

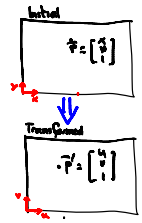
$$\begin{bmatrix} A & \bar{t} \\ 0\ 0 & 1 \end{bmatrix}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} A\begin{bmatrix} x \\ y \end{bmatrix} + \bar{t} \\ 1 \end{bmatrix}$$

(becomes equality)

Transforming Euclidean points: (i.e. non-homogeneous)

1. append a 3rd coord of 1 $\begin{bmatrix} x \\ y \end{bmatrix} \rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$
2. apply H: $\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = H\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$
3. delete 3rd coord of result

Affine-Transforming 2D Points



General form of matrix H

$$H = \begin{bmatrix} A & \bar{t} \\ 0\ 0 & 1 \end{bmatrix}$$

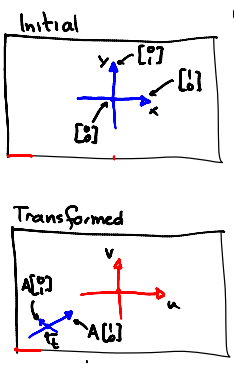
(arbitrary 2x2 matrix; 2x1 vector)

$$\begin{bmatrix} x \\ y \end{bmatrix} \xrightarrow{H} \begin{bmatrix} u \\ v \end{bmatrix} = A\begin{bmatrix} x \\ y \end{bmatrix} + \bar{t}$$

Transforming Euclidean points: (i.e. non-homogeneous)

1. append a 3rd coord of 1 $\begin{bmatrix} x \\ y \end{bmatrix} \rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$
2. apply H: $\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = H\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$
3. delete 3rd coord of result

Affine-Transforming 2D Points



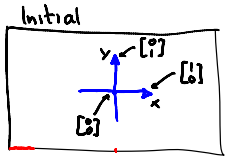
General form of matrix H

$$H = \begin{bmatrix} A & \bar{t} \\ 0\ 0 & 1 \end{bmatrix}$$

(arbitrary 2x2 matrix; 2x1 vector)

$$\begin{bmatrix} x \\ y \end{bmatrix} \xrightarrow{H} \begin{bmatrix} u \\ v \end{bmatrix} = A\begin{bmatrix} x \\ y \end{bmatrix} + \bar{t}$$

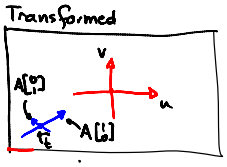
Geometric Interpretation of Affine Matrix



General form of matrix H

$$H = \begin{bmatrix} A & \bar{t} \\ 0\ 0 & 1 \end{bmatrix}$$

(A: arbitrary 2x2 matrix; $\bar{t}$: 2x1 vector)



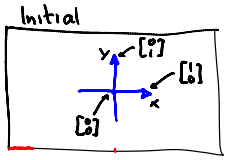
$A\begin{bmatrix}1\\0\end{bmatrix} = \begin{bmatrix}a & b\\ c & d\end{bmatrix}\begin{bmatrix}1\\0\end{bmatrix} = \begin{bmatrix}a\\c\end{bmatrix}$

$A\begin{bmatrix}0\\1\end{bmatrix} = \begin{bmatrix}a & b\\ c & d\end{bmatrix}\begin{bmatrix}0\\1\end{bmatrix} = \begin{bmatrix}b\\d\end{bmatrix}$

1st column of A ⇒ transforms the x-axis

2nd column of A ⇒ transforms the y-axis

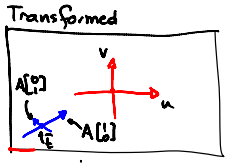
Geometric Interpretation of Affine Matrix



General form of matrix H

$$H = \begin{bmatrix} A & \bar{t} \\ 0\ 0 & 1 \end{bmatrix}$$

(A: arbitrary 2x2 matrix; $\bar{t}$: 2x1 vector)

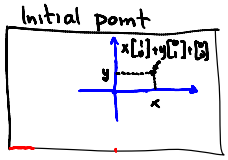


$\bar{t}$ ⇒ translates the origin to point $\bar{t}$

1st column of A ⇒ transforms the x-axis

2nd column of A ⇒ transforms the y-axis

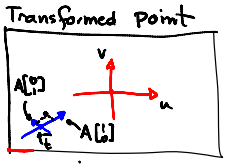
Geometric Interpretation of Affine Matrix



General form of matrix H

$$H = \begin{bmatrix} A & \bar{t} \\ 0\ 0 & 1 \end{bmatrix}$$

(A: arbitrary 2x2 matrix; $\bar{t}$: 2x1 vector)

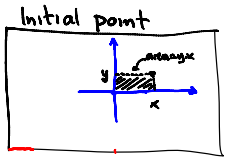


$\bar{t}$ ⇒ translates the origin to point $\bar{t}$

1st column of A ⇒ transforms the x-axis

2nd column of A ⇒ transforms the y-axis

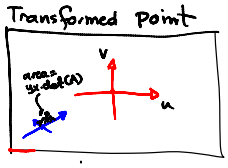
How Affine Transformations Affect Area



General form of matrix H

$$H = \begin{bmatrix} A & \bar{t} \\ 0\ 0 & 1 \end{bmatrix}$$

(A: arbitrary 2x2 matrix; $\bar{t}$: 2x1 vector)

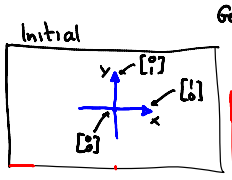


The area of any closed region will be multiplied by det(A) after the transformation

$\det\begin{bmatrix}a & b\\ c & d\end{bmatrix} = ad - bc$

⇒ If A singular (non-invertible) region is 'squashed' to zero

From Affine to Rigid Transformations

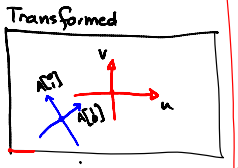


General: $\begin{bmatrix} A & \bar{t} \\ * & * & 1 \end{bmatrix}$ Affine: $\begin{bmatrix} A & \bar{t} \\ 0\ 0 & 1 \end{bmatrix}$

General linear (preserve lines)

Affine (preserve parallelism)

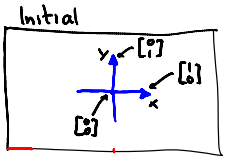
- Arbitrary shearing
- General scaling



Rigid (preserve lengths)

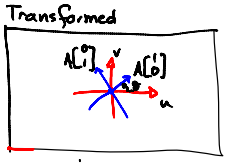
- Matrix A takes a special form!
- $\bar{t}$ can be arbitrary

Rigid Transformations: Rotations



- Rigid (a.k.a "Euclidean") transforms are composed of 2D rotations and/or translations $\begin{bmatrix} A & \bar{t} \\ 0\ 0 & 1 \end{bmatrix}$
- Rotation by a counter-clockwise angle θ: about origin

$$A = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

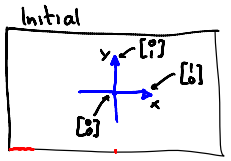


- "Pure" rotation transformation

$$\begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

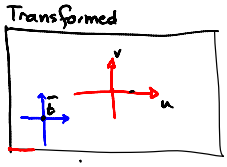
no translation of origin

Rigid Transformations: Translations



- Rigid (aka "Euclidean") transforms are composed of 2D rotations and/or translations $\begin{bmatrix} A & \bar{t} \\ 0\,0 & 1 \end{bmatrix}$
- Translation by vector $\bar{t}$

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

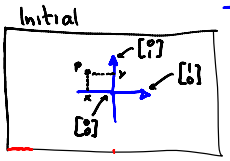


- "Pure" translation transformation

$$\begin{bmatrix} 1 & 0 & \bar{t} \\ 0 & 1 & \\ 0\,0 & & 1 \end{bmatrix}$$

no rotation

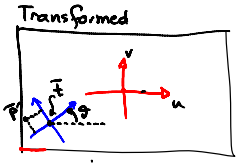
Composition of 2D Rotations & Translations



Example 1: Rotation followed by translation

- First rotate about θ:

$$\bar{p} \rightarrow \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \bar{p}$$

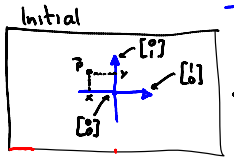


- Then translate the result by vector $\bar{t}$:

$$\bar{p}' = \begin{bmatrix} 1 & 0 & \bar{t} \\ 0 & 1 & \\ 0\,0 & & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \bar{p}$$

$$= \begin{bmatrix} \cos\theta & -\sin\theta & \bar{t} \\ \sin\theta & \cos\theta & \\ 0 & 0 & 1 \end{bmatrix} \bar{p}$$

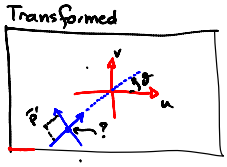
Composition of 2D Translations & Rotations



Example 2: Translation followed by rotation

- First translate by $\bar{t}$

$$\bar{p} \rightarrow \begin{bmatrix} 1\,0 & \bar{t} \\ 0\,1 & \\ 0\,0 & 1 \end{bmatrix} \bar{p}$$

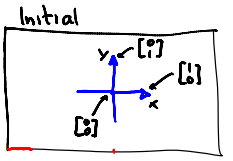


- Then rotate the result by θ

$$\bar{p}' = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1\,0 & \bar{t} \\ 0\,1 & \\ 0\,0 & 1 \end{bmatrix} \bar{p}$$

$$= ?$$

From Affine to Conformal Transformations



Affine: $\begin{bmatrix} A & \bar{t} \\ 0\,0 & 1 \end{bmatrix}$

General linear (preserve lines)

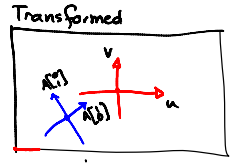
Affine (preserve parallelism)

- Arbitrary shearing
- General scaling

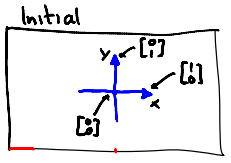
Conformal (preserve angles)

- Uniform scaling
- Reflection

- Translation
- Rotation

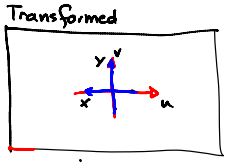


Conformal Transformations: Reflection

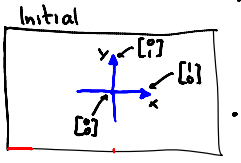


- Conformal transforms include reflections and uniform scalings $\begin{bmatrix} A & \bar{t} \\ 0\,0 & 1 \end{bmatrix}$
- Pure reflection (about y)

$$A = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \rightsquigarrow \begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

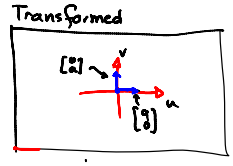


Conformal Transformations: Uniform Scaling



- Conformal transforms include reflections and uniform scalings $\begin{bmatrix} A & \bar{t} \\ 0\,0 & 1 \end{bmatrix}$
- Pure reflection (about y)

$$A = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \rightsquigarrow \begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



- "Pure" uniform scaling

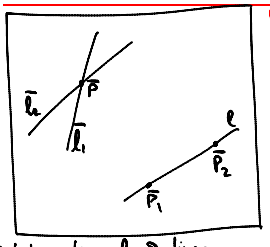
$$\begin{bmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Topic 4:

Coordinate-Free Geometry (CFG)

- A brief introduction & basic ideas

Doing Geometry Without Coordinates



- Style of expressing geometric objects & relations that avoids reliance on a coordinate system
- Useful in CG where we often deal with many coord systems

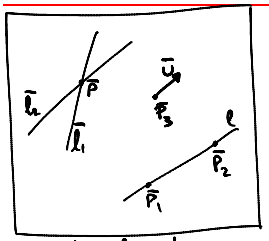
(#1) Intersection of 2 lines

$\bar{p} = \bar{l}_1 \times \bar{l}_2$

(#2) Line through 2 points

$\bar{l} = \bar{p}_1 \times \bar{p}_2$

CFG: Key Objects & their Homogeneous Repr.



Key objects:

- Points $\bar{p}$ $\begin{bmatrix} x \\ y \\ w \end{bmatrix}, w \neq 0$
- Lines $\bar{l}$ $\begin{bmatrix} a \\ b \\ c \end{bmatrix}$
- Vectors $\bar{v}$ $\begin{bmatrix} \alpha \\ \beta \\ 0 \end{bmatrix}$

all represented as homogeneous 3-vectors

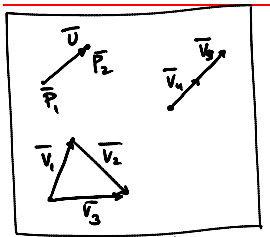
(#1) Intersection of 2 lines

$\bar{p} = \bar{l}_1 \times \bar{l}_2$

(#2) Line through 2 points

$\bar{l} = \bar{p}_1 \times \bar{p}_2$

CFG: Basic Geometric Operations



Key objects:

- Points $\bar{p}$ $\begin{bmatrix} x \\ y \\ w \end{bmatrix}, w \neq 0$
- Lines $\bar{l}$ $\begin{bmatrix} a \\ b \\ c \end{bmatrix}$
- Vectors $\bar{v}$ $\begin{bmatrix} \alpha \\ \beta \\ 0 \end{bmatrix}$

(#1) Intersection of 2 lines

$\bar{p} = \bar{l}_1 \times \bar{l}_2$

(#2) Line through 2 points

$\bar{l} = \bar{p}_1 \times \bar{p}_2$

(#3) Point-vector addition

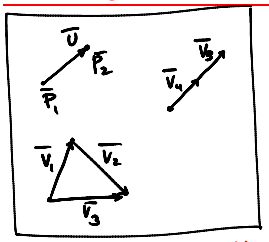
$\bar{p}_2 = \bar{p}_1 + \bar{v}$

(#4) Vector-vector addition

$\bar{v}_3 = \bar{v}_1 + \bar{v}_2$

(#5) Vector scaling: $\bar{v}_5 = \lambda \bar{v}_4$

CFG: "Legal" vs. "Undefined" Geometric Ops



Key objects:

- Points $\bar{p}$ $\begin{bmatrix} x \\ y \\ w \end{bmatrix}, w \neq 0$
- Lines $\bar{l}$ $\begin{bmatrix} a \\ b \\ c \end{bmatrix}$
- Vectors $\bar{v}$ $\begin{bmatrix} \alpha \\ \beta \\ 0 \end{bmatrix}$

CAUTION: Addition possible only when 3rd homogeneous coordinate not affected

e.g. $\bar{p}_1 + \bar{p}_2$ = UNDEFINED!

(#3) Point-vector addition

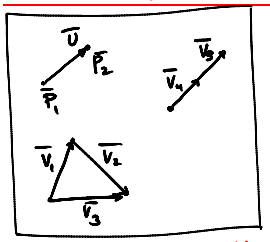
$\bar{p}_2 = \bar{p}_1 + \bar{v}$

(#4) Vector-vector addition

$\bar{v}_3 = \bar{v}_1 + \bar{v}_2$

(#5) Vector scaling: $\bar{v}_5 = \lambda \bar{v}_4$

More CFG Ops: Linear Vector Combination



Key objects:

- Points $\bar{p}$ $\begin{bmatrix} x \\ y \\ w \end{bmatrix}, w \neq 0$
- Lines $\bar{l}$ $\begin{bmatrix} a \\ b \\ c \end{bmatrix}$
- Vectors $\bar{v}$ $\begin{bmatrix} \alpha \\ \beta \\ 0 \end{bmatrix}$

CAUTION: Addition possible only when 3rd homogeneous coordinate not affected

(#6) Linear vector combination

$\bar{v} = \sum_{i=1}^{n} \lambda_i \bar{v}_i$

(#3) Point-vector addition

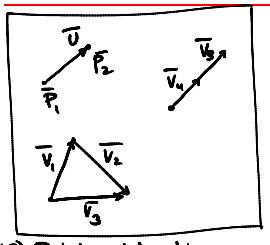
$\bar{p}_2 = \bar{p}_1 + \bar{v}$

(#4) Vector-vector addition

$\bar{v}_3 = \bar{v}_1 + \bar{v}_2$

(#5) Vector scaling: $\bar{v}_5 = \lambda \bar{v}_4$

More CFG Ops: Affine Point Combination



Key objects:

- Points $\bar{p}$ $\begin{bmatrix} x \\ y \\ w \end{bmatrix}, w \neq 0$
- Lines $\bar{l}$ $\begin{bmatrix} a \\ b \\ c \end{bmatrix}$
- Vectors $\bar{v}$ $\begin{bmatrix} x \\ y \\ 0 \end{bmatrix}$

(#3) Point subtraction

$\bar{v} = \bar{P}_2 - \bar{P}_1$ only when $\bar{P}_1, \bar{P}_2$ have same 3rd coord!

(#4) Point-vector addition

$\bar{P}_2 = \bar{P}_1 + \bar{v}$

(#5) Affine point combination

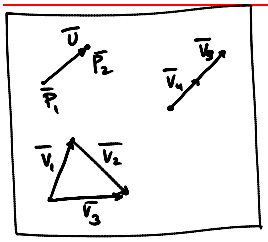
$\bar{P} = \sum_{i=1}^{k} \lambda_i \bar{P}_i$ only when all $\bar{P}_i$ have same 3rd coord

$= \bar{P}_1 + (\lambda_1 - 1)\bar{P}_1 + \sum_{i=2}^{k} \lambda_i \bar{P}_i$

AND

- $\sum_{i=1}^{k} \lambda_i = 1$ i.e. circled expression is a vector $\Rightarrow$ reduces to (#3)
- OR $\sum_{i=1}^{k} \lambda_i = 0$ i.e. circled expression is a point $= \bar{Q}$ with same 3rd coord $\Rightarrow$ reduces to (#7)

More CFG Ops: Operations w/ Scalar Result



Key objects:

- Points $\bar{p}$ $\begin{bmatrix} x \\ y \\ w \end{bmatrix}, w \neq 0$
- Lines $\bar{l}$ $\begin{bmatrix} a \\ b \\ c \end{bmatrix}$
- Vectors $\bar{v}$ $\begin{bmatrix} x \\ y \\ 0 \end{bmatrix}$

(#9) $\|\bar{v}\|$ magnitude of vector $\bar{v}$

(#10) $\bar{v}_1 \cdot \bar{v}_2$ dot product of two vectors (also written as $(\bar{v}_1)^T(\bar{v}_2)$ in matrix notation)

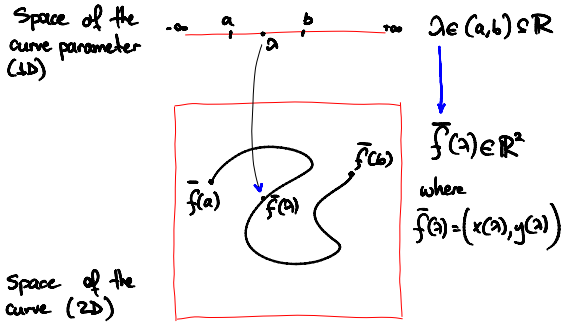
(#11) $\bar{l} \cdot \bar{P}$ dot product of a line and a point

Topic 5:

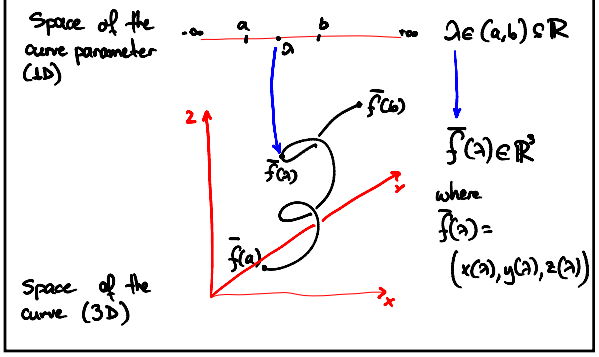
3D Objects

- General curves & surfaces in 3D
- Normal vectors, surface curves & tangent planes
- Implicit surface representations
- Example surfaces:
  surfaces of revolution, bilinear patches, quadrics

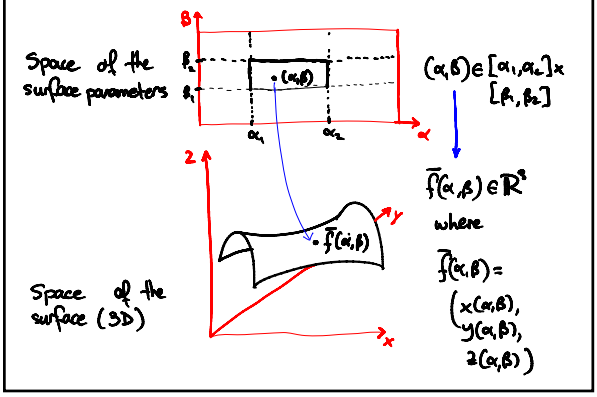
Reminder: Curves in 2D



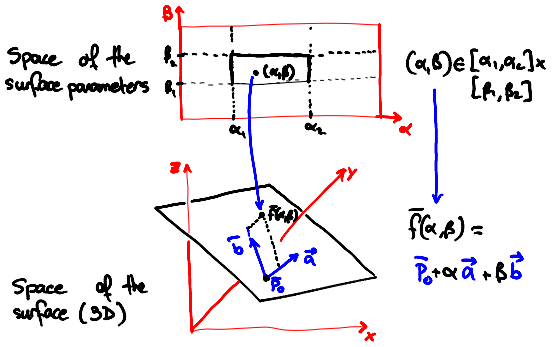
Curves in 3D



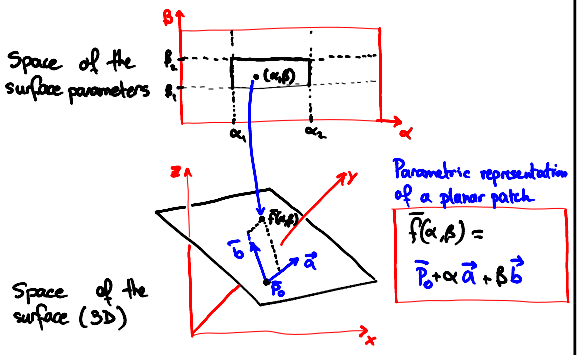
Surfaces in 3D



Surface Example: Planes in 3D



Surface Example: Planes in 3D



Topic 5:

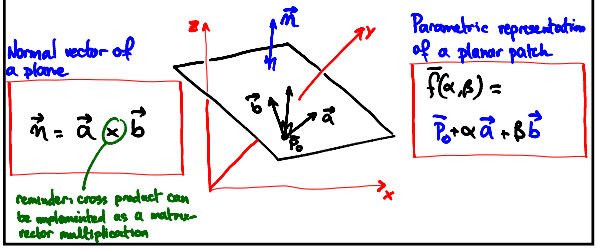
3D Objects

- General curves & surfaces in 3D
- Normal vectors, surface curves & tangent planes
- Implicit surface representations
- Example surfaces:
  surfaces of revolution, bilinear patches, quadrics

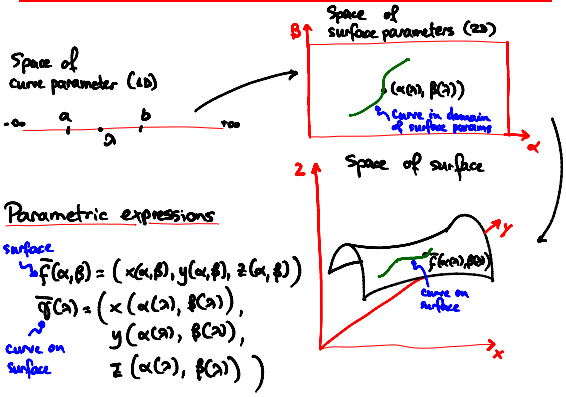
Normal Vector of a Plane

Definition

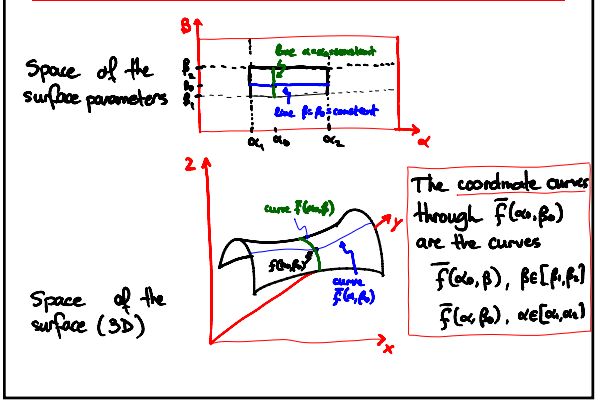
The normal vector of a plane is any vector that is perpendicular to it (ie. normal $\vec{n}$ is such that $\vec{n}\cdot\vec{a}=\vec{n}\cdot\vec{b}=0$)



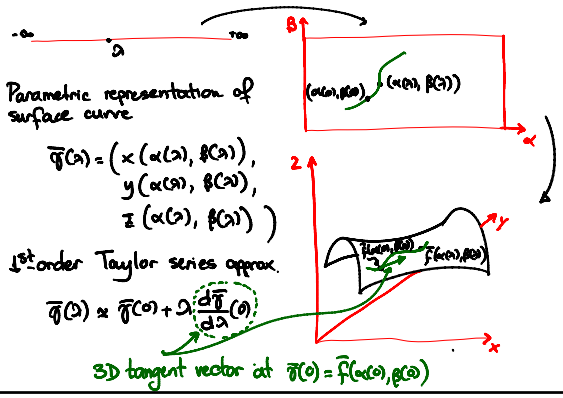
Surface Curves: Parametric Representation



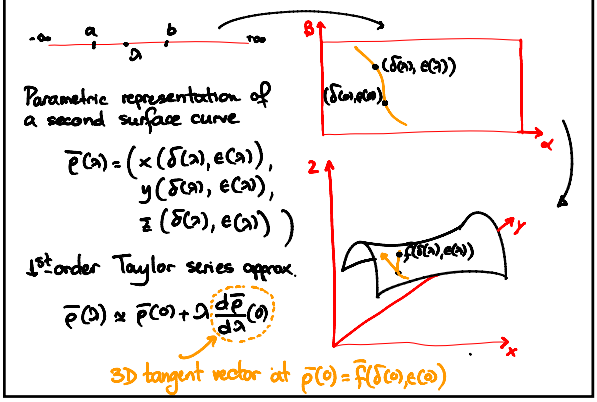
The Coordinate Curves of a Surface



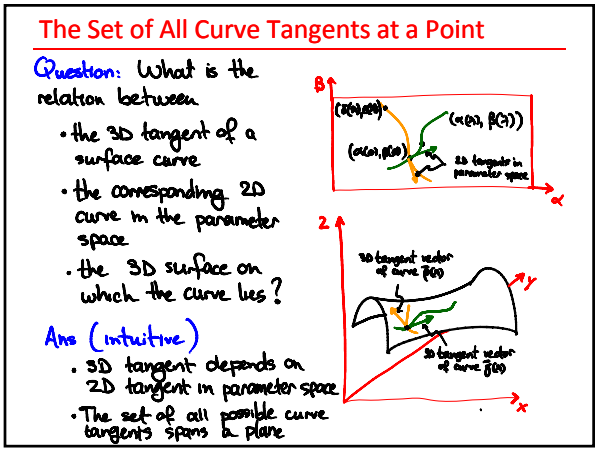
The Tangent Vector of a Surface Curve



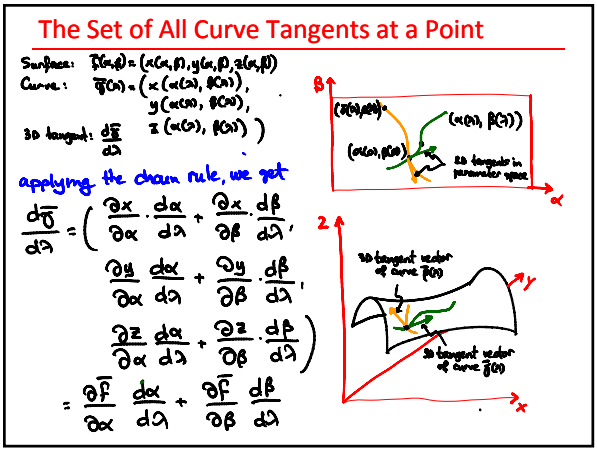
The Tangent Vector of a Surface Curve



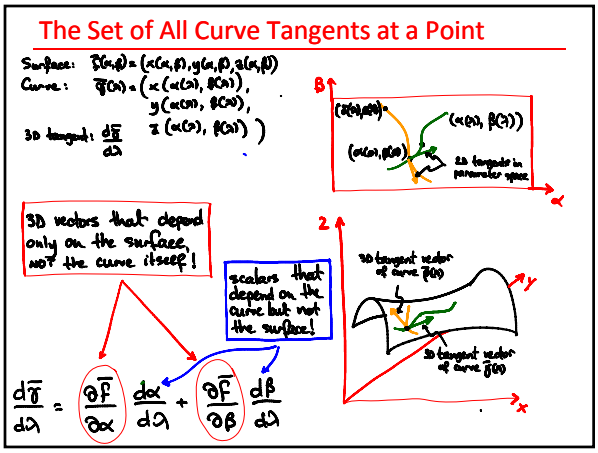
The Set of All Curve Tangents at a Point



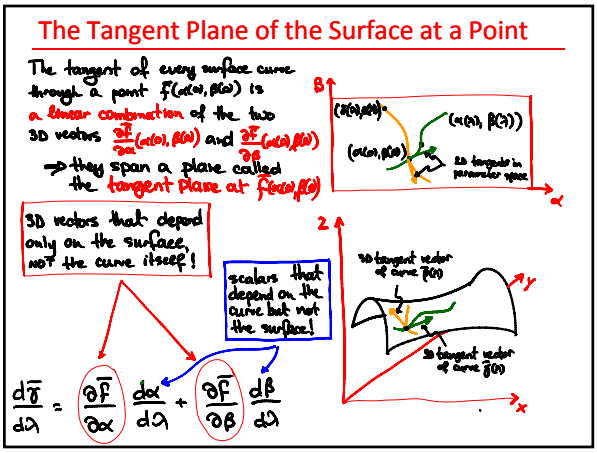
The Set of All Curve Tangents at a Point



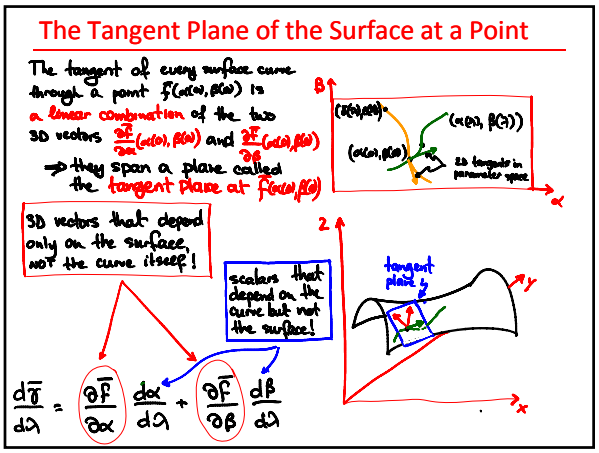
The Set of All Curve Tangents at a Point



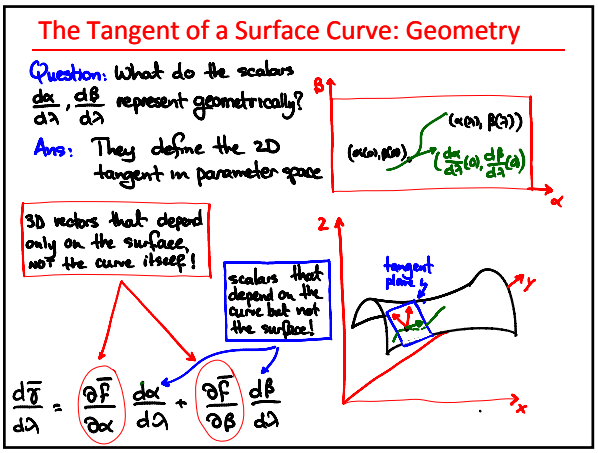
The Tangent Plane of the Surface at a Point



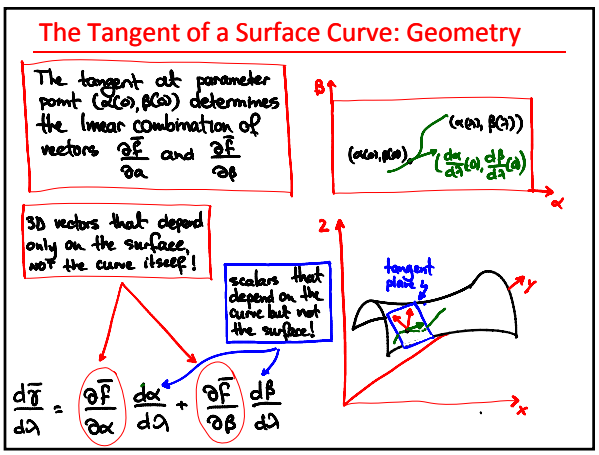
The Tangent Plane of the Surface at a Point



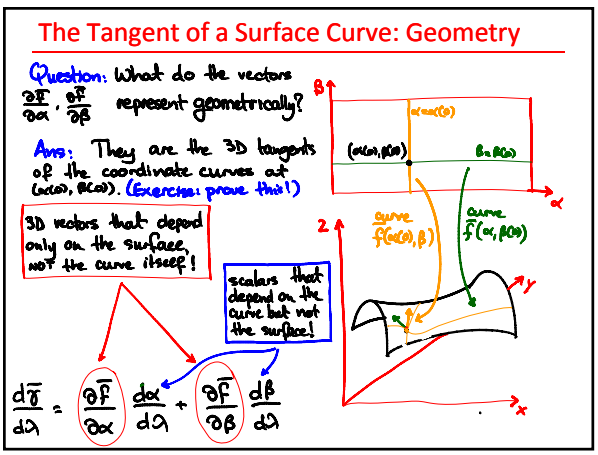
The Tangent of a Surface Curve: Geometry



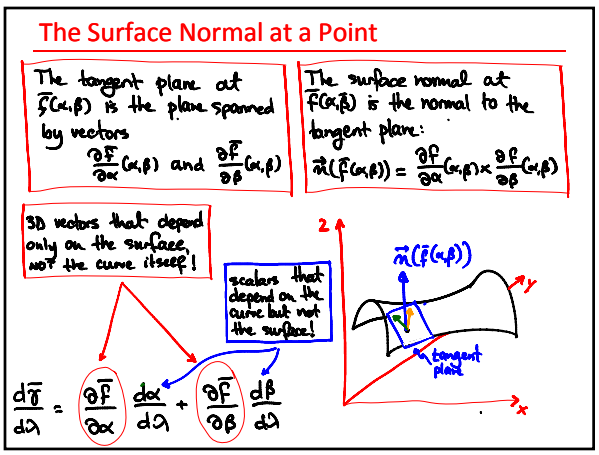
The Tangent of a Surface Curve: Geometry



The Tangent of a Surface Curve: Geometry



The Surface Normal at a Point



Topic 5:

3D Objects

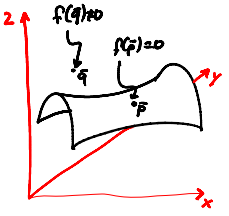
- General curves & surfaces in 3D
- Normal vectors, surface curves & tangent planes
- Implicit surface representations
- Example surfaces:
  surfaces of revolution, bilinear patches, quadrics

Representing Surfaces by an Implicit Function

- Representation consists of a scalar function $f: \mathbb{R}^3 \to \mathbb{R}$ (called the Implicit Function)
- Surface defined as the set

$$\mathcal{A}_0 = \{\bar{p} \in \mathbb{R}^3 \mid f(\bar{p}) = 0\}$$

- Intuitively, f can be thought of as measuring a "distance" to the surface
- Typically, f does NOT measure Euclidean distance to the surface

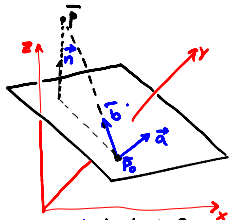


Example: The Implicit Function of a Plane

- Representation consists of a scalar function $f: \mathbb{R}^3 \to \mathbb{R}$
- Surface defined as the set

$$\mathcal{A}_0 = \{\bar{p} \in \mathbb{R}^3 \mid f(\bar{p}) = 0\}$$

- Intuitively, f can be thought of as measuring a "distance" to the surface
- Typically, f does NOT measure Euclidean distance to the surface



Example: Implicit function for a plane through $\bar{p}_0$ with normal $\vec{n}$:

$$f(\bar{p}) = (\bar{p} - \bar{p}_0) \cdot \vec{n}$$

The Level Sets of an Implicit Function

- Representation consists of a scalar function $f: \mathbb{R}^3 \to \mathbb{R}$
- Surface defined as the set

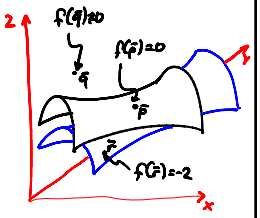
$$\mathcal{A}_0 = \{\bar{p} \in \mathbb{R}^3 \mid f(\bar{p}) = 0\}$$

- Intuitively, f can be thought of as measuring a "distance" to the surface

Given a value $c \in \mathbb{R}$, the set

$$\mathcal{A}_c = \{\bar{p} \in \mathbb{R}^3 \mid f(\bar{p}) = c\}$$

also defines a surface called the c-level set of f



Example: Point $\bar{r}$ belongs to the -2-level set of f.

Surface Normals from the Implicit Function

If $\bar{p}$ is a surface point, the normal at $\bar{p}$ is given by

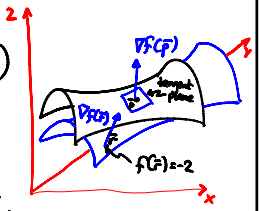
$$\vec{n}(\bar{p}) = \nabla f(\bar{p})$$

where

$$\nabla f(\bar{p}) = \left(\frac{\partial f}{\partial x}(\bar{p}), \frac{\partial f}{\partial y}(\bar{p}), \frac{\partial f}{\partial z}(\bar{p})\right)$$

(a.k.a. the gradient of f)

Note: the above definition works for any level set: if $\bar{r} \in \mathcal{A}_c$, the normal of the c-level-set at point $\bar{r}$ is given by $\nabla f(\bar{r})$



Surface Normals from the Implicit Function

Proof: Let $\bar{q}(\lambda) = (x(\lambda), y(\lambda), z(\lambda))$ be a curve on the surface $\mathcal{A}_c$ with $\bar{q}(0) = \bar{r}$.

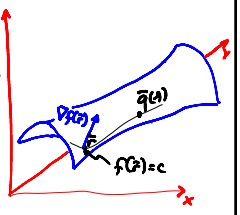
$$\Leftrightarrow \forall \lambda \quad f(\bar{q}(\lambda)) = c$$

$$\Leftrightarrow \frac{df}{d\lambda}(\bar{q}(\lambda)) = 0$$

$$\Leftrightarrow \frac{\partial f}{\partial x}\frac{dx}{d\lambda} + \frac{\partial f}{\partial y}\frac{dy}{d\lambda} + \frac{\partial f}{\partial z}\frac{\partial z}{\partial \lambda} = 0$$

$$\Leftrightarrow \nabla f(\bar{q}(\lambda)) \cdot \frac{d\bar{q}}{d\lambda}(0) = 0$$

Note: the above definition works for any level set: if $\bar{r} \in \mathcal{A}_c$, the normal of the c-level-set at point $\bar{r}$ is given by $\nabla f(\bar{r})$



Surface Normals from the Implicit Function

Proof: Let $\bar{q}(\lambda) = (x(\lambda), y(\lambda), z(\lambda))$ be a curve on the surface $\mathcal{A}_c$ with $\bar{q}(0) = \bar{r}$.

$$\Leftrightarrow \forall \lambda \quad f(\bar{q}(\lambda)) = c$$

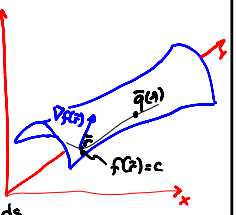
$$\Leftrightarrow \frac{df}{d\lambda}(\bar{q}(\lambda)) = 0$$

$$\Leftrightarrow \frac{\partial f}{\partial x}\frac{dx}{d\lambda} + \frac{\partial f}{\partial y}\frac{dy}{d\lambda} + \frac{\partial f}{\partial z}\frac{\partial z}{\partial \lambda} = 0$$

$$\Leftrightarrow \nabla f(\bar{q}(\lambda)) \cdot \frac{d\bar{q}}{d\lambda}(0) = 0$$

gradient at $\bar{r}$ — 3D tangent at $\bar{q}(0)$

since the above orthogonality holds for any curve in $\mathcal{A}_c$ through $\bar{r}$, the gradient must be perpendicular to the tangent plane QED

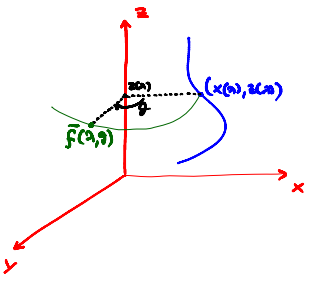


Topic 5:

3D Objects

- General curves & surfaces in 3D
- Normal vectors, surface curves & tangent planes
- Implicit surface representations
- Example surfaces:
  surfaces of revolution, bilinear patches, quadrics

Surfaces of Revolution: Basic Construction



Conceptual steps:

1. Define a 2D curve on the xz-plane:
2. Rotate it about the z-axis

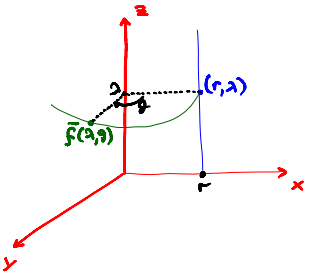
Another equivalent view:

- Point $(x(\lambda), y(\lambda))$ will trace a circle in xy-plane
- The circle's radius is equal to $x(\lambda)$
- θ ranges in $[0, 2\pi]$

Parametric representation

$$\bar{f}(\lambda,\theta) = (x(\lambda)\cos\theta, x(\lambda)\sin\theta, z(\lambda))$$

Example: The Cylinder



Question: how do we express the cylinder of radius r?

Ans:

$$(x(\lambda), z(\lambda)) = (r, \lambda)$$

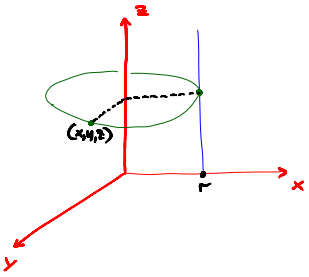
So

$$\bar{f}(\lambda,\theta) = (r\cos\theta, r\sin\theta, \lambda)$$

Parametric representation

$$\bar{f}(\lambda,\theta) = (x(\lambda)\cos\theta, x(\lambda)\sin\theta, z(\lambda))$$

Example: Implicit Function of the Cylinder



Question: how do we express the cylinder of radius r?

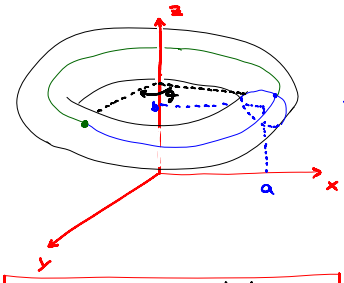
Ans:

The points (x,y,z) on the cylinder have constant distance r from z-axis

Implicit equation

$$f(x,y,z) = x^2 + y^2 - r^2 = 0$$

Example: The Torus as a Surface of Revolution



Question: how do we express the torus as a surface of revolution?

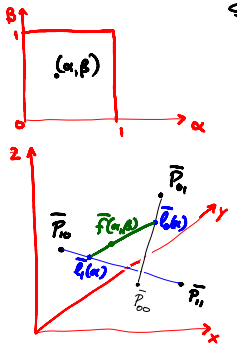
Ans: torus is formed by rotating a circle about the z-axis

$$(x(\lambda), z(\lambda)) = (r\cos\lambda + a, r\sin\lambda + b)$$

Parametric representation

$$\bar{f}(\lambda,\theta) = (x(\lambda)\cos\theta, x(\lambda)\sin\theta, z(\lambda))$$

Bilinear Patches: Basic Construction



Space of parameters

$$(\alpha,\beta) \in [0,1] \times [0,1]$$

Given 4 3D points $\bar{P}_{00}, \bar{P}_{01}, \bar{P}_{10}, \bar{P}_{11}$ the pair (α,β) uniquely determines $\bar{f}(\alpha,\beta)$:

$$\bar{l}_0(\alpha) = (1-\alpha)\bar{P}_{00} + \alpha\bar{P}_{01}$$

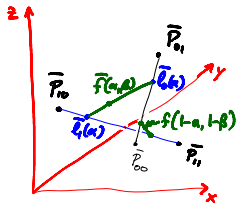
$$\bar{l}_1(\alpha) = (1-\alpha)\bar{P}_{10} + \alpha\bar{P}_{11}$$

$$\bar{f}(\alpha,\beta) = (1-\beta)\bar{l}_0(\alpha) + \beta\bar{l}_1(\alpha)$$

Bilinear Patches: Basic Construction

Question: What kind of surface do we get?
Is it a plane?

Ans: It is a plane only if $\bar{P}_{00}, \bar{P}_{01}, \bar{P}_{10}, \bar{P}_{11}$ are coplanar!



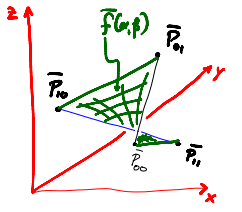
Given 4 3D points $\bar{P}_{00}, \bar{P}_{01}, \bar{P}_{10}, \bar{P}_{11}$ the pair (α,β) uniquely determines $\bar{f}(\alpha,\beta)$:

$$\bar{l}_0(\alpha) = (1-\alpha)\bar{P}_{00} + \alpha\bar{P}_{01}$$
$$\bar{l}_1(\alpha) = (1-\alpha)\bar{P}_{10} + \alpha\bar{P}_{11}$$
$$\bar{f}(\alpha,\beta) = (1-\beta)\bar{l}_0(\alpha) + \beta\bar{l}_1(\alpha)$$

Bilinear Patches: Basic Construction

Question: What kind of surface do we get?
Is it a plane?

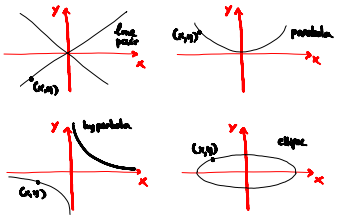
Ans: It is a plane only if $\bar{P}_{00}, \bar{P}_{01}, \bar{P}_{10}, \bar{P}_{11}$ are coplanar!



Given 4 3D points $\bar{P}_{00}, \bar{P}_{01}, \bar{P}_{10}, \bar{P}_{11}$ the pair (α,β) uniquely determines $\bar{f}(\alpha,\beta)$:

$$\bar{l}_0(\alpha) = (1-\alpha)\bar{P}_{00} + \alpha\bar{P}_{01}$$
$$\bar{l}_1(\alpha) = (1-\alpha)\bar{P}_{10} + \alpha\bar{P}_{11}$$
$$\bar{f}(\alpha,\beta) = (1-\beta)\bar{l}_0(\alpha) + \beta\bar{l}_1(\alpha)$$

Refresher on (2D) Conic Sections

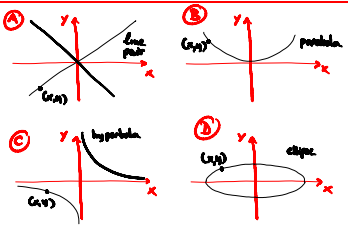


General implicit equation
$Ax^2 + By^2 + Cxy + Dx + Ey + F = 0$
shape controlled by params A,B,C,D,E,F (one of A,B,C must be $\neq 0$)

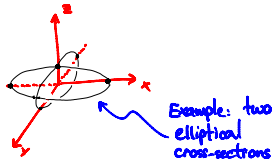
or $\begin{bmatrix} x & y & 1 \end{bmatrix} \begin{bmatrix} A & C/2 & D/2 \\ C/2 & B & E/2 \\ D/2 & E/2 & F \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = 0$
same eq but in homogeneous/matrix notation

Quadric Surfaces: Basic Construction

Definition:
Surfaces whose planar cross-sections are conics

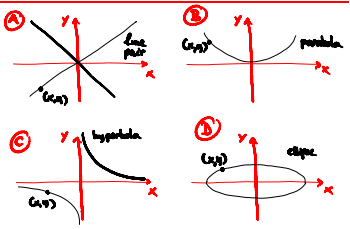


Intuition:
To visualize all possible shapes, consider their cross-sections with planes parallel to XZ- and YZ- planes

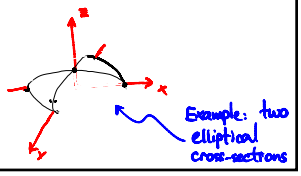


Quadric Surfaces: Basic Construction

Case D+D:
Ellipsoid

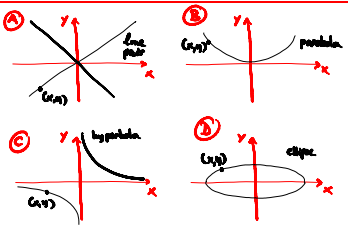


Intuition:
To visualize all possible shapes, consider their cross-sections with planes parallel to XZ- and YZ- planes

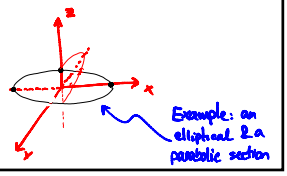


Quadric Surfaces: Basic Construction

Case B+D
Elliptical paraboloid



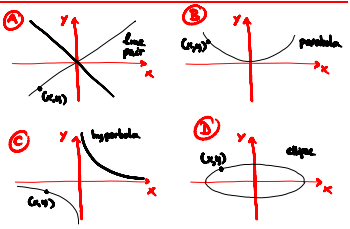
Intuition:
To visualize all possible shapes, consider their cross-sections with planes parallel to XZ- and YZ- planes



Quadric Surfaces: Basic Construction

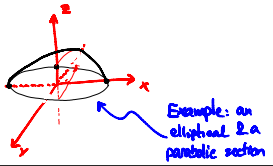
Case (B)+(D)

Elliptical paraboloid



Intuition:

To visualize all possible shapes, consider their cross-sections with planes parallel to XZ- and YZ- planes



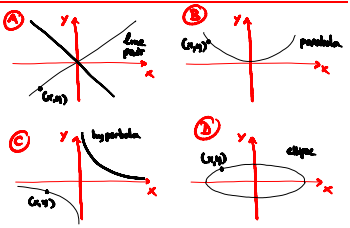
Quadric Surfaces: Basic Construction

(B)+(E)

Hyperbolic paraboloid

(C)+(D)

Elliptic hyperboloid



Quadric Surfaces: Implicit Equation

General implicit equation

$$[x\ y\ z\ 1]\begin{bmatrix} A & D & E & G \\ D & B & F & H \\ E & F & C & I \\ G & H & I & J \end{bmatrix}\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = 0$$

defined up to a scale factor

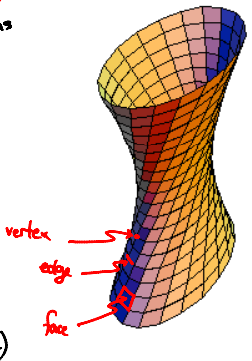
- 9 total degrees of freedom
- Planar cross-sections produce curves with an implicit eq of a conic section
- Only 5 parameters used (those circled) when cross-sections are "centered" on xy-, yz-, xz-planes

Polygonal Meshes

Definition (polygonal mesh)

A collection of polygons
- vertices
- edges
- faces

- Polyhedron: a closed, connected, polygonal mesh
- Face: Planar polygonal patch inside a mesh
- A mesh is simple when it has no holes (i.e. topologically equiv to a sphere)

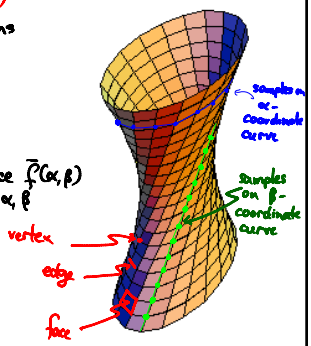


Polygonal Meshes

Definition (polygonal mesh)

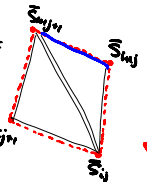
A collection of polygons
- vertices
- edges
- faces

Given a parametric surface $\bar{f}(\alpha,\beta)$ we can sample values of α, β to define a mesh that approximates it



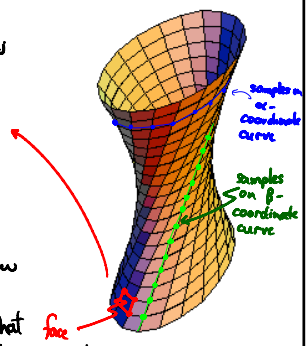
Generating Triangle Meshes

A quadruple of points may not define a planar face



Solution: break up quad into 2 triangles

* See online notes on how to do this properly
* See online notes about what data structure to use to store a mesh.

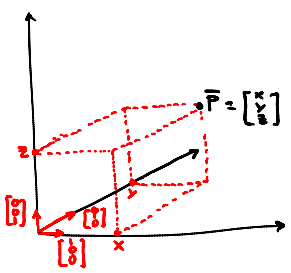


Topic 6:

3D Transformations

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

Representing Points by Euclidean 3D Coords



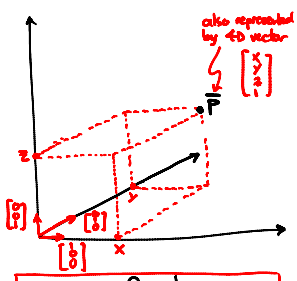
- "Standard" (Euclidean) representation of a point $\bar{P}$:

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

basis vectors

Euclidean coordinates

Euclidean Coords ⇒ Homogeneous Coords



also represented by 4D vector $\begin{bmatrix}x\\y\\z\\1\end{bmatrix}$

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

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

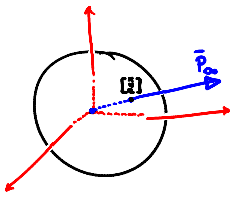
- Homogeneous (aka. Projective) representation of $\bar{P}$

Converting from homogeneous to Euclidean 3D Coords

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

3D coordinates	homogeneous 3D coordinates
$\begin{bmatrix}x\\y\\z\end{bmatrix}$	$\rightarrow \begin{bmatrix}x\\y\\z\\1\end{bmatrix}$

Points at ∞ in Homogeneous Coordinates

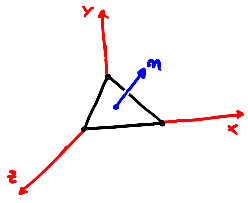


- A point at infinity does not represent a physical location on the plane
- It represents a direction

$$\bar{P}_\infty = \begin{bmatrix}x\\y\\z\\0\end{bmatrix}$$ 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, $\vec{n}$

or $\bar{\Pi}^T \cdot \bar{P} = 0$

vector holding plane parameters

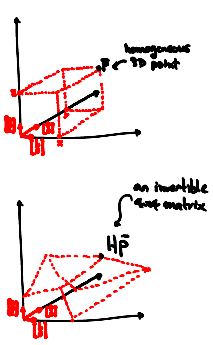
vector holding homogeneous coordinates of a point

Topic 6:

3D Transformations

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

General Linear 3D Transformations



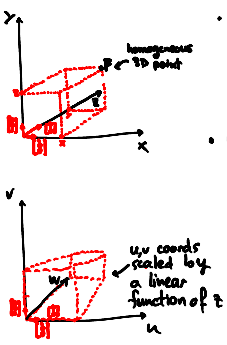
- The matrix H represents a very general set of transformations

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

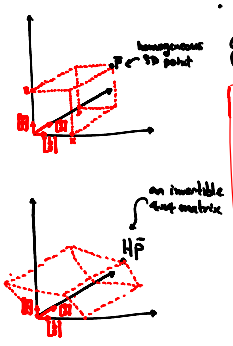
$(x, y, z) \rightarrow (\alpha(z)x, \alpha(z)y, \alpha(z)z)$ (non-linear in Euclidean coords)

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

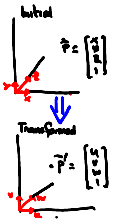


- The matrix H represents a very general set of transformations

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

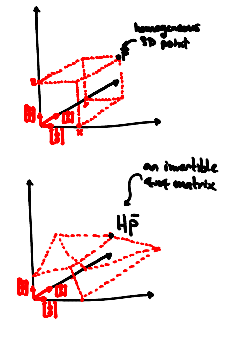
arbitrary 3x3 matrix, 3x1 vector

$$H = \begin{bmatrix} A & \vdots & \bar{t} \\ \cdots & \cdots & \cdots \\ 0\ 0\ 0 & \vdots & 1 \end{bmatrix}$$

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

$$\begin{bmatrix} A & \vdots & \bar{t} \\ \cdots & \cdots & \cdots \\ 0\ 0\ 0 & \vdots & 1 \end{bmatrix} \begin{bmatrix} \bar{x} \\ \cdots \\ 1 \end{bmatrix} = \begin{bmatrix} A\bar{x} + \bar{t} \\ \cdots \\ 1 \end{bmatrix}$$

From Affine to Rigid Transformations



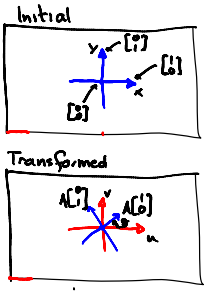
Affine: $\begin{bmatrix} A & \vdots & \bar{t} \\ \cdots & \cdots & \cdots \\ 0\ 0\ 0 & \vdots & 1 \end{bmatrix}$

General linear (preserve lines)
Affine (preserve parallelism)
- Arbitrary shearing
- General scaling

Conformal (preserve angles)
- Uniform scaling
- Reflection

- Translation
- Rotation

Rigid Transformations: Rotations in 3D



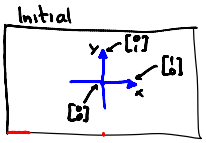
$$\begin{bmatrix} A_{3\times3} & \vdots & 0 \\ \cdots & \cdots & \cdots \\ 0\ 0\ 0 & \vdots & 1 \end{bmatrix} \quad A = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

$$\begin{bmatrix} A_{3\times3} & \vdots & 0 \\ \cdots & \cdots & \cdots \\ 0\ 0\ 0 & \vdots & 1 \end{bmatrix} \quad A_z = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

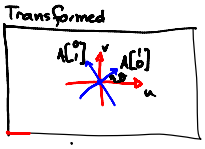
does not affect z-coords

Elementary Rotations in 3D

Initial



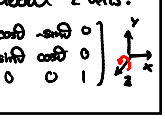
Transformed



$$\begin{bmatrix} A_x & | & 0 \\ \cdots & & \cdots \\ 0 & | & 1 \end{bmatrix} A = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

Rotation by θ about x axis

$$H_x^\theta = \begin{bmatrix} A_x & | & 0 \\ \cdots & & \cdots \\ 0 & | & 1 \end{bmatrix} \quad A_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix}$$

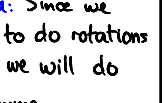


Rotation by θ about y axis:

$$H_y^\theta = \begin{bmatrix} A_y & | & 0 \\ \cdots & & \cdots \\ 0 & | & 1 \end{bmatrix} \quad A_y = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}$$

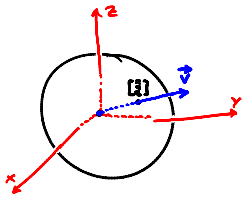
Rotation by θ about z axis:

$$H_z^\theta = \begin{bmatrix} A_z & | & 0 \\ \cdots & & \cdots \\ 0 & | & 1 \end{bmatrix} \quad A_z = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



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

$$H_x^\theta = \begin{bmatrix} A_x & | & 0 \\ \cdots & & \cdots \\ 0 & | & 1 \end{bmatrix} \quad A_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix}$$

Rotation by θ about y axis:

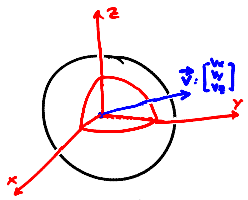
$$H_y^\theta = \begin{bmatrix} A_y & | & 0 \\ \cdots & & \cdots \\ 0 & | & 1 \end{bmatrix} \quad A_y = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}$$

Rotation by θ about z axis:

$$H_z^\theta = \begin{bmatrix} A_z & | & 0 \\ \cdots & & \cdots \\ 0 & | & 1 \end{bmatrix} \quad A_z = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Rotation About Arbitrary Vector: Construction

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



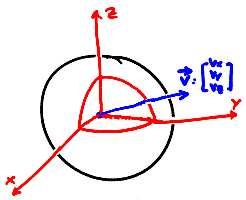
Ans: Express it as a composition of the three elementary matrices A_x, A_y, A_z

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

* Align $\vec{v}$ with the 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 ψ 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

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

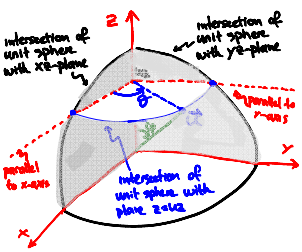
Step 2: Compute the spherical coordinates of $\vec{u}$ i.e. coordinates of $\vec{u}$ in the parameterization of a sphere

Aside: Spherical Coordinates of a Unit Vector

The cross-section of plane $z = u_z$

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

radius of circular cross-section



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

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

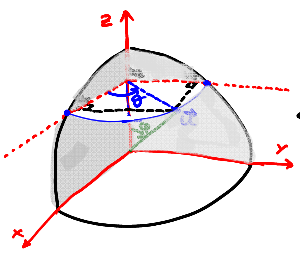
Step 2: Compute the spherical coordinates of $\vec{u}$ i.e. coordinates of $\vec{u}$ in the parameterization of a sphere

Aside: Spherical Coordinates of a Unit Vector

The cross-section of plane $z = u_z$

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

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



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

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

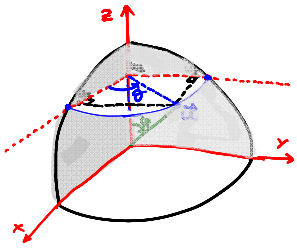
Step 2: Compute the spherical coordinates of $\vec{u}$ i.e. coordinates of $\vec{u}$ in the parameterization of a sphere

Aside: Spherical Coordinates of a Unit Vector

The cross-section of plane $z=u_z$

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

$u_z = \cos\varphi \rightarrow \varphi = \arccos(u_z)$



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

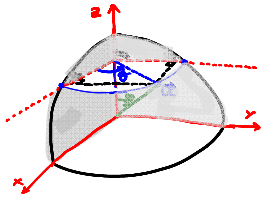
$$\vec{u} = \frac{\vec{v}}{\sqrt{v_x^2+v_y^2+v_z^2}}$$

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

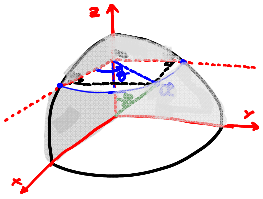
$$\vec{u} = \frac{\vec{v}}{\sqrt{v_x^2+v_y^2+v_z^2}}$$

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:

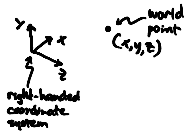
$$\bar{p}' = H_z^{\vartheta} H_y^{\varphi} H_z^{\psi} H_y^{-\varphi} H_z^{-\vartheta} \bar{p}$$

Topic 7:

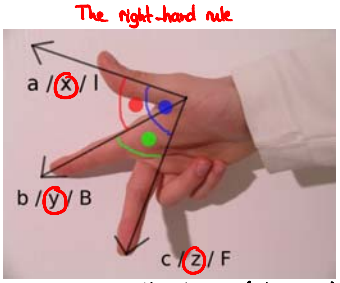
3D Viewing

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

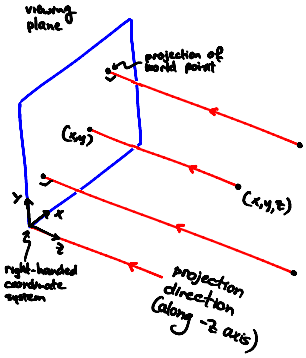
The Camera-Centered Coordinate System



The right-hand rule



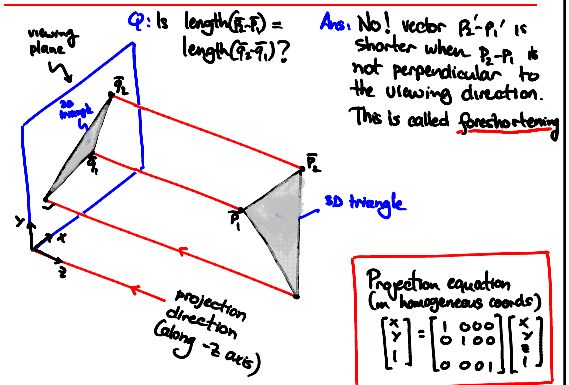
Orthographic Projection



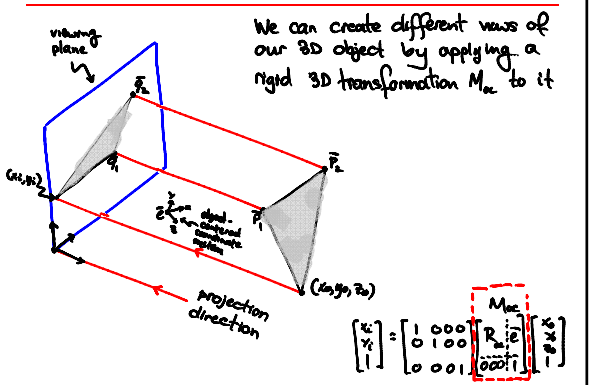
Projection equation (in homogeneous coords)

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

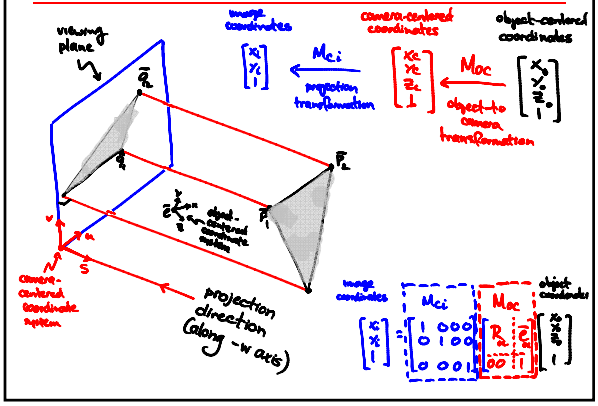
Orthographic Projection (cont.)



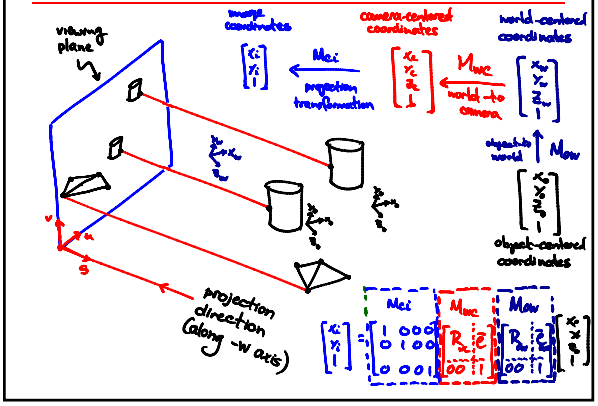
The Object-to-Camera Transformation



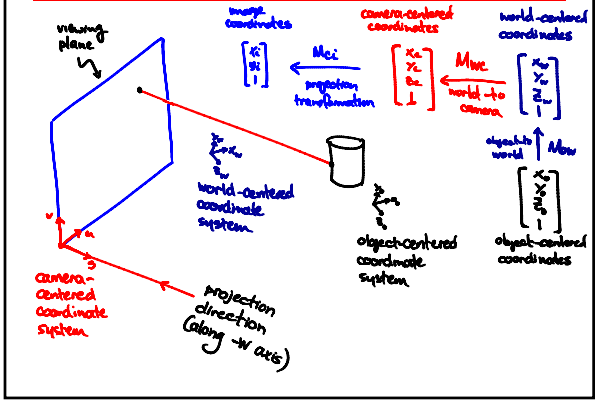
Main Transformations Used in 3D Viewing



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 3x4 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 3x4 matrix that maps camera-centered 3D coords to 2D image coords

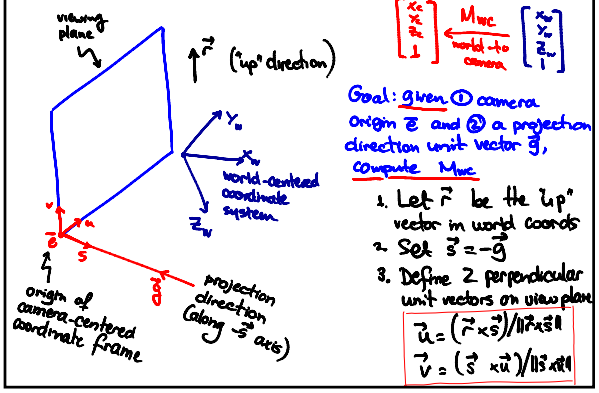
Q: How do we define M_{wc}?

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



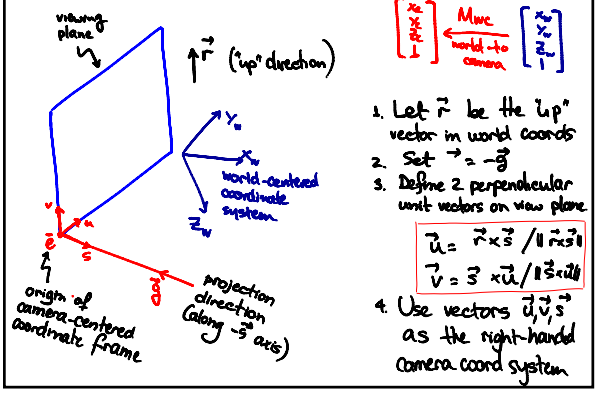
Goal: given ① camera origin $\vec{e}$ and ② a projection direction unit vector $\vec{g}$, compute M_{wc}

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

$$\vec{u} = (\vec{r} \times \vec{s}) / \|\vec{r} \times \vec{s}\|$$

$$\vec{v} = (\vec{s} \times \vec{u}) / \|\vec{s} \times \vec{u}\|$$

Computing the World-to-Camera Transform



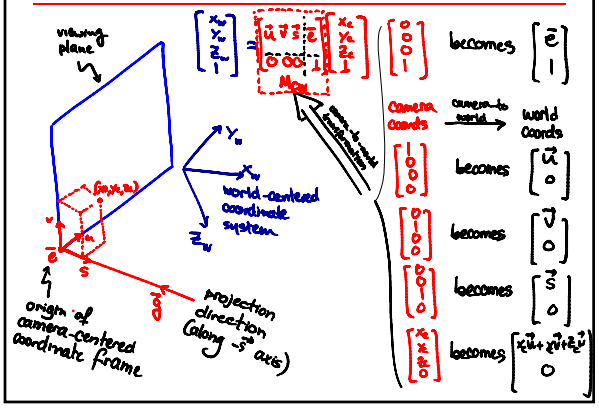
1. Let $\vec{r}$ be the "up" vector in world coords
2. Set $\vec{s} = -\vec{g}$
3. Define 2 perpendicular unit vectors on view plane

$$\vec{u} = \vec{r} \times \vec{s} \,/\, \|\vec{r} \times \vec{s}\|$$

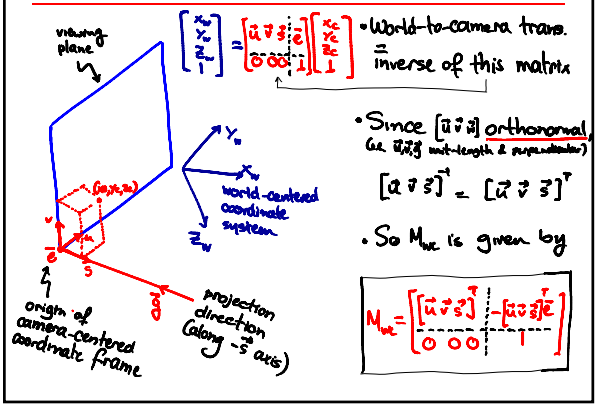
$$\vec{v} = \vec{s} \times \vec{u} \,/\, \|\vec{s} \times \vec{u}\|$$

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

First Compute Camera-to-World Transform...



... then Compute its Inverse



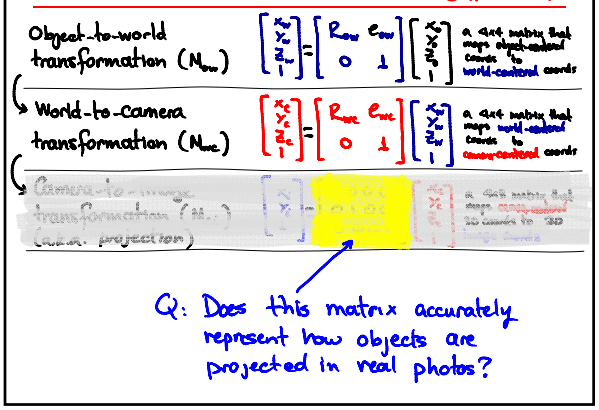
- World-to-camera trans. = inverse of this matrix
- Since $[\vec{u}\ \vec{v}\ \vec{s}]$ orthonormal (i.e. $\vec{u},\vec{v},\vec{s}$ unit-length & perpendicular)

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

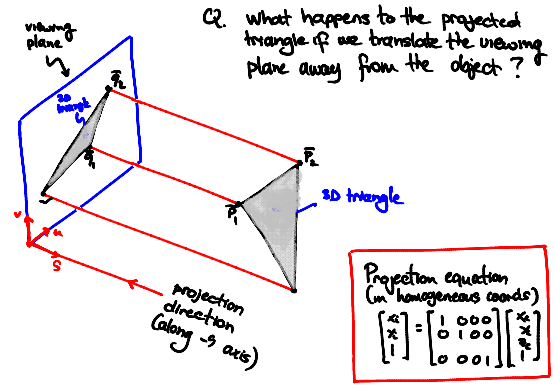
- So M_{wc} is given by

$$M_{wc} = \begin{bmatrix} [\vec{u}\ \vec{v}\ \vec{s}]^{T} & -[\vec{u}\ \vec{v}\ \vec{s}]^{T}\vec{e} \\ 0\ 0\ 0 & 1 \end{bmatrix}$$

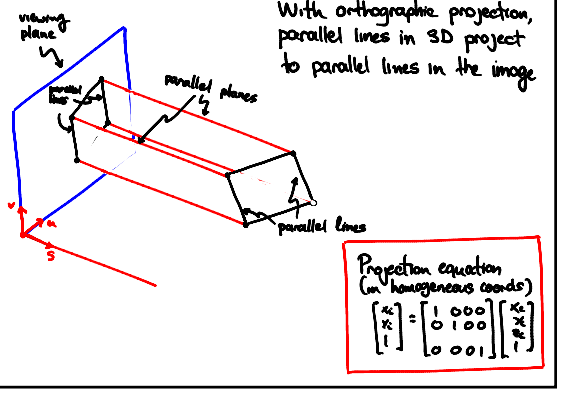
Transformation Chain for 3D Viewing (partial)

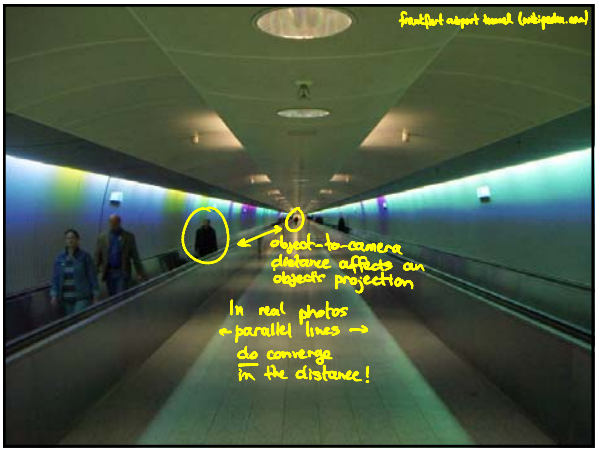


How Accurate is Orthographic Projection?

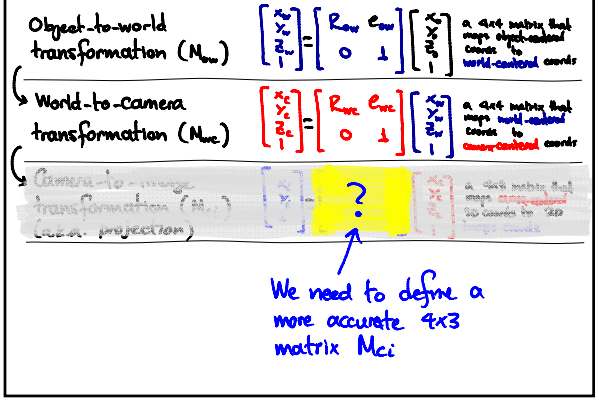


How Accurate is Orthographic Projection?





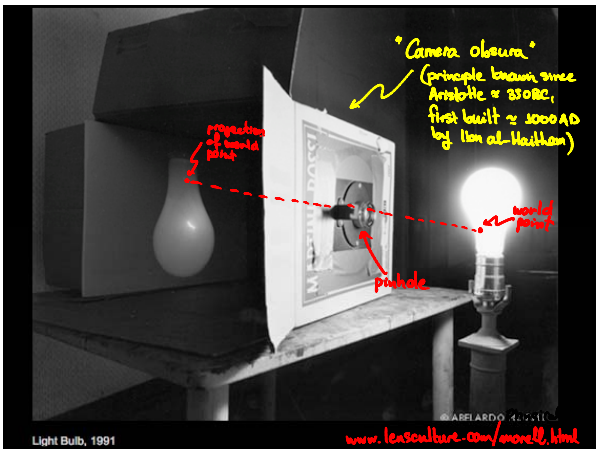
Transformation Chain for 3D Viewing (partial)



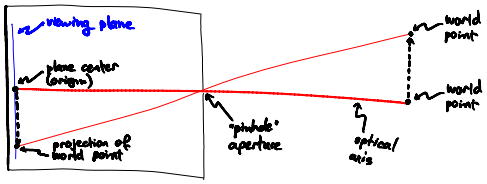
Topic 7:

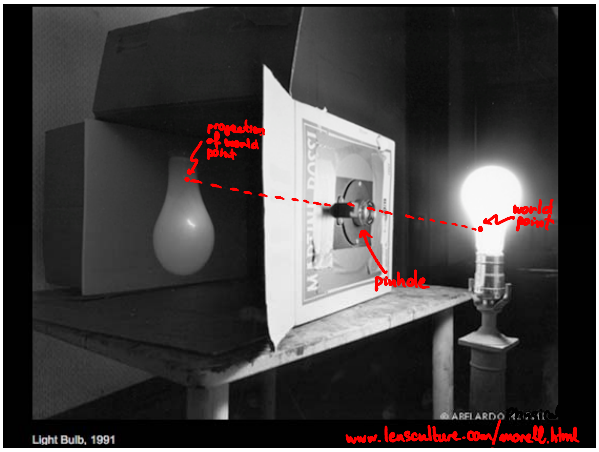
3D Viewing

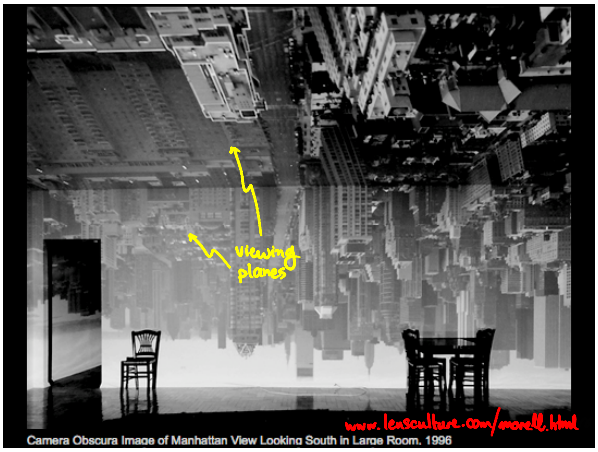
- Orthographic projection
- The world-to-camera transformation
- Perspective projection
- The transformation chain for 3D viewing

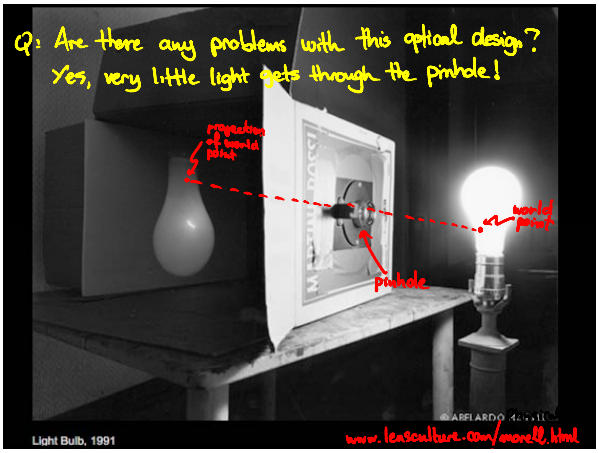


The Pinhole Camera

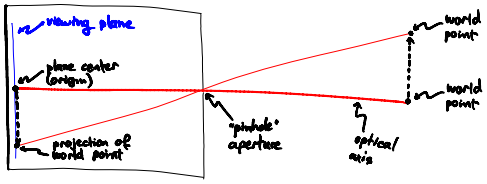




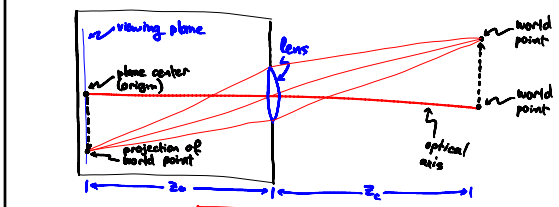




The Pinhole Camera



Simple Lens-Based Camera & Thin-Lens Law

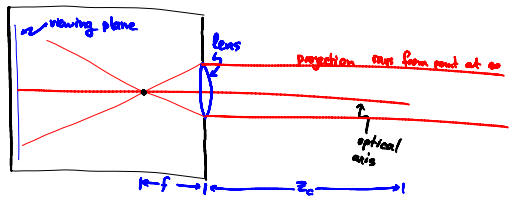


When $z_c \to \infty$, $z_o \to f$
So f is the distance at which rays converge for a world point at ∞

Thin-lens law:

$$\frac{1}{z_o} + \frac{1}{z_c} = \frac{1}{f}$$

Simple Lens-Based Camera & Thin-Lens Law



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

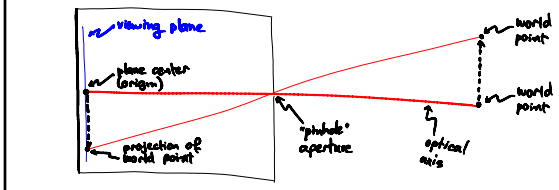
Thin-lens law:

$$\frac{1}{z_o} + \frac{1}{z_c} = \frac{1}{f}$$

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

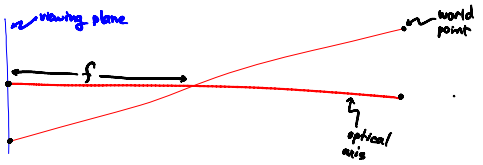
The Pinhole Camera

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



The Pinhole Camera: Basic Geometry

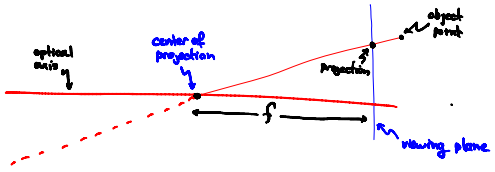
We will only consider the idealized pinhole model here (a.k.a. perspective projection)



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

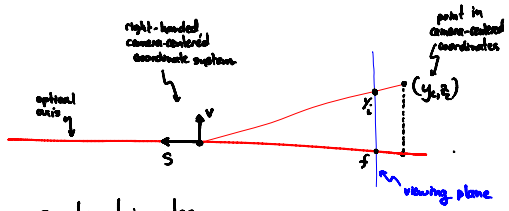
The Pinhole Camera: Basic Geometry in 2D

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

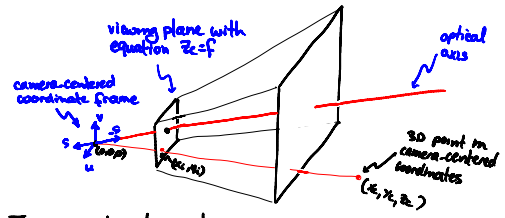


From similar triangles,

$$\frac{y_c}{z_c} = \frac{y_i}{f} \Rightarrow y_i = \frac{f}{z_c} y_c$$

The perspective projection equation

The Perspective Projection Equations in 3D



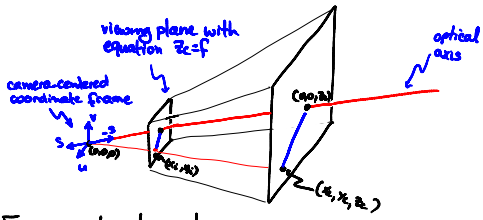
From similar triangles,

$$\frac{y_c}{z_c} = \frac{y_i}{f} \Rightarrow y_i = \frac{f}{z_c} y_c$$

analogously, for x_i: $\frac{x_c}{z_c} = \frac{x_i}{f} \Rightarrow x_i = \frac{f}{z_c} x_c$

The perspective projection equations

The Perspective Projection Equations in 3D

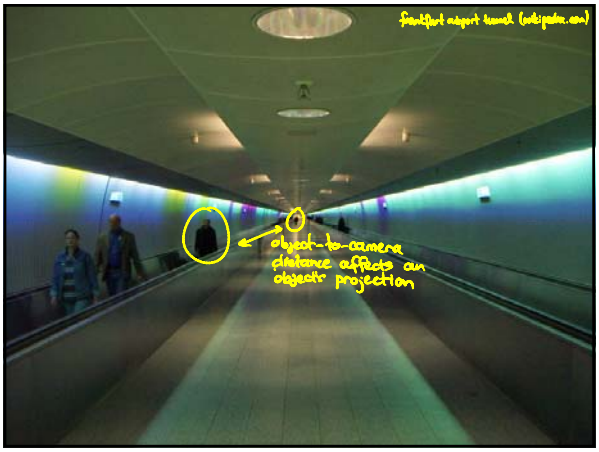


From similar triangles,

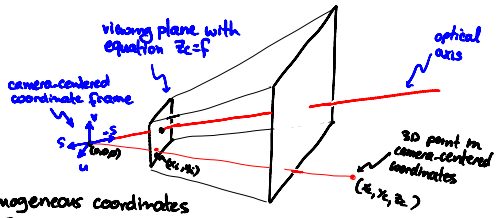
$$\frac{y_c}{z_c} = \frac{y_i}{f} \Rightarrow y_i = \frac{f}{z_c} y_c$$

analogously, for x_i: $\frac{x_c}{z_c} = \frac{x_i}{f} \Rightarrow x_i = \frac{f}{z_c} x_c$

⇒ As objects move farther away (i.e. $|z_c|$ increases) their projection gets smaller and smaller



Perspective Projection & Homogeneous Coords



In homogeneous coordinates

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

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

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}) (i.e. projection)

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

a 4x4 matrix that maps camera-centered 3D coords to 2D

models perspective projection

The Canonical View Volume Transform

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

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

In homogeneous coordinates:

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

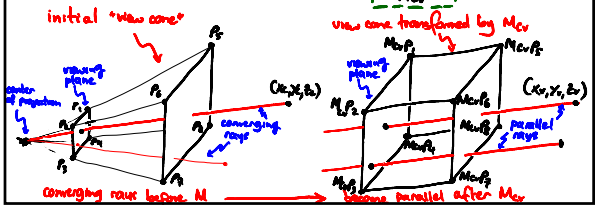
$$x_i = \frac{f}{z_c} x_c \qquad y_i = \frac{f}{z_c} y_c$$

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

$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} \cong \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_v \\ y_v \\ z_v \\ 1 \end{bmatrix} \qquad \begin{bmatrix} x_v \\ y_v \\ z_v \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & \frac{1}{z_p} & 0 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix}$$

M_{cv}



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})

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

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

↳ Canonical new-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