Geometric definition of homography

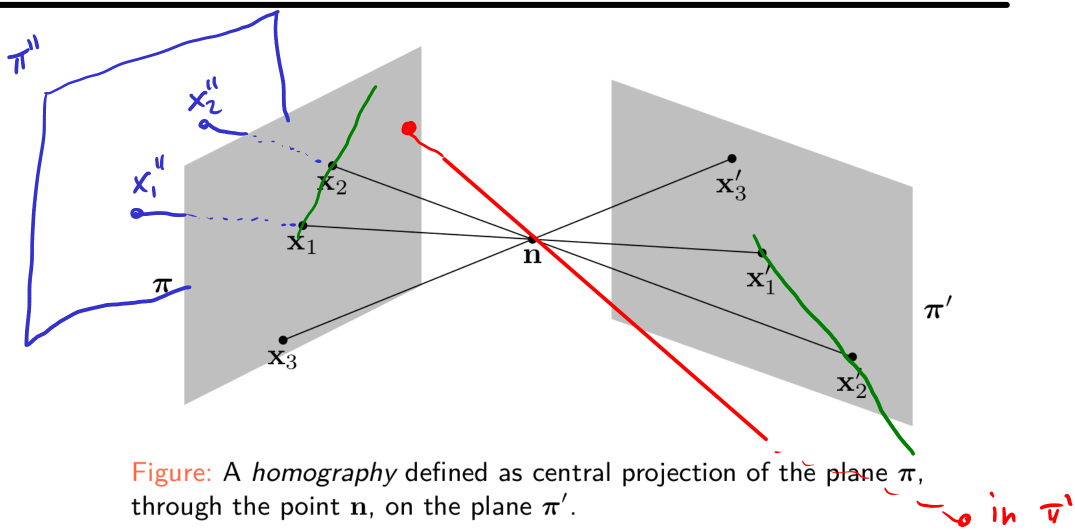


Figure: A *homography* defined as central projection of the plane π, through the point $\mathbf{n}$, on the plane π'.

Observations

- A homography is always invertible.
- A homography becomes a perspective pinhole camera, if everything in the scene lies in a plane.
- The two planes can lie on the 'same side' of the projection centre.
- A homography maps straight lines in one image to straight lines in the other image.

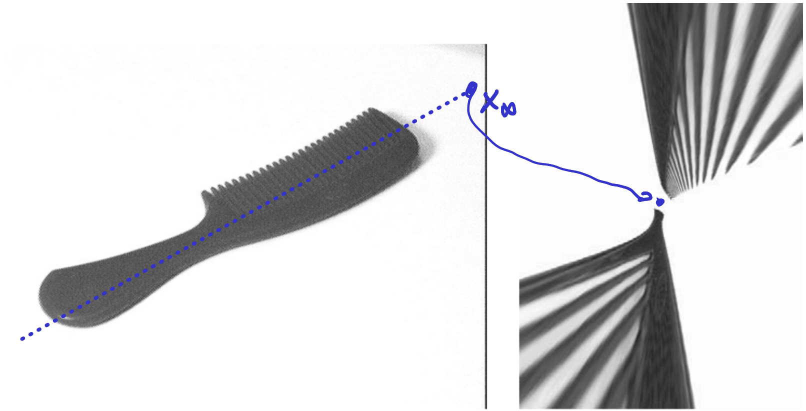
The "only" invariant of a homography!

- A homography can map proper points to ideal points, and vice versa.

Example of a homography

Figure: The right image is a synthetic view generated from the left image by applying a suitable homography.

Example of a homography (crazy case)



a b

(From Hartley & Zisserman, "Multiple View Geometry in Computer Vision", 2004)

Algebraic representation of homographies

A homography is represented by an invertible 3x3 matrix,

$x' \sim H_1 x \iff x' \sim H_2 x$

and

$H_1 \sim H_2$

all 3x3 invertible matrices represent homographies.

$\Rightarrow H_1$ and H_2 represent the same transform

Generalises affine transforms:

recall: $T_{affine} = \begin{pmatrix} A & t \\ 0^T & 1 \end{pmatrix}$

$H = \begin{pmatrix} A & t \\ c^T & d \end{pmatrix}$

but must be invertible!

Invariants:

Affine	homographies
parallel lines	straight lines

Degrees of Freedom: Not 9! Eight! (remove one for scaling)

Homographies & Cameras

Theorem centre: $P'r' = 0 \Rightarrow r' \sim \begin{pmatrix} -A^{-1}b \\ 1 \end{pmatrix}$

Assume $\mathbf{P} = [\mathbf{I} \;\; \mathbf{0}]$ *and* $\mathbf{P}' = [\mathbf{A} \;\; \mathbf{b}]$ *are projective cameras, and let* $\boldsymbol{\pi} = (\boldsymbol{\nu}, d)$ *be a plane which does not contain any of the camera centres. If* $\mathbf{X}$ *is a point on the plane and projects into the two views as* $\mathbf{x} \sim \mathbf{P}\mathbf{X}$ *and* $\mathbf{x}' \sim \mathbf{P}'\mathbf{X}$*, then*

$$\mathbf{x}' \sim (\mathbf{A} - \mathbf{b}\boldsymbol{\nu}^\top / d)\mathbf{x},$$

and the matrix $\mathbf{H} = \mathbf{A} - \mathbf{b}\boldsymbol{\nu}^\top / d$ *is a homography.*

splitting up H into R, t, n is the "homography decomposition problem"

Assumption: $P = K[I \;\; 0]$ and $P' = K[R \;\; t]$

Homographies & Cameras (contd.)

Proof (except invertibility):

We know $x \sim PX = \underset{3\times 4}{(I \ \ 0)} \underset{4\times 1}{X} = (I \ \ 0)\underbrace{\begin{pmatrix} x \\ s \end{pmatrix}}_{X(s)}$

In the second camera:

$$x' \sim P'X(s) = (A \ \ b)\begin{pmatrix} x \\ s \end{pmatrix} = Ax + bs$$

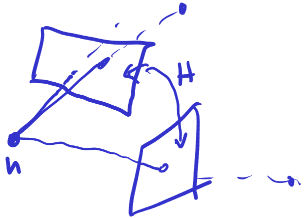
Since $X(s)$ lies on π, we have $\pi^T X(s) = 0 \Leftrightarrow (v^T \ \ d)\begin{pmatrix} x \\ s \end{pmatrix} = 0 \Leftrightarrow v^T x + ds = 0 \Rightarrow s = -\frac{v^T x}{d}$

note: $d \neq 0$ by design

$$\Rightarrow x' \sim Ax + bs = Ax - \frac{b v^T x}{d} = \left(A - \frac{b v^T}{d}\right) x \quad \therefore$$

Cameras with identical centre

Consider $P = (I\ 0)$ and $P' = (A\ 0)$.



Exactly the situation on the first slide!

Generate panoramas!

Point Correspondences

In many cases it is useful to consider point correspondences between two (or more) images.

$$x_j \leftrightarrow x'_j , \quad j = 1, \ldots, n$$

- There exist many methods for automatically finding (putative) point correspondences, based on feature extraction (SIFT, SURF, ...).

- Not all correspondences they propose are correct! ("Outliers", more on that in TSBB15.)

Direct Linear Transformation (DLT)

$[u]_\times v = u \times v$

Suppose we are given a number of point correspondences $x_j \leftrightarrow x_j'$

and want to find a homography that transforms all x_j into x_j'.

Equations: We want $x_j' \overset{\text{not an equation!}}{\sim} H x_j$

Note: $x_j' \in \mathbb{R}^3,\ Hx_j \in \mathbb{R}^3$

Formulate using cross product!

$$\Rightarrow\ x_j' \times Hx_j = 0 \Leftrightarrow \underbrace{[x_j']_\times}_{\text{known}} \overbrace{H}^{\text{unknowns!}} \underbrace{x_j}_{\text{known}} = 0$$

Pull H outside using vec/kronecker!

$$\Rightarrow \underbrace{\left(x_j^T \otimes [x_j']_\times\right)}_{A_j} \operatorname{vec} H = 0$$

DLT constraint

$x_j = (u_j, v_j, 1),\quad x_j' = (u_j', v_j', 1)$

$$A_j = \left(u_j[x_j']_\times \quad v_j[x_j']_\times \quad [x_j']_\times \right) = \left(\begin{array}{ccc|ccc|ccc} 0 & -u_j & u_j v_j' & 0 & -v_j & v_j v_j' & 0 & -1 & v_j' \\ u_j & 0 & -u_j u_j' & v_j & 0 & -v_j u_j' & 1 & 0 & -u_j' \\ -u_j v_j' & u_j u_j' & 0 & -v_j v_j' & v_j u_j' & 0 & -v_j' & u_j' & 0 \end{array} \right)$$

Conclusion: Cumbersome! Better to use Kronecker...

Direct Linear Transformation (DLT) (contd.)

We stack all DLT constraints for $j=1,\dots,n$ into a "data matrix" A:

$$\underset{3\times 9}{A_j = x_j^T \otimes [x_j']_\times} \qquad \underset{3n\times 9}{A = \begin{pmatrix} A_1 \\ \vdots \\ A_n \end{pmatrix}}$$

DLT system: $A \,\mathrm{vec} H = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$ 3n zeros

Note: $\operatorname{rank} A_j = 2 \Rightarrow \operatorname{rank} A \le 2n$

But $\operatorname{rank} A \le 9$

$\operatorname{rank} A \le \min(2n, 9)$

Four point correspondences gives "unique" solution H