Computer Vision I - Appearance-based Matching and Projective Geometry

Carsten Rother

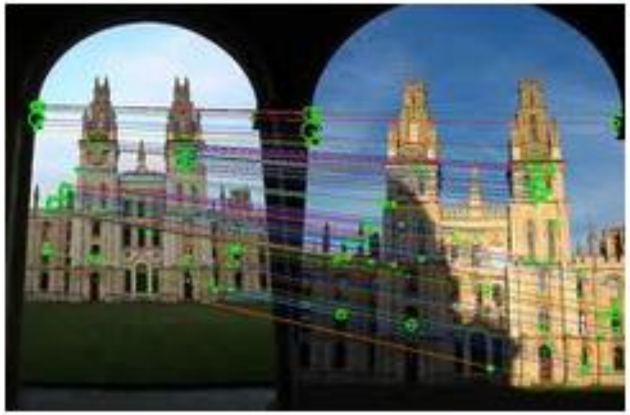
05/11/2015

Roadmap for next four lectures

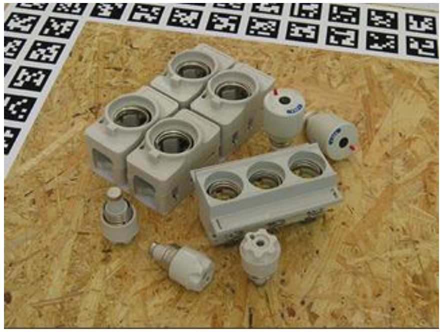
- Appearance-based Matching (sec. 4.1)
- Projective Geometry - Basics (sec. 2.1.1-2.1.4)
- Geometry of a Single Camera (sec 2.1.5, 2.1.6)
  - Camera versus Human Perception
  - The Pinhole Camera
  - Lens effects
- Geometry of two Views (sec. 7.2)
  - The Homography (e.g. rotating camera)
  - Camera Calibration (3D to 2D Mapping)
  - The Fundamental and Essential Matrix (two arbitrary images)
- Robust Geometry estimation for two cameras (sec. 6.1.4)
- Multi-View 3D reconstruction (sec. 7.3-7.4)
  - General scenario
  - From Projective to Metric Space
  - Special Cases

Objective

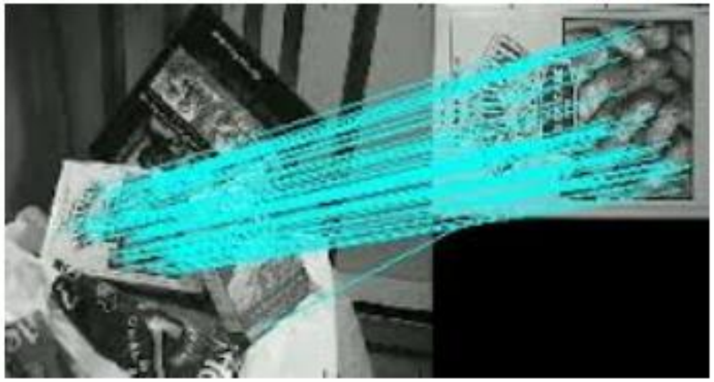
- Input: Two images which have some common scene geometry.
  Assume the common scene part is textured.
  Goal: Match interest points of the common scene geometry

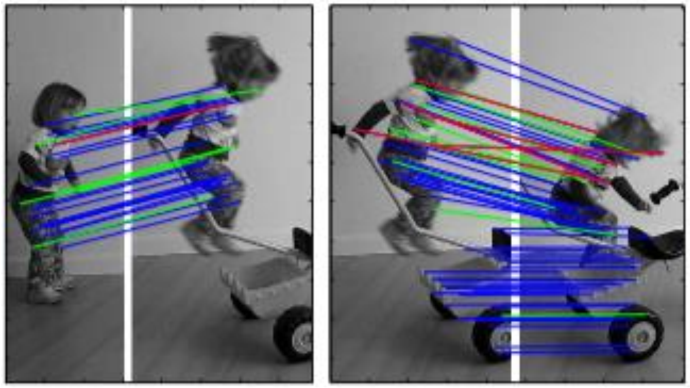


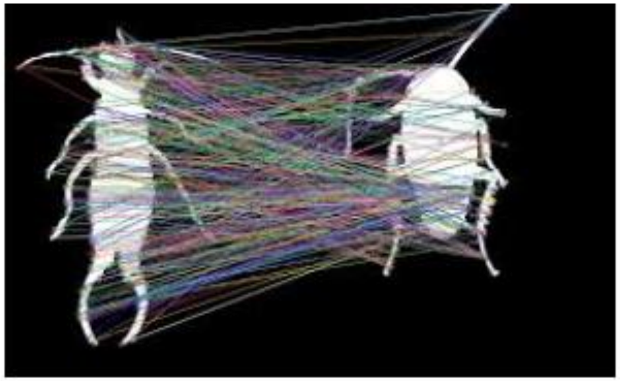
- Matching of objects which are texture-less is harder (later lecture)

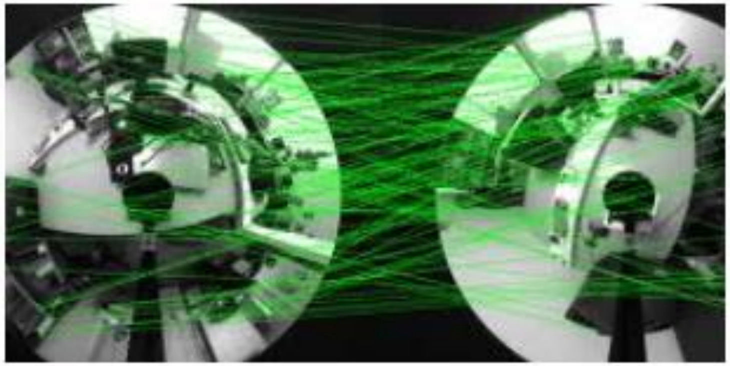


Examples: Appearance-based matching



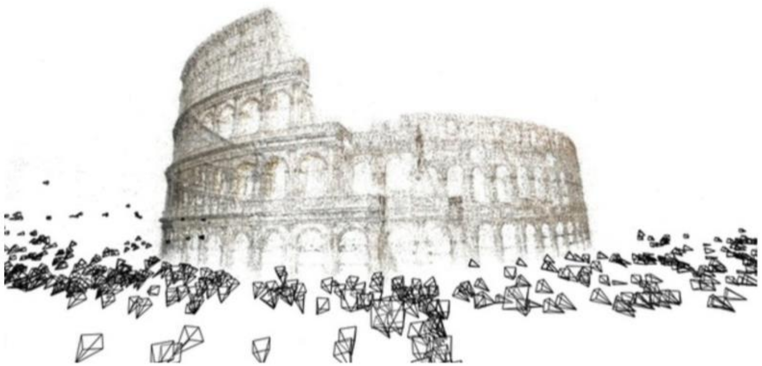






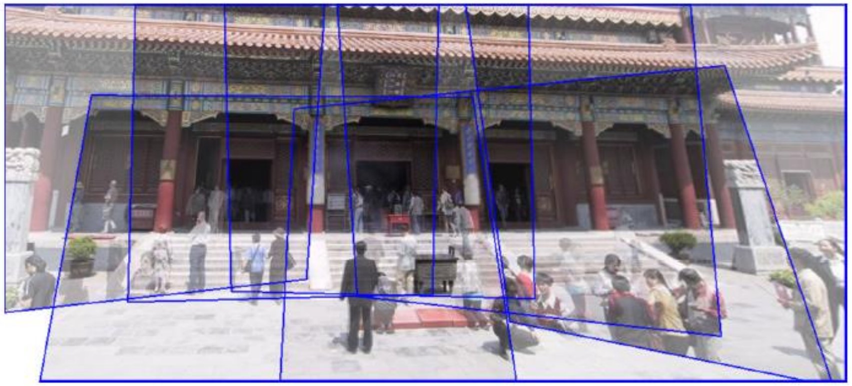
Applications

- 3D reconstruction:



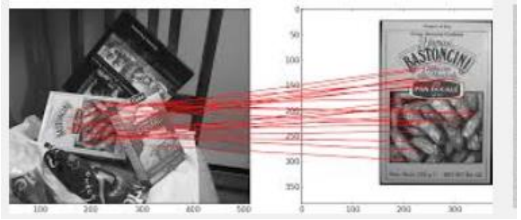
- Augmented Realty:

- Panoramic Stitching:



- Robotics

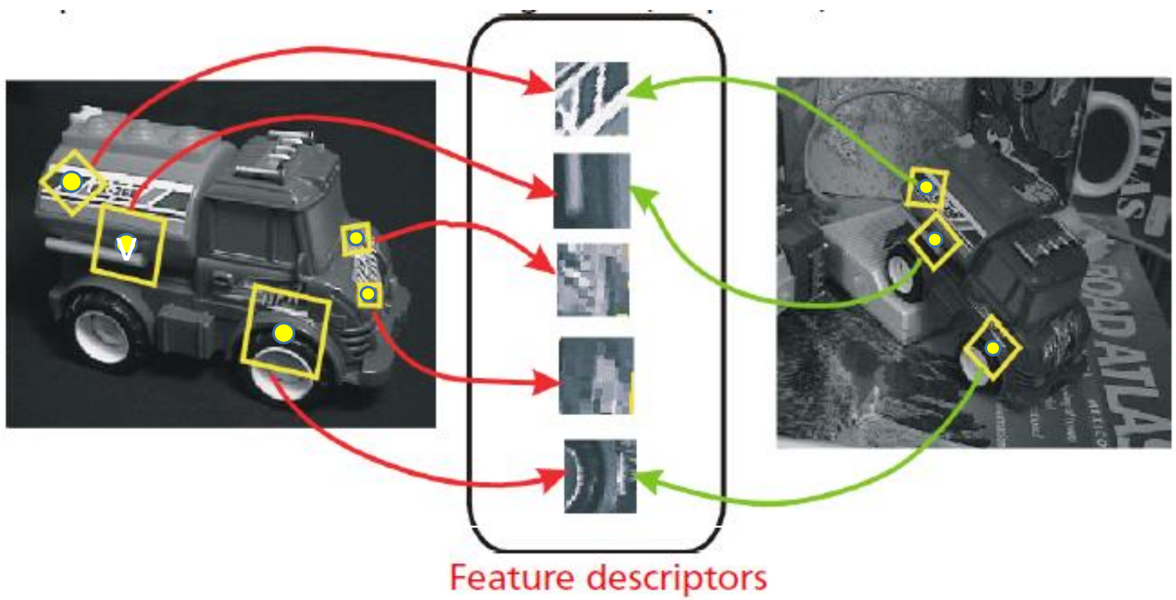
Camera re-localization



Grasping known objects

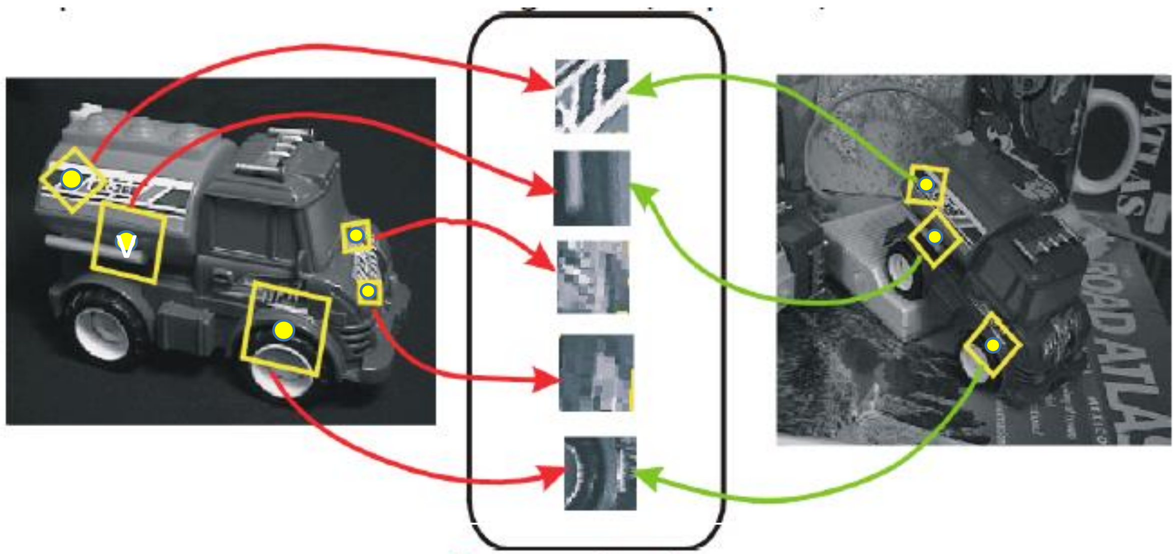
Matching Points between two Images

- Find interest points
- Find orientated patches around interest points to capture appearance
- Encode patch appearance in a descriptor
- Find matching patches according to appearance (similar descriptors)
- Verify matching patches according to geometry (later lecture)



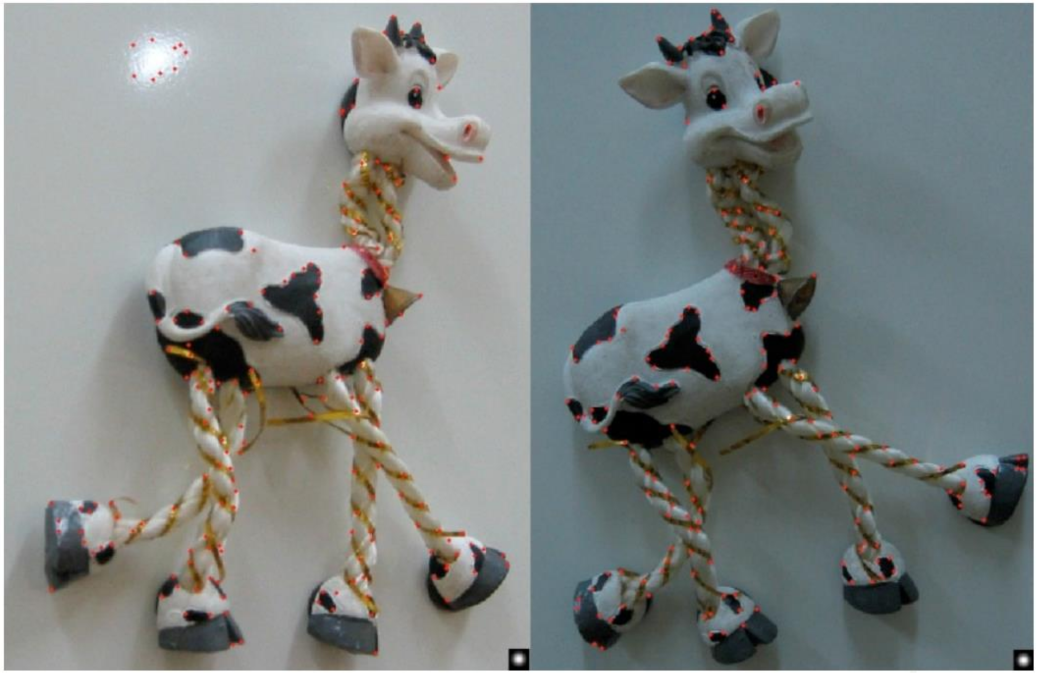
Matching Points between two Images

- Find interest points
- Find orientated patches around interest points to capture appearance
- Encode patch appearance in a descriptor
- Find matching patches according to appearance (similar descriptors)
- Verify matching patches according to geometry (later lecture)



Feature descriptors

Reminder: Harris Corner Detector



Compute:

1. $Q(x,y) = \left[\begin{array}{cc} \sum_W I_x(u,v)^2 & \sum_W I_x(u,v) I_y(u,v) \\ \sum_W I_x(u,v) I_y(u,v) & \sum_W I_y(u,v)^2 \end{array} \right] = \left[\begin{array}{cc} A & B \\ B & C \end{array} \right]$

   Based on so-called auto-correlation function: $c(x,y,\triangle x,\triangle y) \approx [\triangle x, \triangle y] Q(x,y) \begin{bmatrix} \triangle x \\ \triangle y \end{bmatrix}$

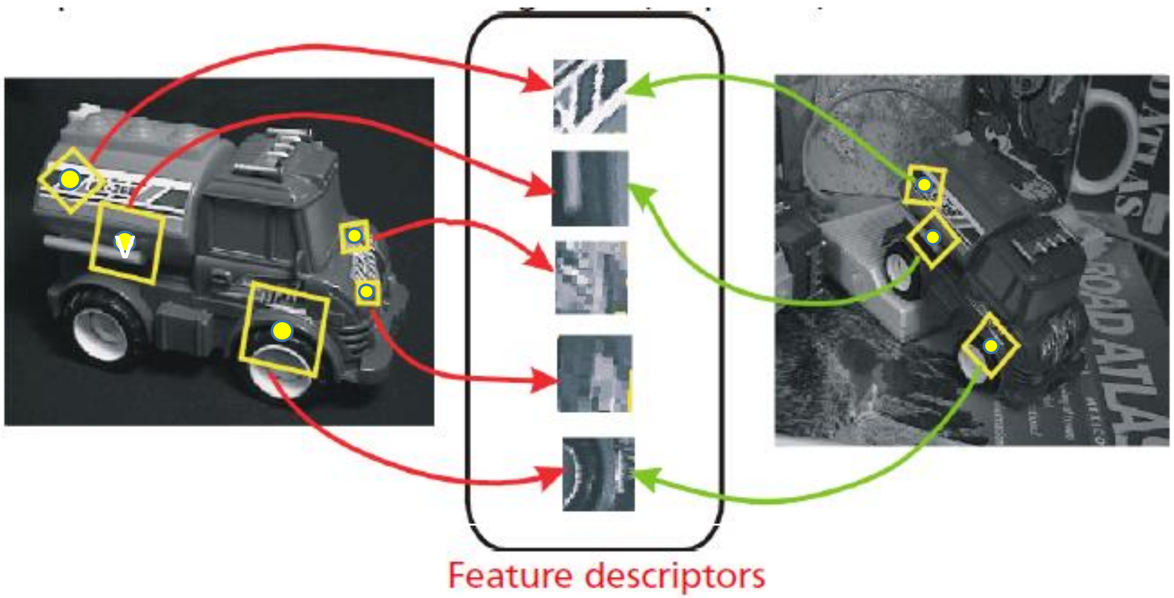
2. $\lambda_1 \lambda_2 = \det Q(x,y) = AC - B^2, \quad \lambda_1 + \lambda_2 = \mathrm{trace}\, Q(x,y) = A + C$

3. Harris measure: $H = \lambda_1 \lambda_2 - 0.04(\lambda_1 + \lambda_2)^2$

4. Take those points (after non-max suppression) with high *H* value

Matching Points between two Images

- Find interest points
- Find orientated patches around interest points to capture appearance
- Encode patch appearance in a descriptor
- Find matching patches according to appearance (similar descriptors)
- Verify matching patches according to geometry (later lecture)



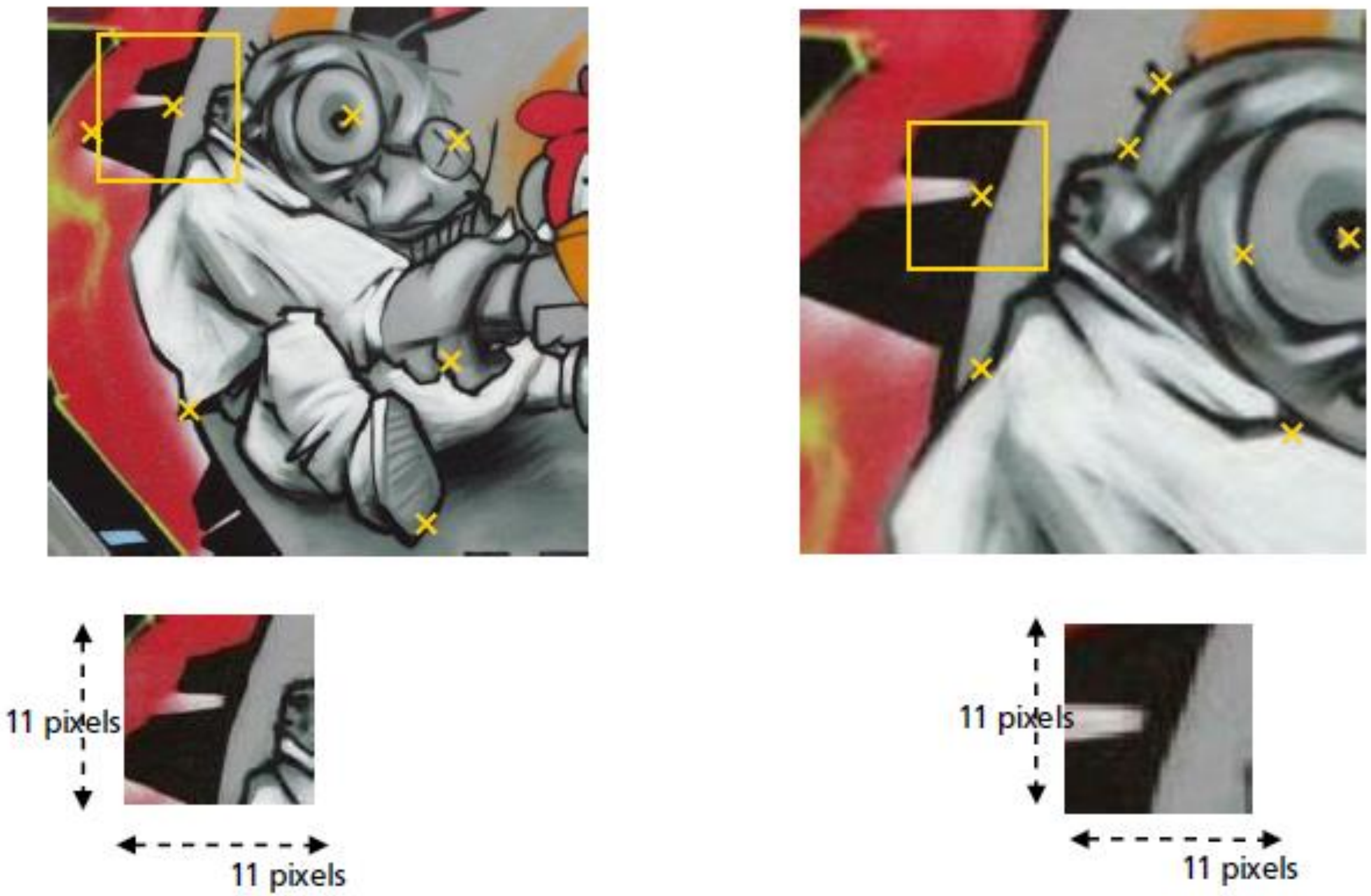
How to deal with orientation

Orientate with image gradient:

$$\nabla I = (I_x, I_y) = \left(\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}\right)$$

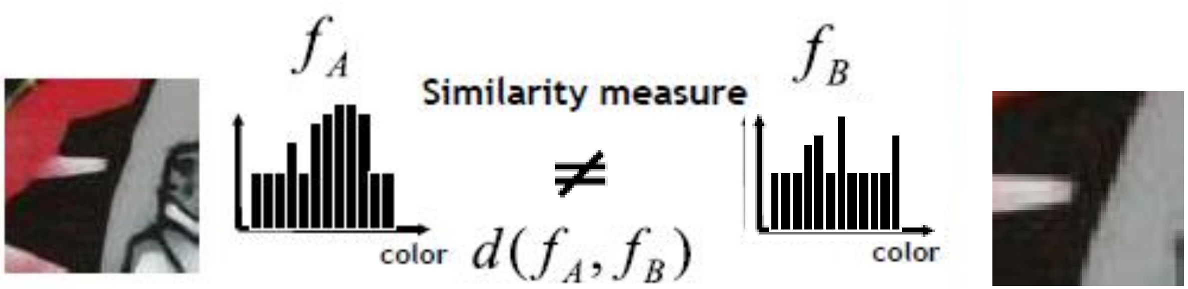
$$\theta = \operatorname{atan}(I_y, I_x)$$

Choose a patch around each point



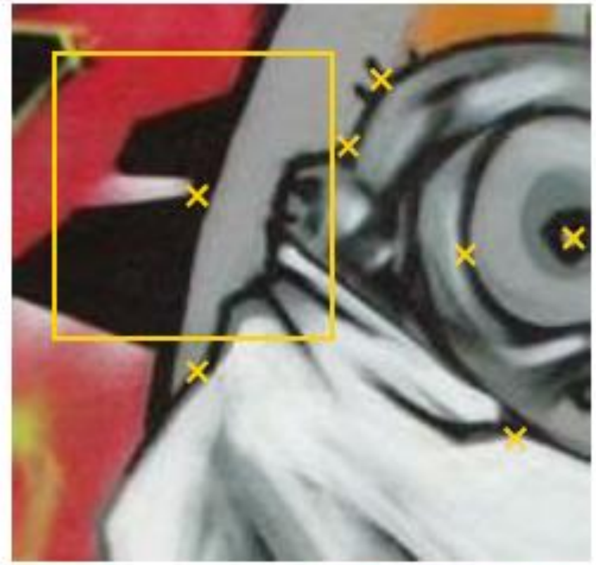
How to deal with scale?

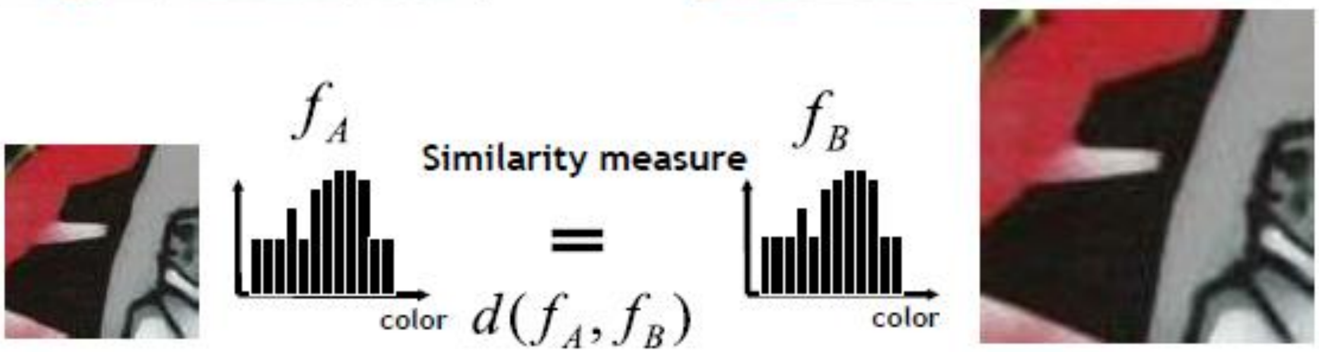
Choose a patch around each point



How to deal with scale?

Choose a patch around each point



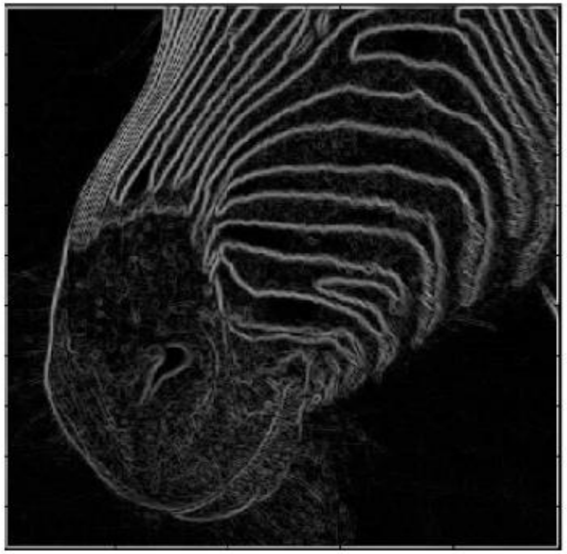


How to deal with scale?

Reminder: Edge detection via image gradient

Image gradient: $\nabla I = ((D_x * G) * I, (D_y * G) * I)$

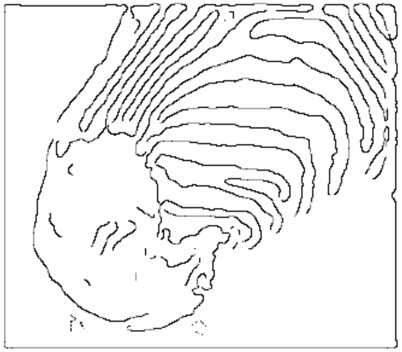
Image



Result of $||\nabla I||$
(using small sigma for Gaussian)

Result of $||\nabla I||$
(using large sigma for Gaussian)

Final result with canny edge detector



Alterative Edge Detector via LoG Operator

- The Laplacian:

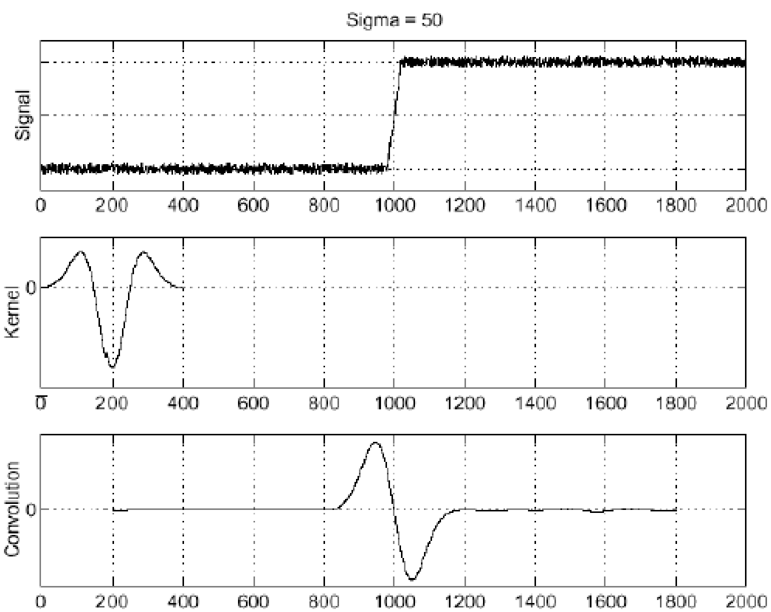
$$\nabla^2 I = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2}$$

- To find an edge we first smooth $\nabla^2 (G * I) = (\nabla^2 G) * I$
- $(\nabla^2 G)$ is called the LoG (Laplacian of Gaussian operator)
- 1D example:

f

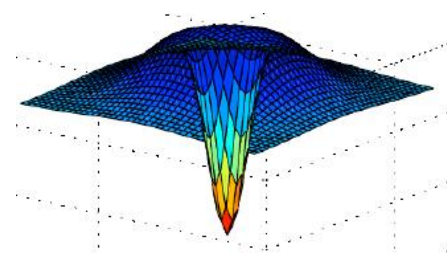
$\frac{d^2}{dx^2} g$

$\left(\frac{d^2}{dx^2} g \right) * f$



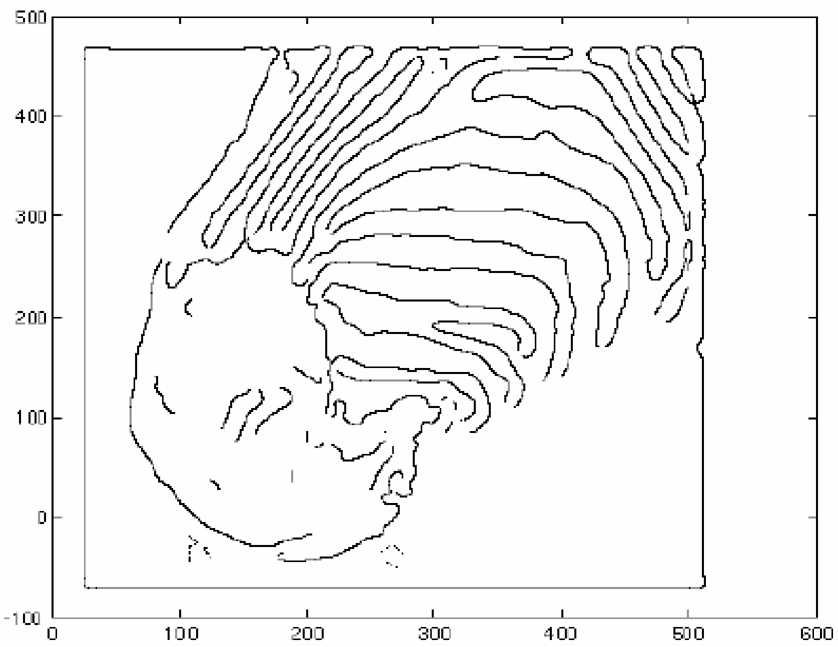
Find zero-crossing

- 2D example (Mexican hat):

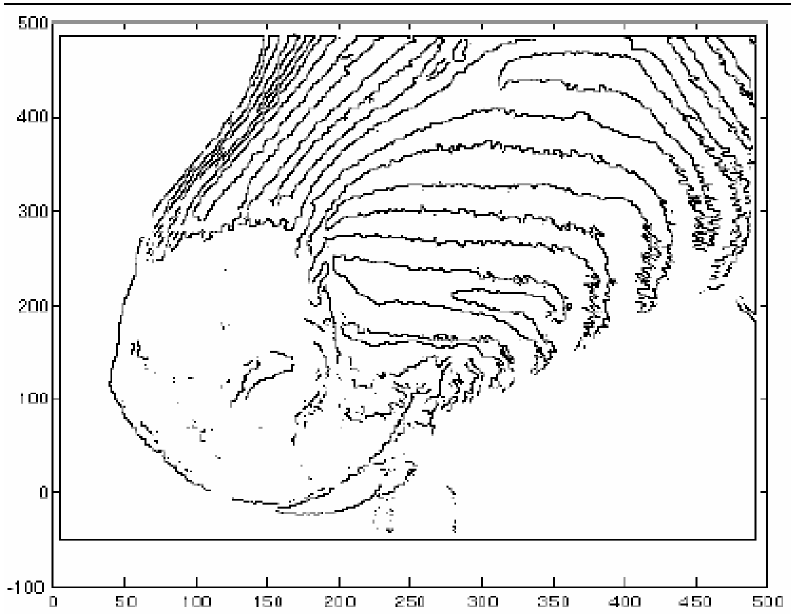


Alterative: Edge detection with LoG Filter

sigma = 4

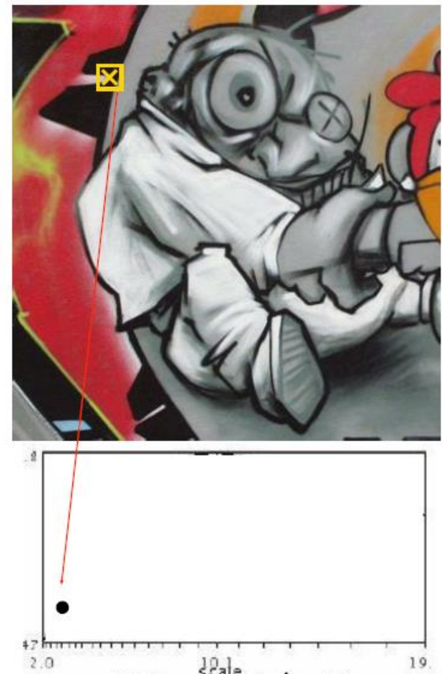


sigma = 2



Scale selection (illustration)

$f\left(I_{r_1 \ldots r_m}(x, \sigma)\right)$



$f\left(I_{r_1 \ldots r_m}(x', \sigma)\right)$

f *is Laplacian of Gaussian (LoG) operator.*
Measures an average edge-ness in all directions

$$\nabla^2(G(\sigma) * I) = \frac{\partial^2\,(G(\sigma) * I)}{\partial x^2} + \frac{\partial^2\,(G(\sigma) * I)}{\partial y^2}$$

(details on page 191)

Scale selection (illustration)

$f\left(I_{r_1 \ldots r_m}(x, \sigma)\right)$

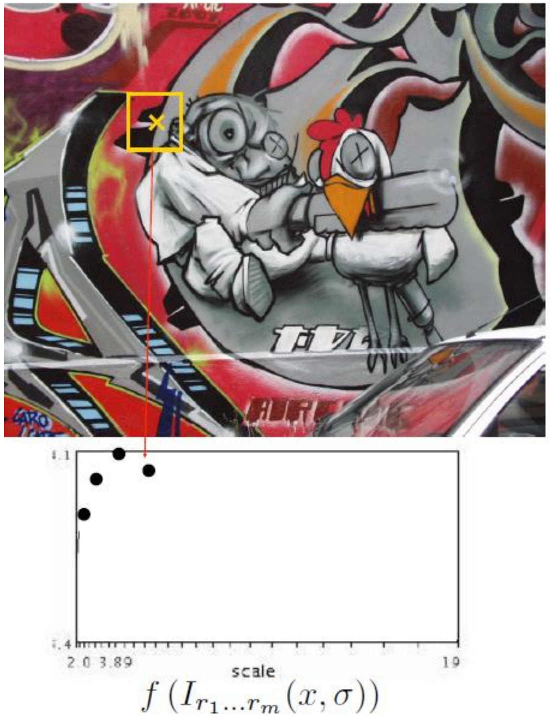
$f\left(I_{r_1 \ldots r_m}(x', \sigma)\right)$

Scale selection (illustration)

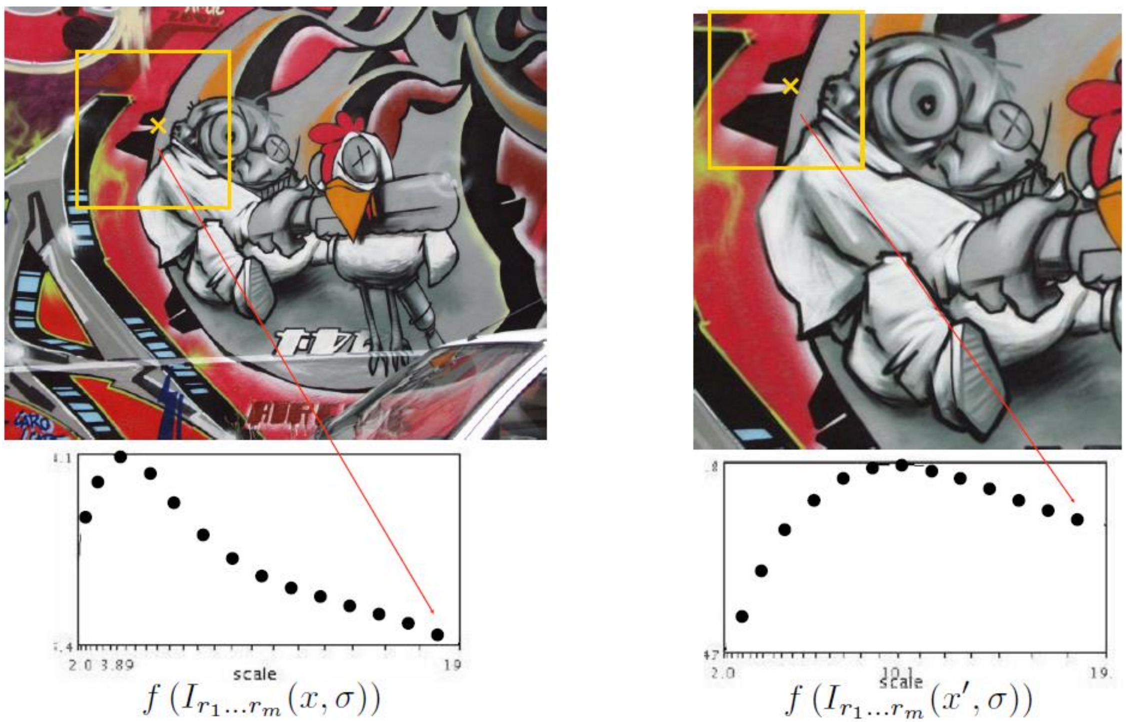
$$f\left(I_{r_1 \ldots r_m}(x,\sigma)\right)$$

$$f\left(I_{r_1 \ldots r_m}(x',\sigma)\right)$$

Scale selection (illustration)

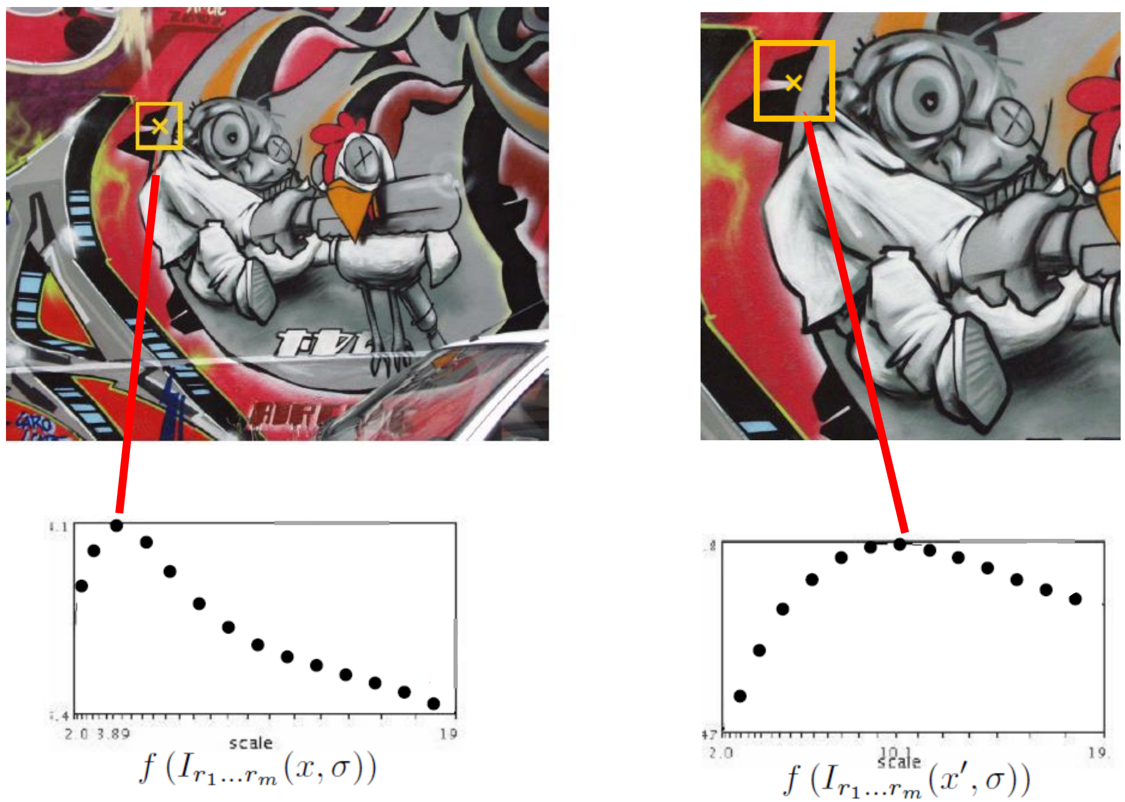


Scale selection (illustration)



We could match up these curves and find unique corresponding points

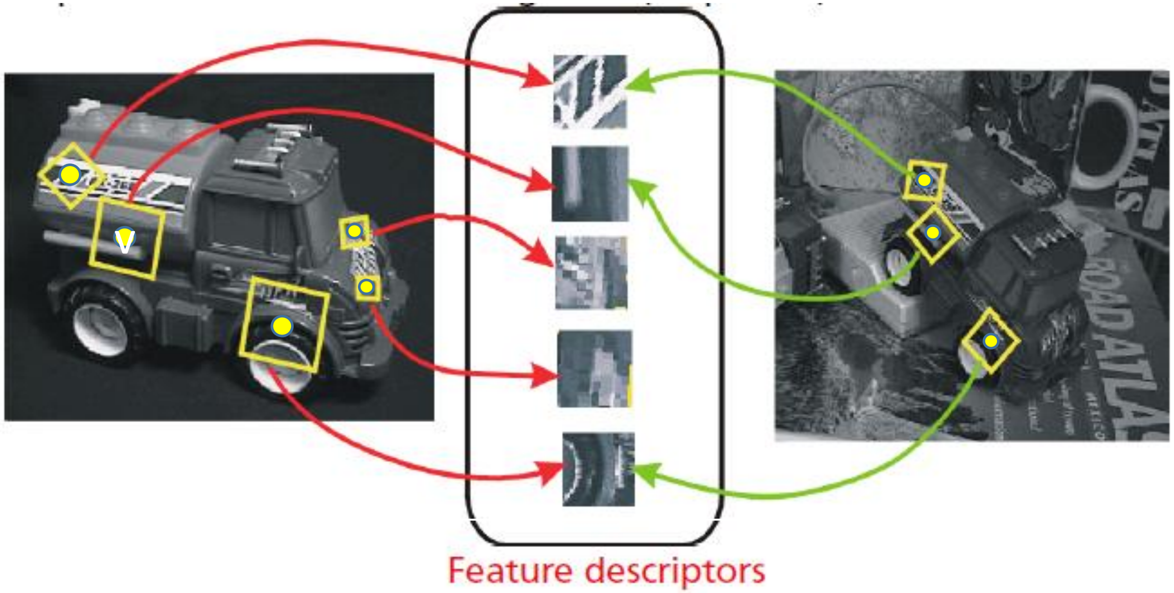
Scale selection (illustration)



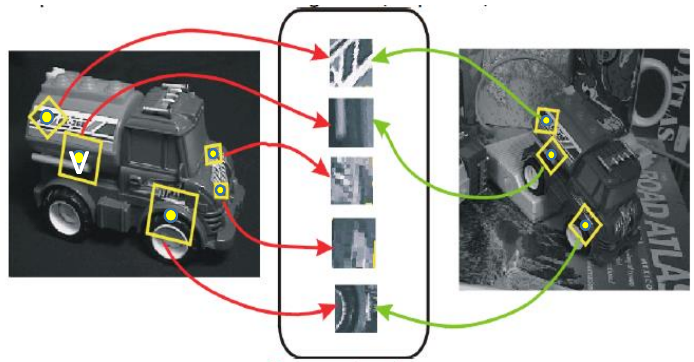
Simpler: Find maxima of the curve

Matching Points between two Images

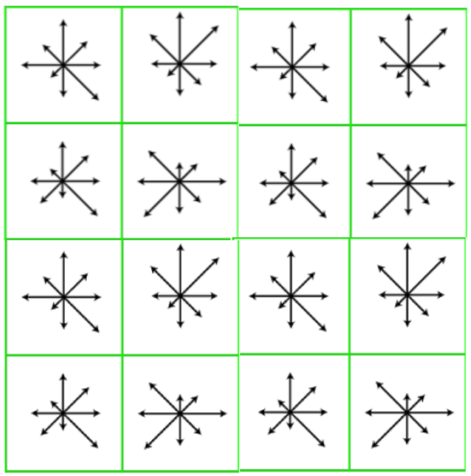
- Find interest points
- Find orientated patches around interest points to capture appearance
- Encode patch appearance in a descriptor
- Find matching patches according to appearance (similar descriptors)
- Verify matching patches according to geometry



SIFT features (Scale Invariant Feature Transform)

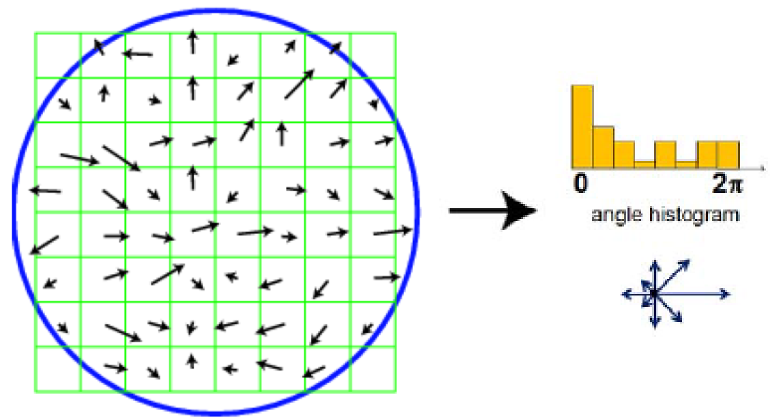


64 pixels



64 pixels

- 4*4=16 cells
- Each cell has an 8-bin histogram
- In total: 16*8 values, i.e. 128D vector



A cell has 16x16 pixels
(here 8x8 for illustration only)
(blue circle shows center weighting)

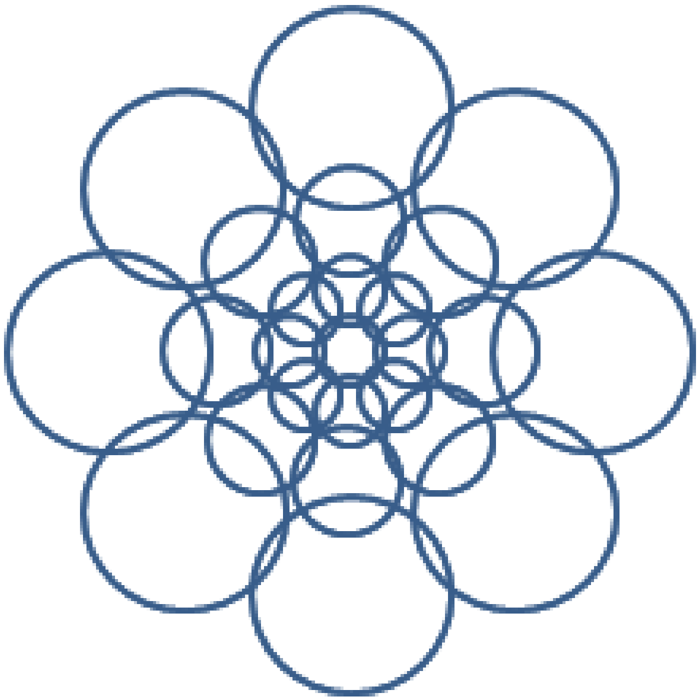
[Lowe 2004]

SIFT feature is very popular

- Fast to compute
- Can handle large changes in viewpoint well (up to 60^o out-of-plane rotation)
- Can handle photometric changes (even day versus night)

Many other feature descriptors

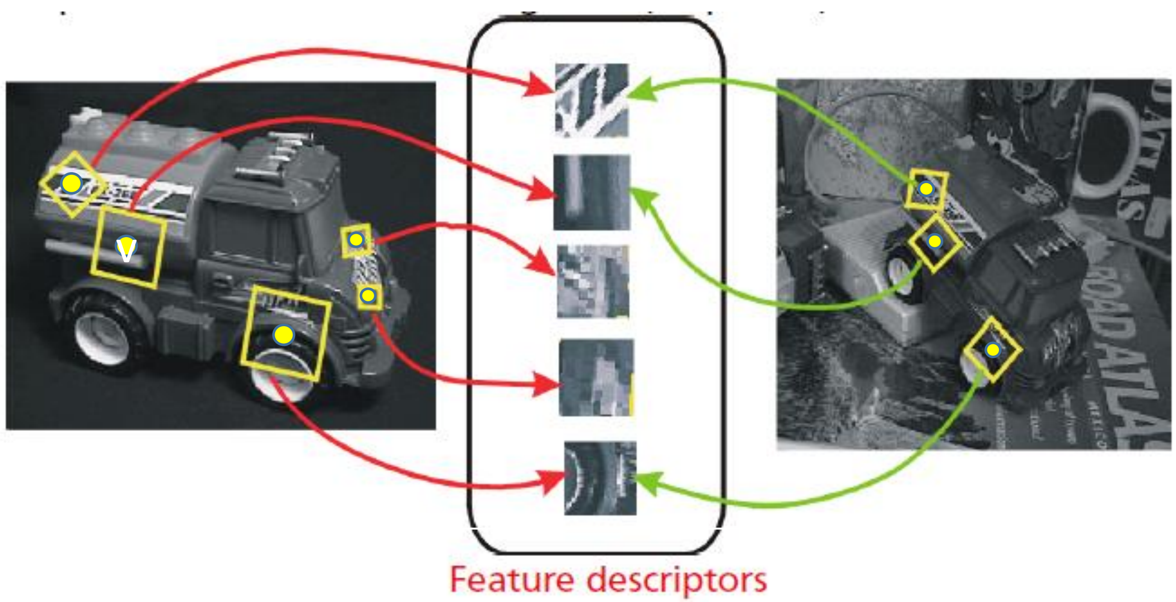
- MOPS [Brown, Szeliski and Winder 2005]
- SURF [Herbert Bay et al. 2006]
- DAISY [Tola, Lepetit, Fua 2010]
- Shape Context
- Deep Learning
- …



DAISY

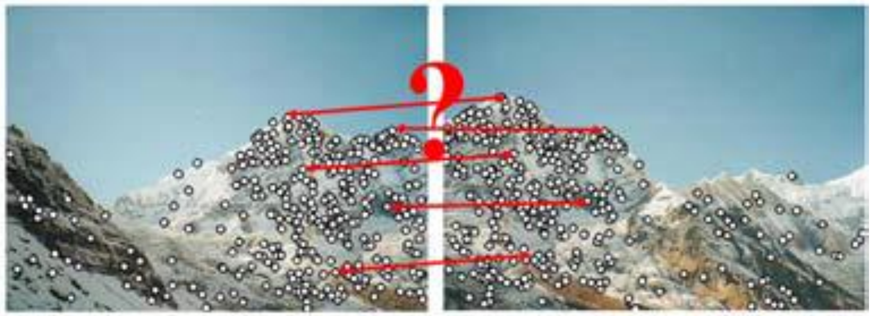
Matching Points between two Images

- Find interest points
- Find orientated patches around interest points to capture appearance
- Encode patch appearance in a descriptor
- Find matching patches according to appearance (similar descriptors)
- Verify matching patches according to geometry (later lecture)



Find matching patches fast

N patches
(e.g. N = 1000)



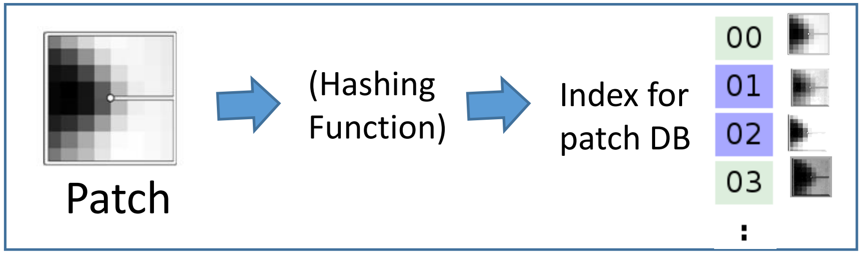
N patches
(e.g. N = 1000)

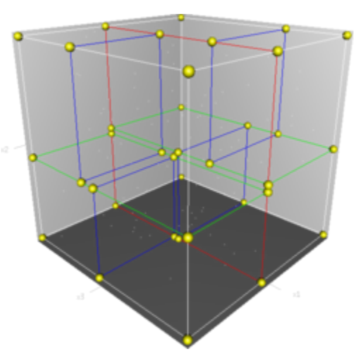
Goal:

1) Find for each patch in left image the closest in right image
2) Accept all those matches where descriptors are similar enough

Methods:

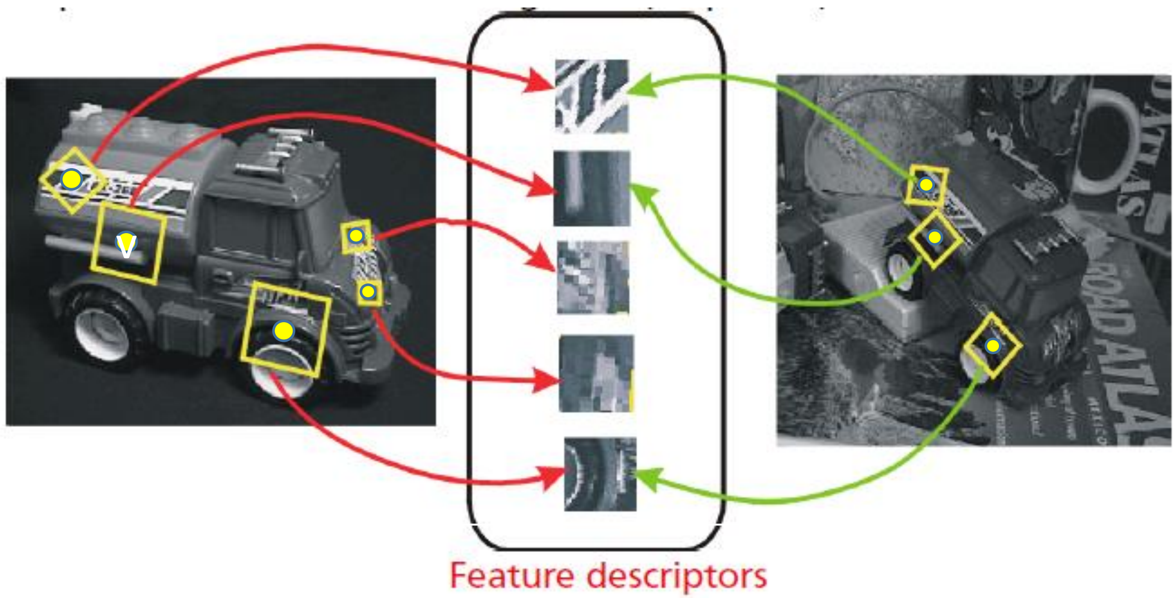
- Naïve: N^2 tests (e.g. 1 Million)
- Hashing
- KD-tree; on average NlogN tests (e.g. ~10,000)





Matching Points between two Images

- Find interest points
- Find orientated patches around interest points to capture appearance
- Encode patch appearance in a descriptor
- Find matching patches according to appearance (similar descriptors)
- Verify matching patches according to geometry (later lecture)



Roadmap for next four lectures

- Appearance-based Matching (sec. 4.1)
- Projective Geometry - Basics (sec. 2.1.1-2.1.4)
- Geometry of a Single Camera (sec 2.1.5, 2.1.6)
  - Camera versus Human Perception
  - The Pinhole Camera
  - Lens effects
- Geometry of two Views (sec. 7.2)
  - The Homography (e.g. rotating camera)
  - Camera Calibration (3D to 2D Mapping)
  - The Fundamental and Essential Matrix (two arbitrary images)
- Robust Geometry estimation for two cameras (sec. 6.1.4)
- Multi-View 3D reconstruction (sec. 7.3-7.4)
  - General scenario
  - From Projective to Metric Space
  - Special Cases

Some Basics

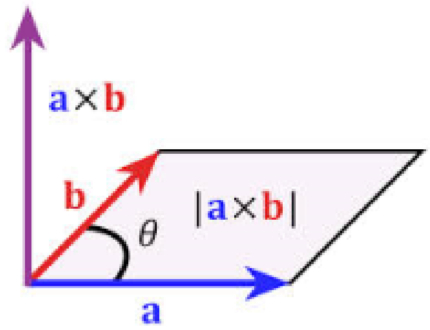
- Real coordinate space R^2 example: $\binom{1}{2}$
- Real coordinate space R^3 example: $\begin{pmatrix} 1 \\ 3 \\ 2 \end{pmatrix}$
- Operations we need are:

<u>scalar product:</u>

$$\boldsymbol{x}\,\boldsymbol{y} = \boldsymbol{x}^t\,\boldsymbol{y} = x_1 y_1 + x_2 y_2 + x_3 y_3 \quad \text{where } \boldsymbol{x} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}$$

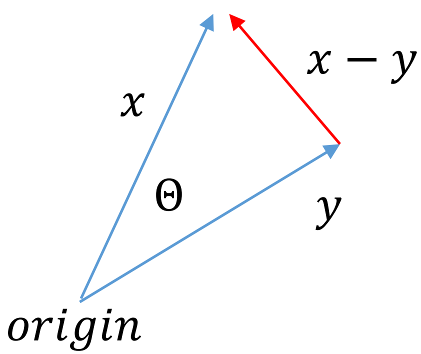
<u>cross/vector product:</u> $\boldsymbol{x} \times \boldsymbol{y} = [\boldsymbol{x}]_\times\, \boldsymbol{y}$

$$[\boldsymbol{x}]_\times = \begin{bmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{bmatrix}$$



Euclidean Space

- Euclidean Space R^2 and R^3 have angles and distances defined
  - Angle defined as: $\theta = \arccos\left(\frac{\boldsymbol{x}\,\boldsymbol{y}}{\|\boldsymbol{x}\|\|\boldsymbol{y}\|}\right)$
  - Length of the vector $\boldsymbol{x}$: $\|\boldsymbol{x}\| = \sqrt{\boldsymbol{x}\,\boldsymbol{x}}$
  - Distance of two vectors: $d(\mathbf{x},\mathbf{y}) = \|\mathbf{x}-\mathbf{y}\| = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$



Projective Space

- 2D Point in a real coordinate space:

$$\binom{1}{2} \in R^2 \text{ has 2 DoF (degrees of freedom)}$$

- 3D Point in a real coordinate space:

$$\begin{pmatrix} 1 \\ 3 \\ 2 \end{pmatrix} \in R^3 \text{ has 3 DoF}$$

- Definition: A point in 2-dimensional projective space P^2 is defined as $p = \begin{pmatrix} x \\ y \\ w \end{pmatrix} \in P^2$, such that all vectors $\begin{pmatrix} kx \\ ky \\ kw \end{pmatrix}$ $(\forall\, k \neq 0)$ define the same point p in P^2 (equivalent classes)

- Sometimes written as: $\begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} \sim \begin{bmatrix} 2 \\ 4 \\ 2 \end{bmatrix}$

- We write as: $\begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 \\ 4 \\ 2 \end{bmatrix} \in P^2$

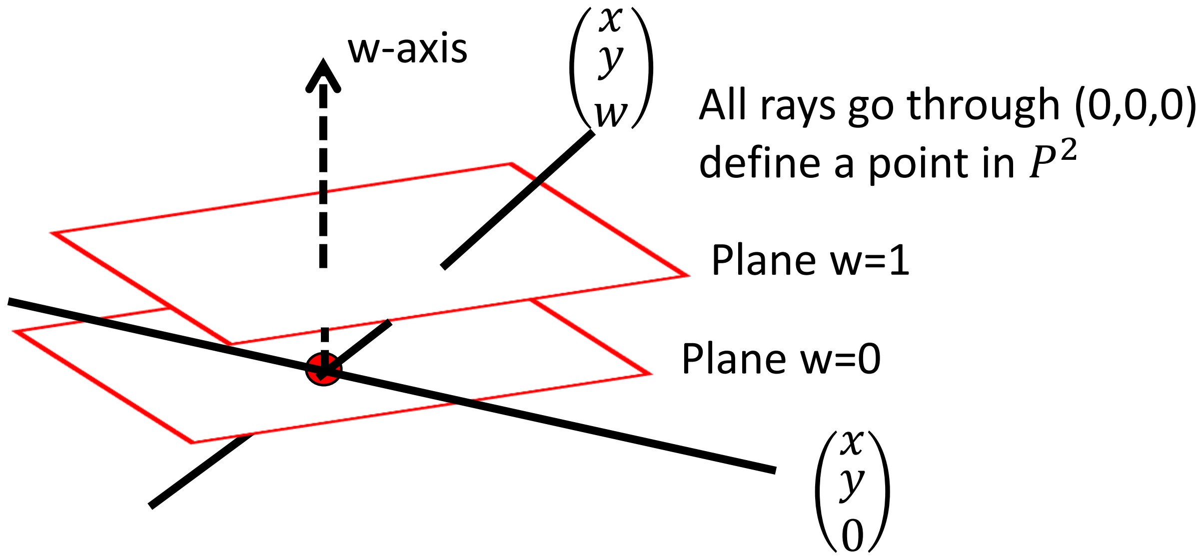
Projective Space - Visualization

<u>Definition</u>: A point in 2-dimensional projective space P^2 is defined as

$$p = \begin{pmatrix} x \\ y \\ w \end{pmatrix} \in P^2, \text{ such that all vectors } \begin{pmatrix} kx \\ ky \\ kw \end{pmatrix} \ (\forall\, k \neq 0)$$

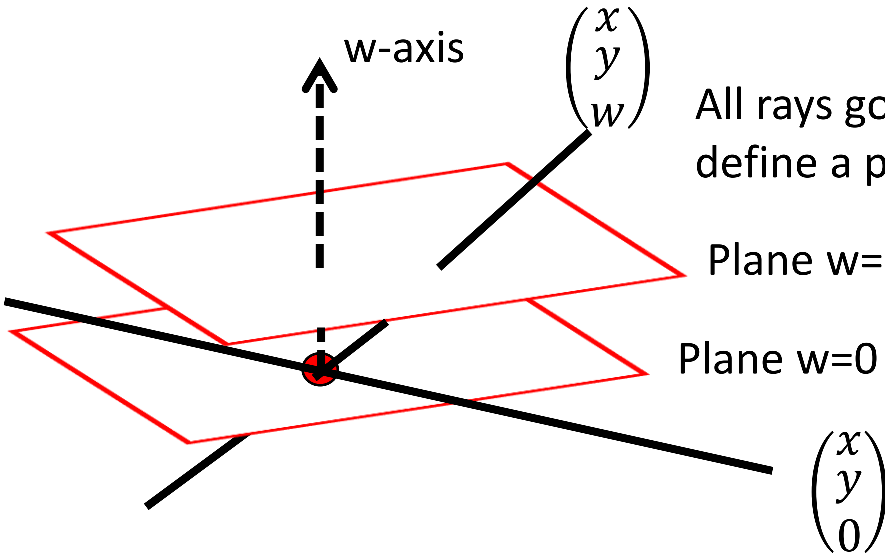
define the same point p in P^2 (equivalent classes)

A point in P^2 is a ray in R^3 that goes through the origin:



Projective Space

- All points in P^2 are given by: $R^3 \setminus \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$
- A point $\begin{pmatrix} x \\ y \\ w \end{pmatrix} \in P^2$ has 2 DoF (3 elements but norm of vector can be set to 1)



From R^2 to P^2 and back

- From R^2 to P^2:

$$\tilde{p} = \begin{pmatrix} x \\ y \end{pmatrix} \in R^2 \quad \Longrightarrow \quad p = \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \in P^2$$

- a point in inhomogeneous coordinates
- we soemtimes write $\tilde{p}$ for inhomogeneous coordinates

- a point in homogeneous coordinates

- From P^2 to R^2:

$$\begin{pmatrix} x \\ y \\ w \end{pmatrix} \in P^2 \quad \Longrightarrow \quad \begin{pmatrix} x/w \\ y/w \end{pmatrix} \in R^2$$

$$for\ w \neq 0$$

what does it mean if w=0?

We can do this transformation with all primitives (points, lines, planes)

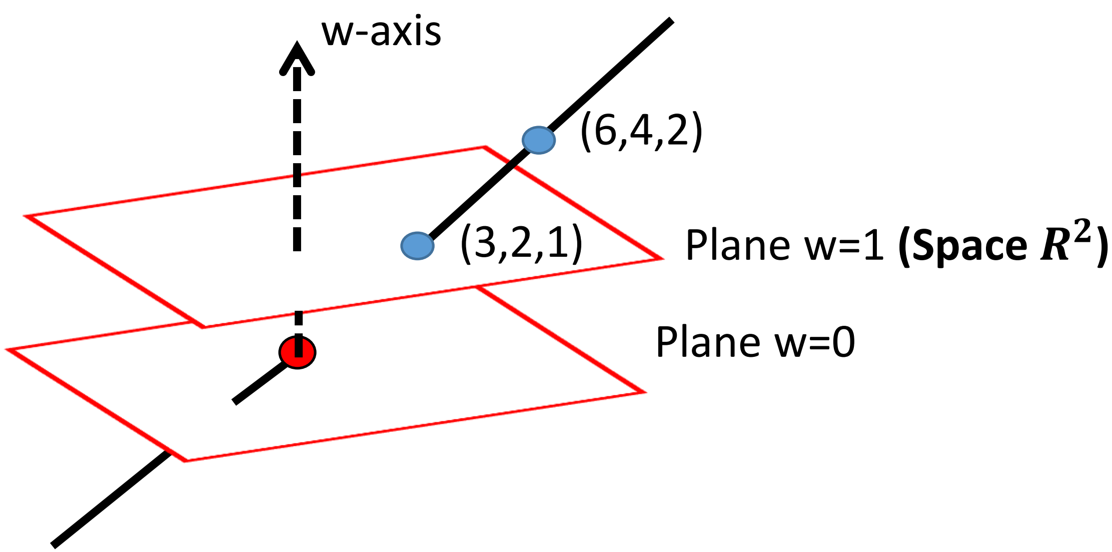
From R^2 to P^2 and back: Example

- From R^2 to P^2:

$$\tilde{p} = \begin{pmatrix} x \\ y \end{pmatrix} \in R^2 \quad \Rightarrow \quad p = \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \in P^2$$

- a point in inhomogeneous coordinates - a point in homogeneous coordinates

$$\tilde{p} = \begin{pmatrix} 3 \\ 2 \end{pmatrix} \in R^2 \quad \Rightarrow \quad p = \begin{pmatrix} 3 \\ 2 \\ 1 \end{pmatrix} = \begin{pmatrix} 4.5 \\ 3 \\ 1.5 \end{pmatrix} = \begin{pmatrix} 6 \\ 4 \\ 2 \end{pmatrix} \in P^2 \quad \Rightarrow \quad \tilde{p} = \begin{pmatrix} 6/2 \\ 4/2 \end{pmatrix} \in R^2$$



Halfway

3 Minutes Break

Things we would like to have and do

We look at these operations in: R^2/R^3, P^2/P^3

Primitives:

- Points in 2D/3D
- Lines in 2D/3D
- Planes in 3D
- Conic in 2D; Quadric in 3D

Operations with Primitives:

- Intersection
- Tangent

Transformations:

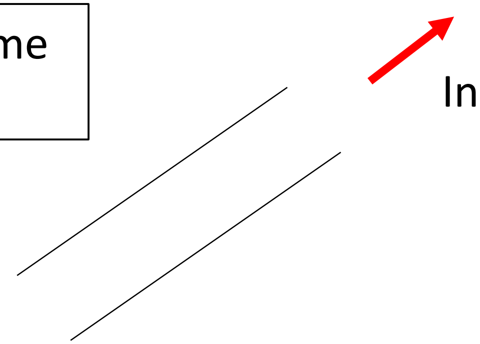
- Rotation
- Translation
- Projective
- ….

Why bother about P^2?

- All Primitives, operations and transformations are defined in R^2 and P^2

- <u>Advantage of P^2:</u>
  - Many transformation and operations are written more compactly (e.g. linear transformations)
  - We will introduce new special “primitives” that are useful when dealing with “parallelism”

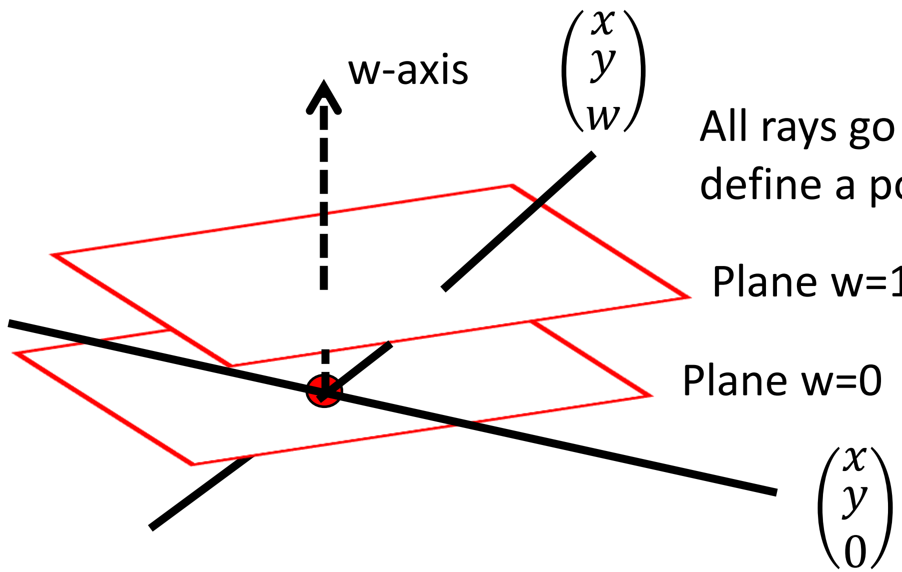
This example will come later in detail.



In P^2 they meet in a „point at inifinty“

Two parallel lines in R^2

Points at infinity



Points with coordinate $\begin{pmatrix} x \\ y \\ 0 \end{pmatrix}$ are ideal points or points at infinity

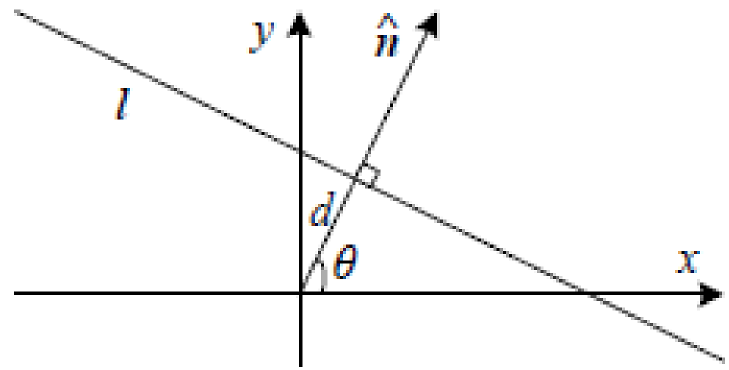
$\begin{pmatrix} x \\ y \\ 0 \end{pmatrix} \in P^2$ ⇒ Not defined in R^2 since $w = 0$

Lines in R^2

- For Lines in coordinate space R^2 we can write

  $l = (n_x, n_y, d)$ with $n = \left(n_x, n_y\right)^t$ is normal vector and $||n|| = 1$

- A line has 2 DoF



- A point (x, y) lies on l if:

  $n_x\, x + n_y\, y + d = 0$

- Normal can also be encoded

  with an angle θ:
  $n = (\cos\theta\,, \sin\theta)^t$

Lines in P^2

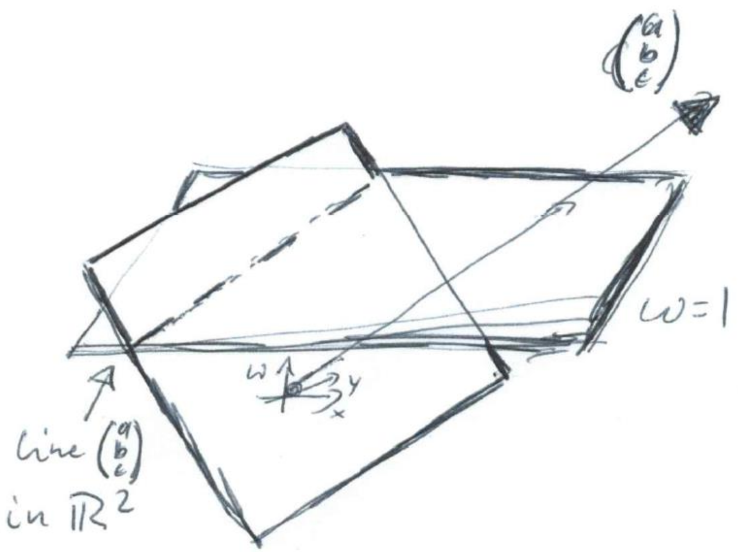
- Points in P^2: $\boldsymbol{x} = (x, y, w)$
- Lines in P^2: $\boldsymbol{l} = (a, b, c)$

  (again equivalent class: $(a, b, c) = (ka, kb, kc)\ \forall\ k \neq 0$)

  Hence also 2 DoF

- All points (x, y, w) on the line (a, b, c) satisfy: $ax + by + cw = 0$

this is the equation of a plane in R^3 with normal (a,b,c) going through (0,0,0)



"think of each line being represented by a vector (a, b, c)"

Converting Lines between P^2 and R^2

- Points in P^2: $\boldsymbol{x} = (x, y, w)$
- Lines in P^2: $\boldsymbol{l} = (a, b, c)$

From P^2 to R^2:

$\boldsymbol{l} = (ka, kb, kc)$ chose k such that $||(ka, kb)|| = 1$

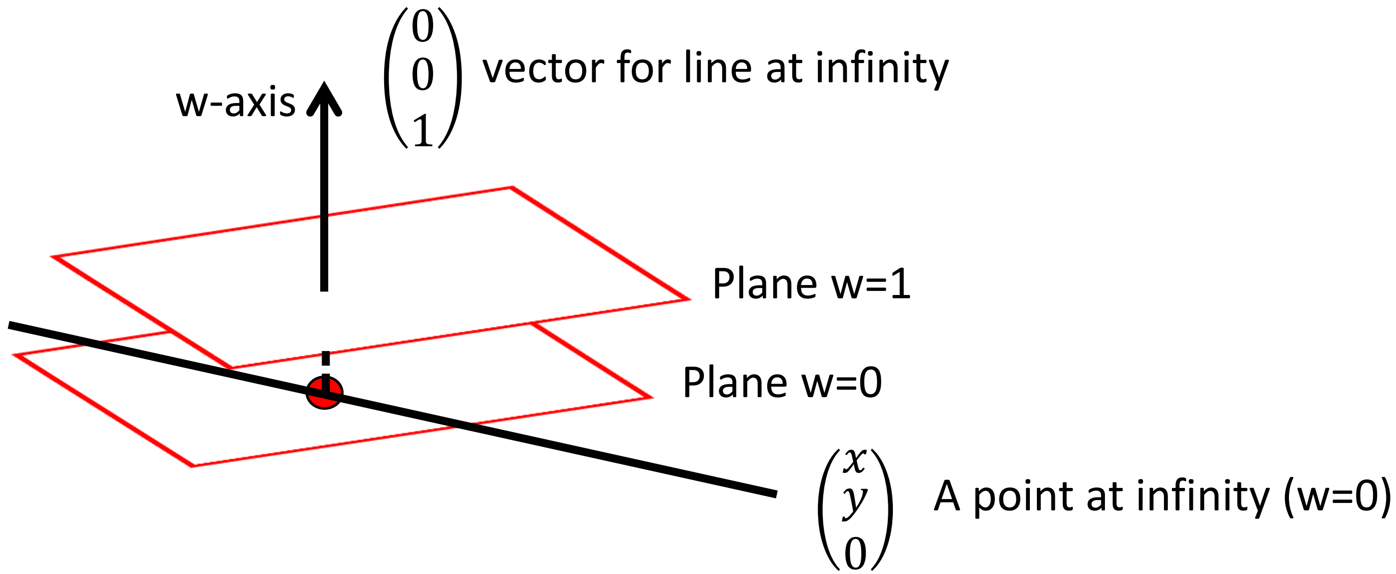
From R^2 to P^2:

$\boldsymbol{l} = (n_x, n_y, d)$ is already a line in P^2

Line at Infininty

- There is a “special” line, called line at infinity: $(0,0,1)$
- All points at infinity $(x, y, 0)$ lie on the line at infinity $(0,0,1)$:

  $x * 0 \; + \; y * 0 \; + \; 0 * 1 \; = \; 0$

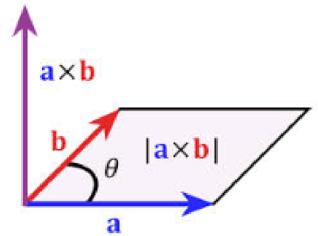


A Line is defined by two points in P^2

- The line through two points $\boldsymbol{x}$ and $\boldsymbol{x}'$ is given by $\boldsymbol{l} = \boldsymbol{x} \times \boldsymbol{x}'$
- Proof:

It is: $\boldsymbol{x}(\boldsymbol{x} \times \boldsymbol{x}') = \boldsymbol{x}' (\boldsymbol{x} \times \boldsymbol{x}') = \boldsymbol{0}$

vectors are orthogonal



This is the same as: $\boldsymbol{x}\,\boldsymbol{l} = \boldsymbol{x}'\boldsymbol{l} = \boldsymbol{0}$

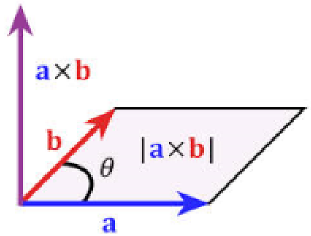
Hence, the line $\boldsymbol{l}$ goes through points $\boldsymbol{x}$ and $\boldsymbol{x}'$

The Intersection of two lines in P^2

- Intersection of two lines $\boldsymbol{l}$ and $\boldsymbol{l}'$ is the point $\boldsymbol{x} = \boldsymbol{l} \times \boldsymbol{l}'$
- Proof:

It is: $\boldsymbol{l}(\boldsymbol{l} \times \boldsymbol{l}') = \boldsymbol{l}'(\boldsymbol{l} \times \boldsymbol{l}') = \mathbf{0}$

vectors are orthogonal



This is the same as: $\boldsymbol{l}\boldsymbol{x} = \boldsymbol{l}'\boldsymbol{x} = \mathbf{0}$

Hence, the point $\boldsymbol{x}$ lies on the lines $\boldsymbol{l}$ and $\boldsymbol{l}'$

Note the „Theorem“ and Proofs have been very similiar, we only interchanged meaning of points and lines

Duality of points and lines

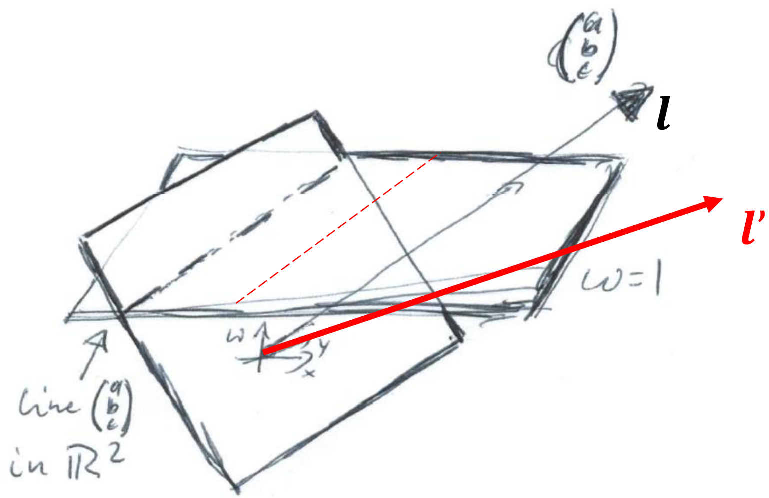
- Note $\boldsymbol{l}\boldsymbol{x} = \boldsymbol{x}\boldsymbol{l} = \mathbf{0}$ ($\boldsymbol{x}$ and $\boldsymbol{l}$ are "interchangeable")

- **Duality Theorem:** To any theorem of 2D projective geometry there corresponds a dual theorem which may be derived by interchanging the roles of points and lines in the original theorem.

  The intersection of two lines $\boldsymbol{l}$ and $\boldsymbol{l}'$ is the point $\boldsymbol{x} = \boldsymbol{l} \times \boldsymbol{l}'$

  The line through two points $\boldsymbol{x}$ and $\boldsymbol{x}'$ is the line $\boldsymbol{l} = \boldsymbol{x} \times \boldsymbol{x}'$

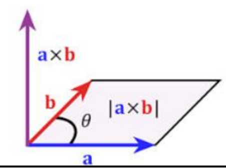
Parallel lines meet in a point at Infinity



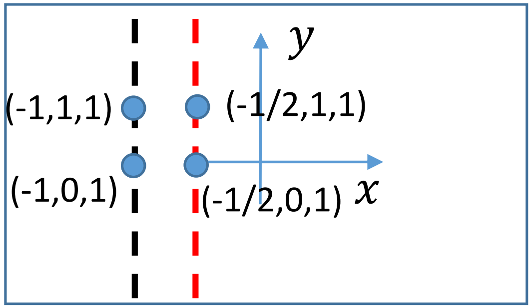
$$\boldsymbol{l} = \begin{pmatrix}1\\0\\1\end{pmatrix};\ \boldsymbol{l'} = \begin{pmatrix}2\\0\\1\end{pmatrix}$$

$$\boldsymbol{x} \times \boldsymbol{y} = [\boldsymbol{x}]_\times\ \boldsymbol{y}$$

$$[\boldsymbol{x}]_\times = \begin{bmatrix}0 & -x_3 & x_2\\ x_3 & 0 & -x_1\\ -x_2 & x_1 & 0\end{bmatrix}$$



$\boldsymbol{l}$ $\boldsymbol{l'}$ In R^2 (Plane $w = 1$)



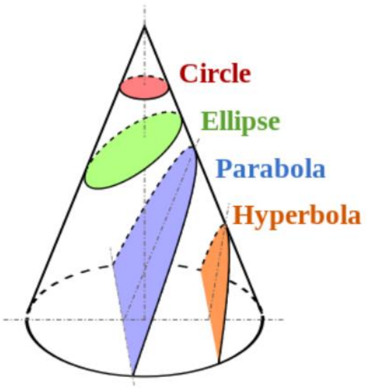
intersection

$$\boldsymbol{l} \times \boldsymbol{l'} = \begin{bmatrix}0 & -1 & 0\\ 1 & 0 & -1\\ 0 & 1 & 0\end{bmatrix}\begin{pmatrix}2\\0\\1\end{pmatrix} = \begin{pmatrix}0\\1\\0\end{pmatrix}$$

Point at infinty

2D conic "Kegelschnitt"

- Conics are shapes that arise when a plane intersects a cone
- In compact form: $\boldsymbol{x}^{\boldsymbol{t}}\boldsymbol{C}\,\boldsymbol{x} \;=\; \boldsymbol{0}$ where $\boldsymbol{C}$ has the form:



$$\boldsymbol{C} = \begin{bmatrix} a & b/2 & d/2 \\ b/2 & c & e/2 \\ d/2 & e/2 & f \end{bmatrix}$$

- This can be written in inhomogenous coordinates:

$$ax^2 + bxy + cy^2 + dx + ey + f = 0$$

where $\tilde{\boldsymbol{x}} = (x, y)$

- $\boldsymbol{C}$ has 5DoF since unique up to scale:

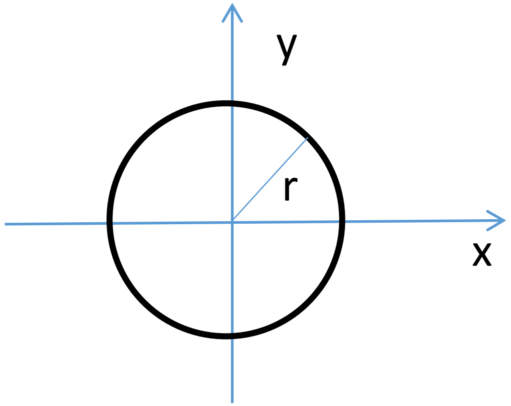
$$\boldsymbol{x}^{\boldsymbol{t}}\boldsymbol{C}\,\boldsymbol{x} \;= k\boldsymbol{x}^{\boldsymbol{t}}\boldsymbol{C}\,\boldsymbol{x} = \boldsymbol{x}^{\boldsymbol{t}}k\boldsymbol{C}\,\boldsymbol{x} = \boldsymbol{0}$$

- Properties: $\boldsymbol{l}$ is tangent to $\boldsymbol{C}$ at a point $\boldsymbol{x}$ if $\boldsymbol{l} \;=\; \boldsymbol{C}\boldsymbol{x}$

Example: 2D Conic

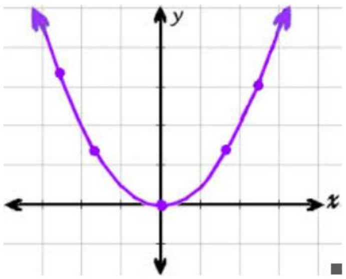
<u>A circle:</u>

$$x^2 + y^2 - r^2 = 0$$



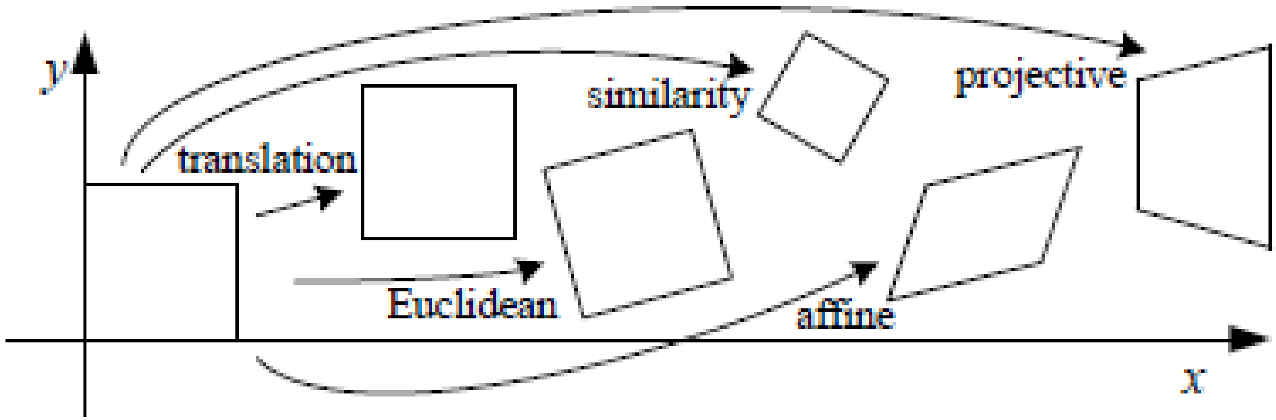
<u>Parabola:</u>

$$-x^2 + y = 0$$



2D Transformations

2D Transformations in R^2



<u>Definition:</u>

- Euclidean: translation + rotation
- Similarity (rigid body transform): Euclidean + scaling
- Affine: Similarity + shearing
- Projective: arbitrary linear transform in homogenous coordinates

2D Transformations of points

- 2D Transformations in homogenous coordinates:

$$\begin{pmatrix} x' \\ y' \\ w' \end{pmatrix} = \begin{bmatrix} a & b & d \\ e & f & h \\ i & j & l \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Transformation matrix

- Example: translation

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

homogeneous coordinates

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix}$$

inhomogeneous coordinates

Advantage of homogeneous coordinates (i.e. P^2)

2D Transformations of points

Group	Matrix	Distortion (of a square)	Invariant properties
Projective 8 dof	$\begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}$		Concurrency, collinearity, **order of contact**: intersection (1 pt contact); tangency (2 pt contact); inflections (3 pt contact with line); tangent discontinuities and cusps. cross ratio (ratio of ratio of lengths).
Affine 6 dof	$\begin{bmatrix} a_{11} & a_{12} & t_x \\ a_{21} & a_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$		Parallelism, ratio of areas, ratio of lengths on collinear or parallel lines (e.g. midpoints), linear combinations of vectors (e.g. centroids). The line at infinity, $\mathbf{l}_\infty$.
Similarity 4 dof	$\begin{bmatrix} sr_{11} & sr_{12} & t_x \\ sr_{21} & sr_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$		Ratio of lengths, angle. The circular points, **I**, **J** (see section 2.7.3). *(two special points on the line at infity)*
Euclidean 3 dof	$\begin{bmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$		Length, area

Here r_{ij} is a 2 x 2 rotation matrix with 1 DoF

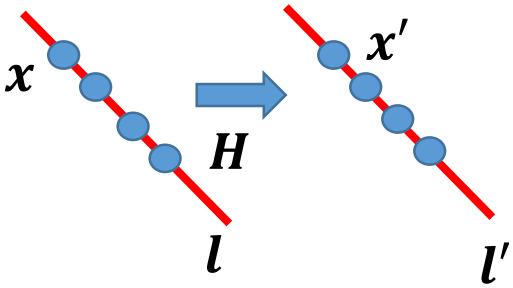
which can be written as: $\begin{bmatrix} \cos\Theta & -\sin\Theta \\ \sin\Theta & \cos\Theta \end{bmatrix}$

[from Hartley Zisserman Page 44]

2D transformations of lines and conics

All points move: $\boldsymbol{x}' = \boldsymbol{H}\boldsymbol{x}$ then:

1) Line (defined by points) moves:
$\boldsymbol{l}' = (\boldsymbol{H}^{-1})^{\mathrm{t}}\ \boldsymbol{l}$



2) conic (defined by points) moves:
$\boldsymbol{C}' = (\boldsymbol{H}^{-1})^{\mathrm{t}}\ \boldsymbol{C}\ \boldsymbol{H}^{-1}$

<u>Proof:</u>

1) Assume $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ lie on $\boldsymbol{l}$, and $\boldsymbol{l}' \underset{!}{=} (\boldsymbol{H}^{-1})\ \ \boldsymbol{l}$.
Show that $\boldsymbol{x}'_1, \boldsymbol{x}'_2$ lie on $\boldsymbol{l}'$.
$(\boldsymbol{x}'_1)^t\ \boldsymbol{l}' \overset{!}{=} 0 \rightarrow (\boldsymbol{H}\boldsymbol{x}_1)^t (\boldsymbol{H}^{-1})^t \boldsymbol{l} = 0 \rightarrow$
$\boldsymbol{x}_1^t \boldsymbol{H}^t (\boldsymbol{H}^{-1})^t \boldsymbol{l} \overset{!}{=} 0 \rightarrow \boldsymbol{x}_1^t (\boldsymbol{H}^{-1}\boldsymbol{H})^t \boldsymbol{l} \overset{!}{=} 0 \rightarrow \boldsymbol{x}_1^t \boldsymbol{l} \overset{!}{=} 0$

2) Homework.

Example: Projective Transformation

Picture from top

1. Circles on the floor are circles in the image
2. Squares on the floor are squares in the image

Affine transformation

1. Circles on the floor are ellipse in the image
2. Squares on the floor are sheared in the image
3. Lines are still parallel

Picture from the side (projective transformation)

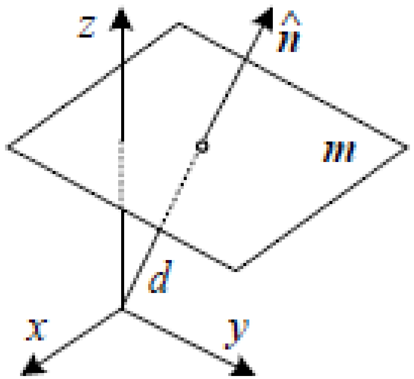
1. Lines converge to a vanishing point (not at infinity in the image)

In 3D: Points

- $\boldsymbol{x} = (x, y, z) \in R^3$ has 3 DoF
- With homogeneous coordinates: $(x, y, z, 1) \in P^3$
- P^3 is defined as the space $R^3 \setminus (0,0,0,0)$ such that points $(x, y, z, w)$and (kx, ky, kz, kw) are the same for all $k \neq 0$
- Points: $(x, y, z, 0) \in P^3$ are called points at infinity

In 3D: Planes

- Planes in R^3 are defined by 4 paramters (3 DoF):
  - Normal: $n = (n_x, n_y, n_z)$
  - Offset: d



- All points (x, y, z) lie on the plane if:

  $x\, n_x + y\, n_y + z\, n_z + d = 0$

- With homogenous coordinates:

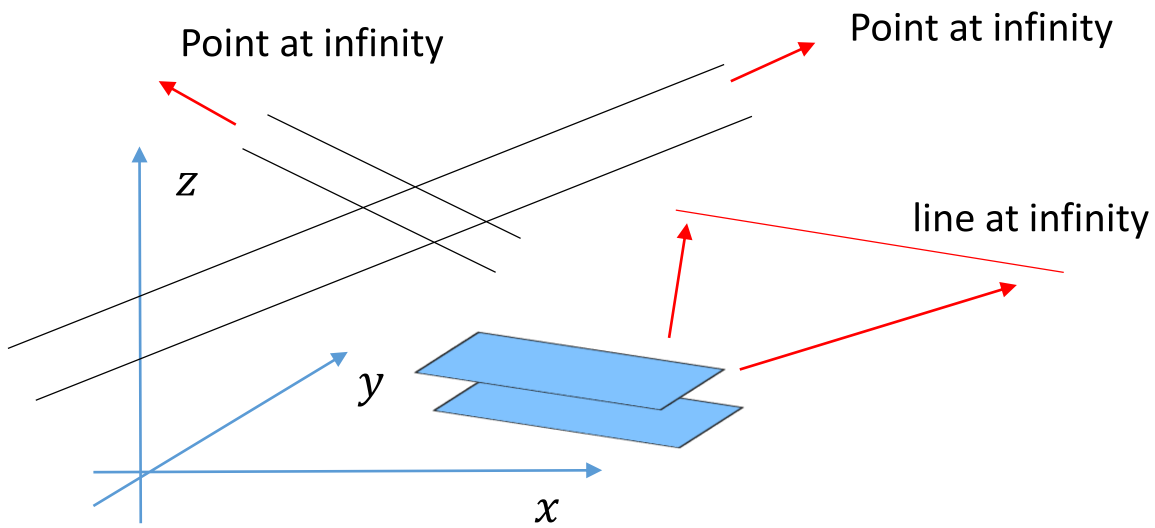
  $\boldsymbol{x}\, \pi = 0$, where $x = (x, y, z, 1)$ and $\pi = (n_x, n_y, n_z, d)$

- Planes in P^3 are written as: $\boldsymbol{x}\, \pi = 0$

- Points and planes are dual in P^3 (as points and lines have been in P^2)

- Plane at infinity is π = (0,0,0,1) since all points at infinity (x,y,z,0) lie on it.

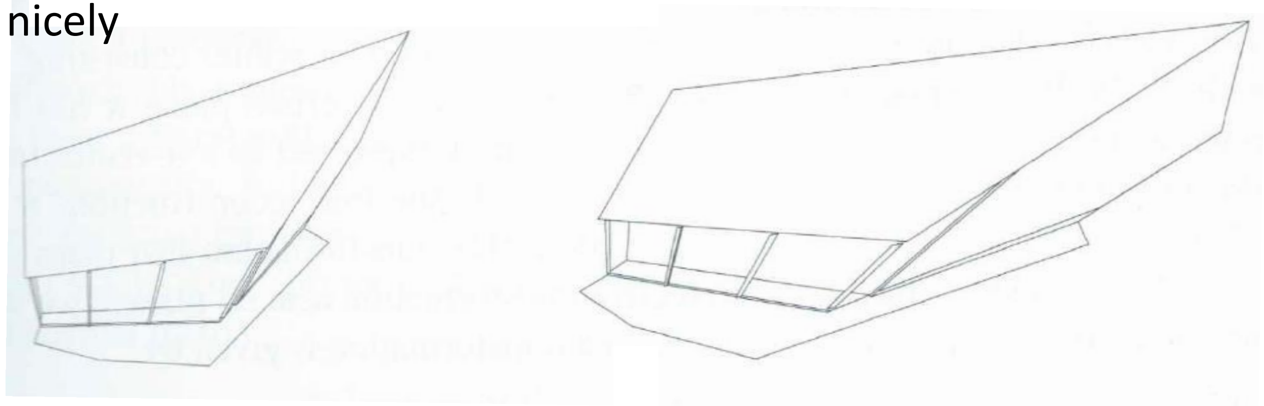
In 3D: Plane at infinity



All of these elements at infinity lie on the plane at infinity

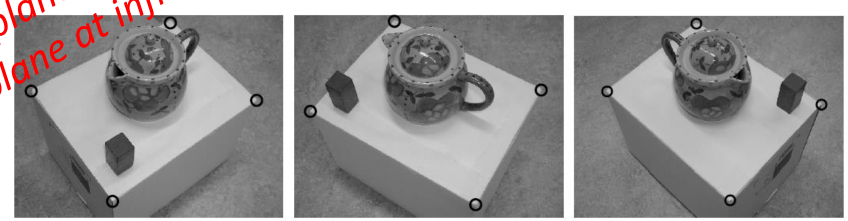
Why is the plane at infinity important (see later)

Plane at infinity is important to visualize 3D reconstructions nicely

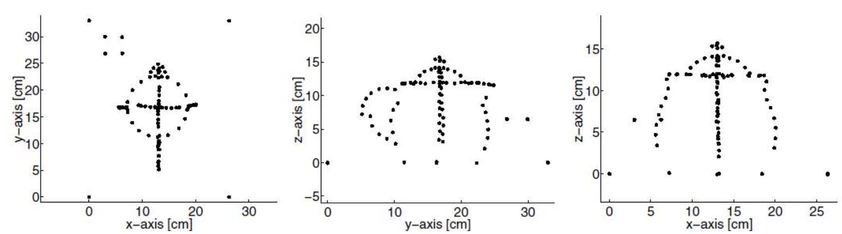


Plane at infinity can be used to simplify 3D reconstruction

Real plane defined as plane at infinity

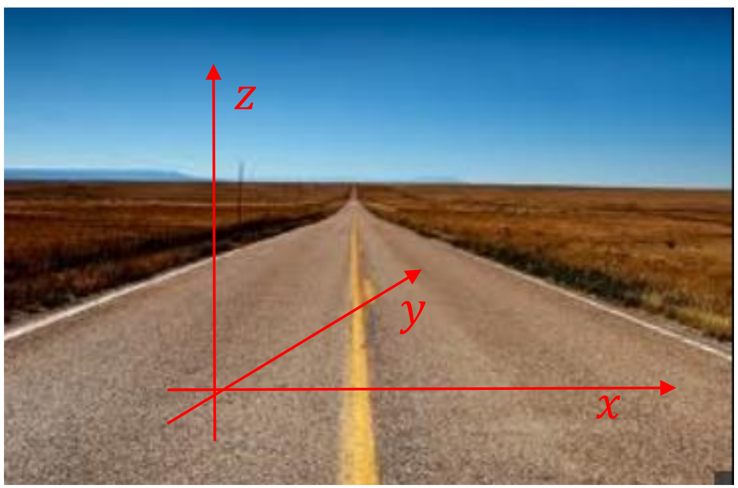


(a) (b) (c)



What is the horizon?

- The ground plane is special (we stand on it)
- Horizon is a line at infinity where plane at infinity intersects ground plane



Ground plane: (0,0,1,0)
Plane at infinity: (0,0,0,1)

Many lines and planes in our real world meet at the horizon (since parallel to ground plane)

In 3D: Points at infinity

- Points at infinity can be real points in a camera

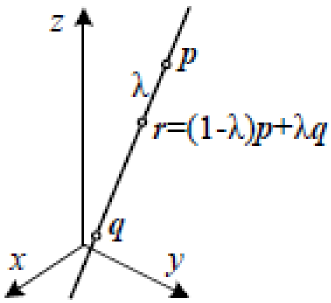
$$\begin{pmatrix} b \\ f \\ 1 \end{pmatrix} = \begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \end{bmatrix} \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$$

Real point in the image

3x4 Camera Matrix 3D->2D projection

3D Point at infinty

In 3D: Lines



- Unfortunately not a compact form (as for points)

- A simple representation in R^3. Define a line via two points $p, q \in R^3$:

$$\boldsymbol{r} = (1-\lambda)\boldsymbol{p} + \lambda\,\boldsymbol{q}$$

- A line has 4 DoF (both points $\boldsymbol{p}, \boldsymbol{q}$ can move arbitrary on the line)

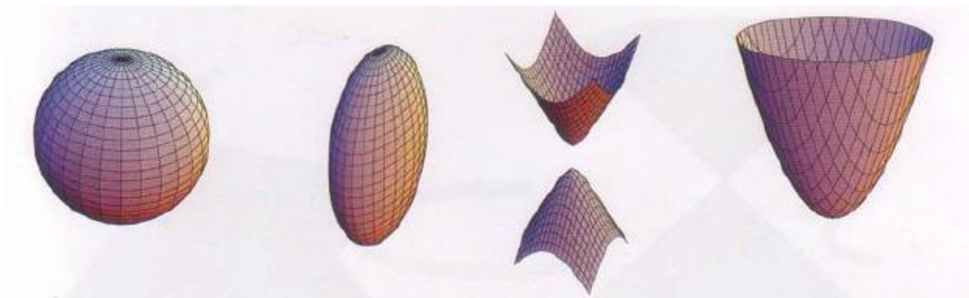
- A more compact, but more complex, way two define a 3D Line is to use Plücker coordinates:

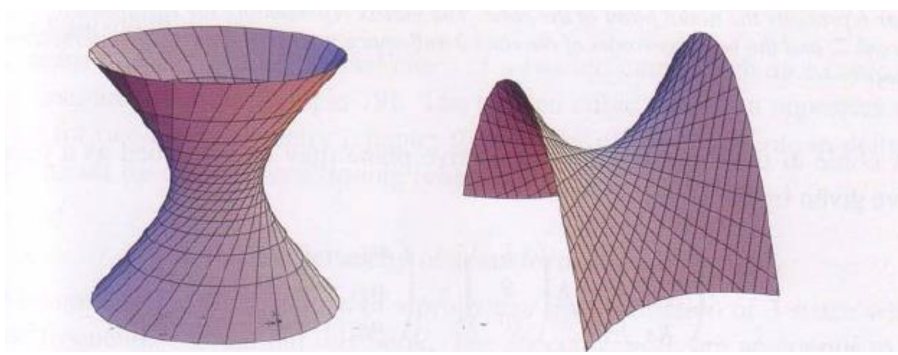
$$\boldsymbol{L} = \boldsymbol{p}\boldsymbol{q}^{\boldsymbol{t}} - \boldsymbol{q}\boldsymbol{p}^{\boldsymbol{t}} \text{ where } \det(\boldsymbol{L}) = 0$$

here $\boldsymbol{L}, \boldsymbol{p}, \boldsymbol{q}$ are in homogenous coordinates

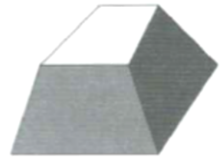
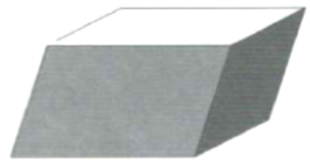
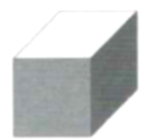
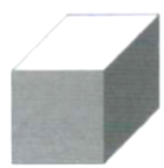
In 3D: Quadrics

- Points $\boldsymbol{X}$ on the quadric if: $\boldsymbol{X}^T\ \boldsymbol{Q}\ \boldsymbol{X}\ =\ 0$
- A quadric $\boldsymbol{Q}$ is a surface in P^3
- A quadric is a symmetric 4×4 matrix with 9 DoF





In 3D: Transformation

Group	Matrix	Distortion (of a cube)	Invariant properties
Projective 15 dof	$\begin{bmatrix} \mathtt{A} & \mathbf{t} \\ \mathbf{v}^\mathsf{T} & v \end{bmatrix}$		Intersection and tangency of surfaces in contact. Sign of Gaussian curvature.
Affine 12 dof	$\begin{bmatrix} \mathtt{A} & \mathbf{t} \\ \mathbf{0}^\mathsf{T} & 1 \end{bmatrix}$		Parallelism of planes, volume ratios, centroids. The plane at infinity, π_∞, (see section 3.5).
Similarity 7 dof	$\begin{bmatrix} s\mathtt{R} & \mathbf{t} \\ \mathbf{0}^\mathsf{T} & 1 \end{bmatrix}$		The absolute conic, Ω_∞, (see section 3.6).
Euclidean 6 dof	$\begin{bmatrix} \mathtt{R} & \mathbf{t} \\ \mathbf{0}^\mathsf{T} & 1 \end{bmatrix}$		Volume.

In 3D: Rotations

Rotation $\boldsymbol{R}$ in 3D has 3 DoF. It is slightly more complex, and several options exist:

1) Euler angles: rotate around, x, y, z-axis in order
   (depends on order, not smooth in parameter space)

2) Axis/angle formulation:

   $$\boldsymbol{R}(\boldsymbol{n}, \Theta) = \boldsymbol{I} + \sin\Theta\,[\boldsymbol{n}]_\times + (1 - \cos\Theta)[\boldsymbol{n}]_\times^2$$

   $\boldsymbol{n}$ is the normal vector (2 DoF) and Θ the angle (1 DoF)

3) Another option is unit quaternions (see book page 40)