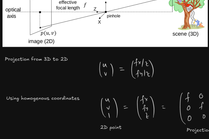
Perspective Projection

From 3D to 2D:



Projection from 3D to 2D:

$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} fx/z \\ fy/z \end{pmatrix}$$

Using homogenous coordinates:

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} fx/z \\ fy/z \\ 1 \end{pmatrix} \Leftarrow \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

2D point — Projection matrix — 3D point

Physical point p $\begin{pmatrix} x \\ y \end{pmatrix}$ represented as $\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$ $\quad x = u/w$, $y = v/w$ (add another coordinate)

Two homogenous coordinate vectors are equivalent if they are proportional to each other, i.e, they only differ by a scale factor

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \lambda \begin{bmatrix} u \\ v \\ w \end{bmatrix} \qquad \lambda \neq 0$$

Transformation	Matrix (Euclidean)	Matrix (homogenous coordinates)	Degrees of freedom	Illustration
Translation	$\mathbf{y} = \mathbf{x} + \mathbf{t}$	$\begin{bmatrix} \mathbf{I} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix}$	2	lengths, angles
Rotation	$\mathbf{y} = \mathbf{R}\mathbf{x}$ $\mathbf{R}^T\mathbf{R} = \mathbf{R}\mathbf{R}^T = \mathbf{I}$ $\det(\mathbf{R}) = +1$	$\begin{bmatrix} \mathbf{R} & \mathbf{0} \\ \mathbf{0}^T & 1 \end{bmatrix}$	1	lengths, angles
Rigid	$\mathbf{y} = \mathbf{R}\mathbf{x} + \mathbf{t}$	$\begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix}$	6	lengths, angles
Affine	$\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{t}$	$\begin{bmatrix} \mathbf{A} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix}$	12	ratios of lengths, parallelism
Projective		4×4 matrix $\boldsymbol{M}$	15	collinearity, incidence

Camera Intrinsics:

The intrinsic parameters describe the internal characteristics of the camera and how a 3D point in the camera's coordinate system is projected onto the image plane

$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} fx/z \\ fy/z \end{pmatrix} \quad \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} fx/z \\ fy/z \\ 1 \end{pmatrix} = \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

when camera center does not coincide with the optical axis, and its center is at (c_x, c_y)

$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} fx/z + c_x \\ fy/z + c_y \end{pmatrix} \quad \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} fx/z + c_x \\ fy/z + c_y \\ 1 \end{pmatrix} = \begin{pmatrix} f & 0 & c_x & 0 \\ 0 & f & c_y & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

If camera pixels are not square but rectangular instead, (not a problem in most modern cameras)

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} \alpha f & 0 & c_x & 0 \\ 0 & \beta f & c_y & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

Image skew

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} \alpha f & s & c_x & 0 \\ 0 & \beta f & c_y & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

(not a problem in most modern cameras)

Projection matrix

only 1 axis is skewed

$$P = \begin{bmatrix} K & 0 \\ 0^T & 1 \end{bmatrix} \begin{bmatrix} R & t \\ 0^T & 1 \end{bmatrix} p$$

$$K = \begin{bmatrix} \alpha f & s & c_x \\ 0 & \beta f & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

upper triangular matrix
K is camera intrinsics
5 parameters (5 DOF)

A projection, $P_{3\times4}$ maps homogenous 3D world coordinates to the view of a camera in homogenous 2D coordinates.

Camera Extrinsics:

The extrinsic parameters describe the relationship between the world coordinate system and the camera coordinate system. These parameters consist of a rotation and a translation that transform points from the 3D world coordinate frame to the camera's coordinate frame.

Rotation matrix R: A 3×3 orthogonal matrix representing the rotation from the world coordinate frame to the camera coordinate frame.
Translation vector t: A 3×1 vector that represents the translation of the camera relative to the world origin.

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \underbrace{\begin{pmatrix} \alpha f & s & c_x \\ 0 & \beta f & c_y \\ 0 & 0 & 1 \end{pmatrix}}_{\text{Intrinsics}} \underbrace{\begin{pmatrix} R & t \\ 0^T & 1 \end{pmatrix}}_{\text{Extrinsics}} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

C = actual location of camera in world coordinate frame; $\tilde{C}$ is not a homogenous coordinate, it is an actual euclidean 3D coordinate
C is subtracted to bring world coordinate frame back to the origin

Total DOF = 5 + 3 + 3 = 11
= 12 − 1 (scale does not matter)

Any point along a ray gets projected to the same point on the image

$$P\tilde{x} = (K|0)\begin{pmatrix} R & -R\tilde{C} \\ 0^T & 1 \end{pmatrix} P = MP$$

$$\Rightarrow (K|0)\begin{pmatrix} R & -R\tilde{C} \\ 0^T & 1 \end{pmatrix} P = (K|0)\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

$$M = \begin{bmatrix} m_1^T \\ m_2^T \\ m_3^T \end{bmatrix}_{3\times4}$$

$$P = KR[I|-\tilde{C}]P = KR[I|-t]P$$

$$\lambda p = KR[I|-t]P$$

rank of M = 3, rank of A = 3, implies $\det(A)$ is not zero
if rank is 2, then we go from 3D point to 1D

$$u = \frac{m_1^T P}{m_3^T P} \qquad v = \frac{m_2^T P}{m_3^T P}$$

Important Observations:
- Straight lines in 3D may be straight lines in 2D
- Object size is inversely proportional to distance
- Parallel lines seem to meet at a point (vanishing point)
- Length magnification = $\alpha = f/z$, Area magnification α^2

Homography

Assume planar world
Parameterize plane using 2 parameters p, q. x coordinate is 0: $\begin{bmatrix} 0 \\ p \\ q \end{bmatrix}$

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ p \\ q \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} \begin{bmatrix} 0 \\ p \\ q \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{12} & r_{13} & t_x \\ r_{22} & r_{23} & t_y \\ r_{32} & r_{33} & t_z \end{bmatrix} \begin{bmatrix} p \\ q \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f r_{12} & f r_{13} & f t_x \\ r_{22} & r_{23} & t_y \\ r_{32} & r_{33} & t_z \end{bmatrix} \begin{bmatrix} p \\ q \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} p \\ q \\ 1 \end{bmatrix}$$

$\Downarrow$ homography (planar projective transformation)
specialized case to map points from one plane to another

Planar homographs maps the view of one camera in 2D homogenous coordinates to the view of another camera in 2D homogenous coordinates.

Purpose:
Homographies are used to describe the transformation of points across images when the observed scene is planar, meaning you can map one image entirely onto the other using a homography matrix. The key idea is that when the camera moves or rotates, but the object of interest lies on a single plane, the transformation between images is governed by a homography.

Estimating Homography

$$(x', y') = \left(\frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + h_{33}}, \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + h_{33}} \right)$$

$$u_i = \frac{h_{11}x_i + h_{12}y_i + h_{13}}{h_{31}x_i + h_{32}y_i + h_{33}} \qquad v_i = \frac{h_{21}x_i + h_{22}y_i + h_{23}}{h_{31}x_i + h_{32}y_i + h_{33}}$$

u_i, v_i, x_i, y_i are known

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -u_1x_1 & -u_1y_1 & -u_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -v_1x_1 & -v_1y_1 & -v_1 \\ \vdots & & & & & & & & \vdots \\ x_n & y_n & 1 & 0 & 0 & 0 & -u_nx_n & -u_ny_n & -u_n \\ 0 & 0 & 0 & x_n & y_n & 1 & -v_nx_n & -v_ny_n & -v_n \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{bmatrix} = 0$$

$\Rightarrow 4$ points

8 DOF $\Rightarrow$ we need 8 rows
$n = 4$(rows)

requires more corresponding points to solve

$$Ah = 0$$

choose at least 8 points.
Perform SVD (Singular Value Decomposition)

Even if we have more than four correspondences (providing more rows in A), the structure of the homography problem leads to dependencies between the equations, which means the matrix A will not be full rank. This is because the transformation described by a homography has a degree of freedom (scaling factor), leading to one singular value of zero.

Important points:
- If scene rotates about its center, then the images are related by a homography irrespective of scene depth
- If scene is planar, then images from cameras at any two positions and orientations are related by a homography
- Homography mapping is a 3×3 matrix with 8 DOF
- We have one point if we know the equation of plane
- We have to find points and their correspondences

RANSAC

Random Sample Consensus

Steps for computing RANSAC:

1. Input Data
Input your data:
- Two sets of corresponding points $x1$ and $x2$, typically obtained from feature matching algorithms (like SIFT or SURF).
- A threshold t for determining whether a point is an inlier.
- The maximum number of iterations k to perform.

2. Repeat for k iterations:
- Random Sampling: Randomly select 4 point correspondences from the full set of matched points between the two images.
A minimum of 4 points is needed because homography has 8 unknowns (degrees of freedom), and each point correspondence provides two equations.

- Compute the Homography Matrix: Using the 4 randomly selected point correspondences, estimate the homography matrix H.
The homography matrix is computed by solving the system of equations derived from the transformation:

$$x_2 = Hx_1$$

3. Inlier Counting:
- Once the homography matrix H is computed, apply the transformation H to all point correspondences in the dataset. For each point correspondence $(x1[i], transform x1 using H

$$x_i' = Hx_i$$

Then, calculate the reprojection error for each point:

$$\text{Error} = \|x_i - x_i'\|$$

If the error is below a given threshold t, consider the point as an inlier. Count the total number of inliers.

4. Best Model Selection:
Keep track of the model (homography matrix) that results in the highest number of inliers. After k iterations, the homography with the largest inlier set is selected as the best model.

s = probability that no 'good' set of correspondences will be found after t trials
t = number of trials necessary to achieve failure probability α
n = number of correspondences drawn in each trial
r = percentage of inliers (100% − all the correspondences are right)

$$t = \frac{(1-r)^n}{s}$$

$$t = \frac{\log \alpha}{\log(1-r^n)} \qquad \alpha = 1-s$$

PnP - Perspective n-Point

$$p = MP$$

$$u_i = \frac{m_1^T P_i}{m_3^T P_i} \qquad v_i = \frac{m_2^T P_i}{m_3^T P_i}$$

$$m_1^T P_i - (m_3^T P_i) u_i = 0$$

$$m_2^T P_i - (m_3^T P_i) v_i = 0$$

$$\begin{bmatrix} P_i^T & 0 & -u_i P_i^T \\ 0 & P_i^T & -v_i P_i^T \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \end{bmatrix} = 0$$

Put all the relations for all the points into a single matrix:

$$\begin{bmatrix} P_1^T & 0 & -u_1 P_1^T \\ 0 & P_1^T & -v_1 P_1^T \\ \vdots & & \\ P_n^T & 0 & -u_n P_n^T \\ 0 & P_n^T & -v_n P_n^T \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \end{bmatrix} = 0 \qquad Am = 0$$

$$A^T A$$

Solve by minimizing: $\|Am\|^2 = m^T A^T A m \quad L = \begin{bmatrix} v_1^T & -u_1 P_1^T \\ \vdots & \vdots \\ v_n P_n^T & -u_n P_n^T \end{bmatrix}$

Subject to: $\|m\| = 1$

If A is symmetric, the minimum of $m^T A m$ is reached at
m = eigenvector of A corresponding to the smallest eigenvalue

Calibration: Minimizing Reprojection Error

$$S = \sum_i \left(u_i - \frac{m_1^T P_i}{m_3^T P_i} \right)^2 + \left(v_i - \frac{m_2^T P_i}{m_3^T P_i} \right)^2$$

minimize the least squares error - done using non-linear optimization techniques like gradient descent or Levenberg-Marquardt

Distortion

Distortion needs to be undone before camera calibration

$$x_c = \frac{L_1 m_1^T P_i}{L_3 m_3^T P_i} \qquad y_c = \frac{L_2 m_2^T P_i}{L_3 m_3^T P_i} \longrightarrow \text{non-linear}$$

make these linear

$$v_i \left(m_1^T P_i \right) - u_i \left(m_2^T P_i \right) = 0$$

$$\begin{bmatrix} v_1 P_1^T & -u_1 P_1^T \\ \vdots & \vdots \\ v_n P_n^T & -u_n P_n^T \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \end{bmatrix} = 0 \qquad m = \begin{bmatrix} m_1 \\ m_2 \end{bmatrix}$$

$$\begin{bmatrix} v_1 P_1^T & -u_1 P_1^T \\ \vdots & \vdots \\ v_n P_n^T & -u_n P_n^T \end{bmatrix} m = 0$$

Solve by minimizing: $\|Lm\|^2 = m^T L^T L m \quad L = \begin{bmatrix} v_1 P_1^T & -u_1 P_1^T \\ \vdots & \vdots \\ v_n P_n^T & -u_n P_n^T \end{bmatrix}$

Subject to: $\|m\| = 1$