Pinhole camera

and homogeneous coordinates

Morten R. Hannemose, mohan@dtu.dk

February 3, 2023

02504 Computer vision course lectures,
DTU Compute, Kgs. Lyngby 2800, Denmark

Learning objectives

After this lecture you should be able to:

- explain homogeneous coordinates
- convert to and from homogeneous coordinates
- perform relevant coordinate transformations
- explain the pinhole camera model

Presentation topics

- Pinhole camera
- Perspective transformations
- Euclidean (rigid) transformations
- Homogeneous coordinates
  - Lines in homogenous coordinates
  - Summary of homogeneous coordinates
- Pinhole camera model
  - Intrinsics
  - Extrinsics

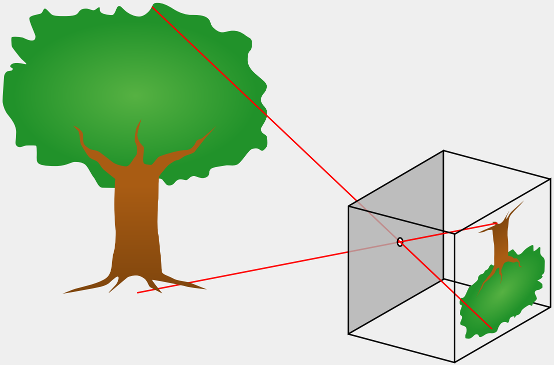
Pinhole camera

What is a “good” camera model?

. . . in terms of accuracy vs. ease-of-use?

1. As accurate as possible
2. As easy to use as possible
3. somewhere between 1 and 2

Pinhole camera



The projected image appears upside down

Each point in an image corresponds to a direction from the camera

Real life example

Can I get two volunteers?

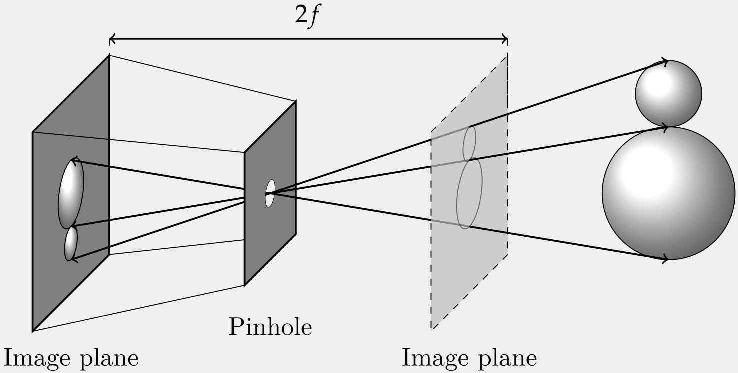
Real life example

Can I get two volunteers?

- A point seen in a single camera must be along a specific line in the other camera.
- Seeing the same point in two cameras is enough to find the point in 3D.

Perspective transformations

Pinhole camera again

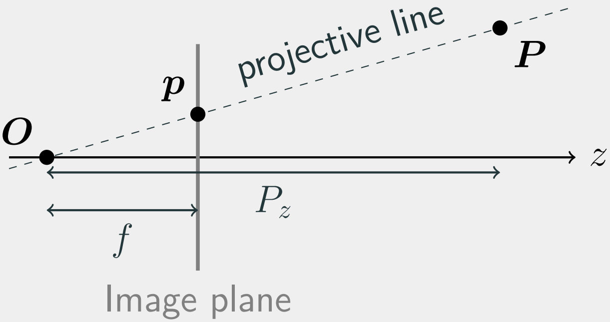


When modelling we place the "image plane" in front of the camera.
The distance from image plane to camera is the focal length f.

Perspective projection

$$\boldsymbol{P} = \begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix},$$

$$\boldsymbol{p} = \frac{f}{P_z} \begin{bmatrix} P_x \\ P_y \end{bmatrix}$$

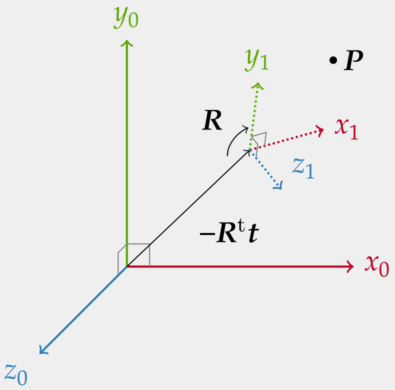


Euclidean (rigid) transformations

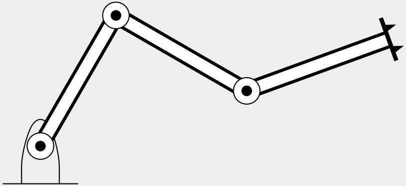
Euclidean (rigid) transformations: rotations and translations

Rotation matrix $\boldsymbol{R}$ and translation vector $\boldsymbol{t}$

$$\boldsymbol{P}_1 = \boldsymbol{t} + \boldsymbol{R}\boldsymbol{P}_0$$



Robot arm transformations



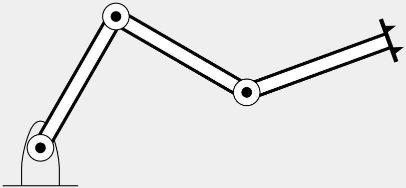
$$\boldsymbol{P}_1 = \boldsymbol{R}_1 \boldsymbol{P}_0 + \boldsymbol{t}_1$$

$$\boldsymbol{P}_2 = \boldsymbol{R}_2 \boldsymbol{P}_1 + \boldsymbol{t}_2$$

$$\boldsymbol{P}_3 = \boldsymbol{R}_3 \boldsymbol{P}_2 + \boldsymbol{t}_3$$

$$\boldsymbol{P}_4 = \boldsymbol{R}_4 \boldsymbol{P}_3 + \boldsymbol{t}_4$$

Robot arm transformations



$$\boldsymbol{P}_4 = \boldsymbol{R}_4(\boldsymbol{R}_3(\boldsymbol{R}_2(\boldsymbol{R}_1\boldsymbol{P}_0 + \boldsymbol{t}_1) + \boldsymbol{t}_2) + \boldsymbol{t}_3) + \boldsymbol{t}_4$$
$$\boldsymbol{P}_4 = \boldsymbol{R}_4\boldsymbol{R}_3\boldsymbol{R}_2\boldsymbol{R}_1\boldsymbol{P}_0 + \boldsymbol{R}_4\boldsymbol{R}_3\boldsymbol{R}_2\boldsymbol{t}_1 + \boldsymbol{R}_4\boldsymbol{t}_3 + \boldsymbol{R}_4\boldsymbol{R}_3\boldsymbol{t}_2 + \boldsymbol{t}_4$$

Homogeneous coordinates

Homogeneous coordinates

Here is a point in 3D:

$$\boldsymbol{P} = \begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix}$$

Homogeneous coordinates

Here is a point in 3D:

$$\boldsymbol{P} = \begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix}$$

What if we made it more complicated and used four numbers?

$$\boldsymbol{P_h} = \begin{bmatrix} sP_x \\ sP_y \\ sP_z \\ s \end{bmatrix}$$

Uhm, okay?

So this means that

$$\begin{bmatrix} 1 \\ 2 \\ 3 \\ 1 \end{bmatrix}, \begin{bmatrix} 2 \\ 4 \\ 6 \\ 2 \end{bmatrix}$$

are the same point in homogeneous coordinates

Euclidean transformations again

Rotation $\boldsymbol{R} = \begin{bmatrix} \boldsymbol{r}_1 & \boldsymbol{r}_2 & \boldsymbol{r}_3 \end{bmatrix}$, with columns $\boldsymbol{r}_i$, and translation $\boldsymbol{t}$

$$\boldsymbol{P}_1 = \boldsymbol{R}\boldsymbol{P}_0 + \boldsymbol{t}$$

Euclidean transformations again

Rotation $\boldsymbol{R} = \begin{bmatrix} \boldsymbol{r}_1 & \boldsymbol{r}_2 & \boldsymbol{r}_3 \end{bmatrix}$, with columns $\boldsymbol{r}_i$, and translation $\boldsymbol{t}$

$$\boldsymbol{P}_1 = \boldsymbol{R}\boldsymbol{P}_0 + \boldsymbol{t} = \boldsymbol{r}_1 P_x + \boldsymbol{r}_2 P_y + \boldsymbol{r}_3 P_z + \boldsymbol{t}$$

Euclidean transformations again

Rotation $\boldsymbol{R} = \begin{bmatrix}\boldsymbol{r}_1 & \boldsymbol{r}_2 & \boldsymbol{r}_3\end{bmatrix}$, with columns $\boldsymbol{r}_i$, and translation $\boldsymbol{t}$

$$\boldsymbol{P}_1 = \boldsymbol{R}\boldsymbol{P}_0 + \boldsymbol{t} = \boldsymbol{r}_1 P_x + \boldsymbol{r}_2 P_y + \boldsymbol{r}_3 P_z + \boldsymbol{t}$$

$$\boldsymbol{P}_1 = \begin{bmatrix}\boldsymbol{r}_1 & \boldsymbol{r}_2 & \boldsymbol{r}_3 & \boldsymbol{t}\end{bmatrix} \begin{bmatrix}P_x \\ P_y \\ P_z \\ 1\end{bmatrix}$$

Euclidean transformations again

Rotation $\boldsymbol{R} = \begin{bmatrix} \boldsymbol{r}_1 & \boldsymbol{r}_2 & \boldsymbol{r}_3 \end{bmatrix}$, with columns $\boldsymbol{r}_i$, and translation $\boldsymbol{t}$

$$\boldsymbol{P}_1 = \boldsymbol{R}\boldsymbol{P}_0 + \boldsymbol{t} = \boldsymbol{r}_1 P_x + \boldsymbol{r}_2 P_y + \boldsymbol{r}_3 P_z + \boldsymbol{t}$$

$$\boldsymbol{P}_1 = \underbrace{\begin{bmatrix} \boldsymbol{r}_1 & \boldsymbol{r}_2 & \boldsymbol{r}_3 & \boldsymbol{t} \end{bmatrix}}_{\text{Transform: } \tilde{\boldsymbol{T}}} \underbrace{\begin{bmatrix} P_x \\ P_y \\ P_z \\ 1 \end{bmatrix}}_{\text{Homogeneous: } \boldsymbol{P}_{0h}} = \tilde{\boldsymbol{T}} \boldsymbol{P}_{0h}$$

Homogeneous euclidean transformations

Fully homogeneous euclidean transformations become

$$\boldsymbol{P}_{1h} = \boldsymbol{T}\boldsymbol{P}_{0h}$$

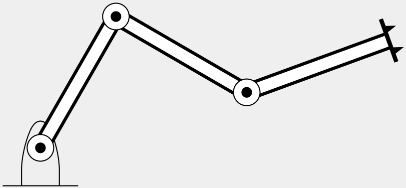
$$\begin{bmatrix} \boldsymbol{P}_1 \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0} & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{P}_0 \\ 1 \end{bmatrix}$$

Homogeneous euclidean transformations

The homogeneous transformation $\boldsymbol{T}$ takes on the 4×4 form

$$\boldsymbol{T} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0} & 1 \end{bmatrix}$$

Robot arm and homogeneous transformations



$$\boldsymbol{Q}_{4h} = \boldsymbol{T}_4\boldsymbol{T}_3\boldsymbol{T}_2\boldsymbol{T}_1\boldsymbol{Q}_{0h}$$

Not just easier; It is computationally faster!

The *in*homogeneous p corresponds to the homogeneous p_h by

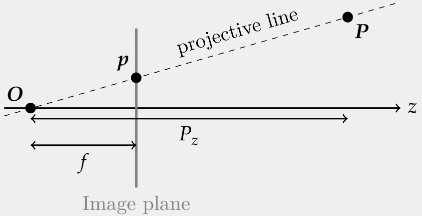
$$\boldsymbol{p}_h = \begin{bmatrix} s\boldsymbol{p} \\ s \end{bmatrix}$$

The projective transformation

Assume $f = 1$:

$$p_x = \frac{P_x}{P_z}, \quad p_y = \frac{P_y}{P_z}$$

$$\boldsymbol{P} = \begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix} = \begin{bmatrix} s p_x \\ s p_y \\ s \end{bmatrix} = \boldsymbol{p}_h$$



Projective transformation is like assuming point in 3D is a 2D homogeneous point.

There is no standard notation for homogeneous coordinates

$$\boldsymbol{q} = \begin{bmatrix} s\boldsymbol{p} \\ s \end{bmatrix}$$

Lines in homogenous coordinates

$$0 = ax + by + c$$

If we have a point in homogeneous coordinates

$$= \begin{bmatrix} a \\ b \\ c \end{bmatrix} \cdot \begin{bmatrix} sx \\ sy \\ s \end{bmatrix}$$

$$= \boldsymbol{l} \cdot \boldsymbol{p}_h$$

Lines in homogenous coordinates

$$0 = ax + by + c$$

If we have a point in homogeneous coordinates

$$= \begin{bmatrix} a \\ b \\ c \end{bmatrix} \cdot \begin{bmatrix} sx \\ sy \\ s \end{bmatrix}$$

$$= \boldsymbol{l} \cdot \boldsymbol{p}_h$$

If $a^2 + b^2 = 1$ then $\boldsymbol{l} \cdot \boldsymbol{p}_h$ yields the closest (signed) *distance* from

The homogeneous coordinate system — summary

The additional imaginary scale s, or alternatively dimension w

$$\boldsymbol{u} = s \begin{bmatrix} \boldsymbol{v} \\ 1 \end{bmatrix} = \begin{bmatrix} s\boldsymbol{v} \\ s \end{bmatrix} = \begin{bmatrix} \boldsymbol{v}' \\ w \end{bmatrix}$$

- Dimensionality is $N + 1$
- Points have a scale $s \neq 0$
- Directions have $w = 0$
- Many-to-one correspondence: $\boldsymbol{u} \in \mathbb{R}^{N+1}$ and $\boldsymbol{v} \in \mathbb{R}^{N}$

Getting *in*homogeneous coordinates back

$$\boldsymbol{v} = \Pi\left(\boldsymbol{u}\right) = \Pi\left(\begin{bmatrix}\boldsymbol{v}' \\ s\end{bmatrix}\right) = \boldsymbol{v}'/s$$

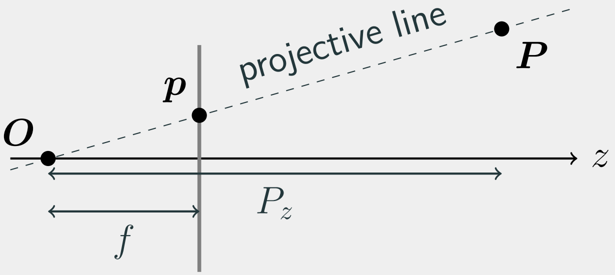
Trivial inverse

Pinhole camera model

Pinhole camera

Image on your computer

Image from perspective projection



Where is (0, 0) in these images?

Principal point

- Image on your computer
  - Upper left corner
- Image from projective transformation
  - Optical centre (approximately. in the middle of the image)

We introduce δ_x and δ_y to move these points to the same place.

This is called the principal point.

Principal point

Projection is now

$$p_x = \frac{f}{P_z} P_x + \delta_x$$
$$p_y = \frac{f}{P_z} P_y + \delta_y$$

Can we write all of this using homogeneous coordinates?

Principal point

Yes we can!

$$\boldsymbol{p}_h = \underbrace{\begin{bmatrix} f & 0 & \delta_x \\ 0 & f & \delta_y \\ 0 & 0 & 1 \end{bmatrix}}_{\boldsymbol{K}} \boldsymbol{P}$$

And it even looks nice! This is called the camera matrix.

It contains intrinsic camera parameters.

Extrinsics

- The extrinsics of the camera are the rotation $(\boldsymbol{R})$ and translation $(\boldsymbol{t})$.
- They describe the pose of the camera.

Extrinsics

- The extrinsics of the camera are the rotation $(\boldsymbol{R})$ and translation $(\boldsymbol{t})$.
- They describe the pose of the camera.
- To project points, we first transform them to the reference frame of the camera:

$$\begin{aligned} \boldsymbol{P}_{cam} &= \boldsymbol{R}\boldsymbol{P} + \boldsymbol{t} \\ &= \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \end{bmatrix} \begin{bmatrix} \boldsymbol{P} \\ 1 \end{bmatrix} \end{aligned}$$

Projection matrix

Projecting a single point in 3D to the camera

$$\boldsymbol{p}_h = \boldsymbol{K}\boldsymbol{P}_{cam}$$

Projection matrix

Projecting a single point in 3D to the camera

$$\begin{aligned} \boldsymbol{p}_h &= \boldsymbol{K}\boldsymbol{P}_{cam} \\ &= \underbrace{\boldsymbol{K}\begin{bmatrix}\boldsymbol{R} & \boldsymbol{t}\end{bmatrix}}_{\mathscr{P}}\boldsymbol{P}_h \end{aligned}$$

Projection matrix

Projecting a single point in 3D to the camera

$$\begin{aligned} \boldsymbol{p}_h &= \boldsymbol{K}\boldsymbol{P}_{cam} \\ &= \underbrace{\boldsymbol{K}\begin{bmatrix}\boldsymbol{R} & \boldsymbol{t}\end{bmatrix}}_{\mathscr{P}}\boldsymbol{P}_h = \end{aligned}$$

$$\begin{bmatrix} sp_x \\ sp_y \\ s \end{bmatrix} = \begin{bmatrix} f & 0 & \delta_x \\ 0 & f & \delta_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} \begin{bmatrix} P_x \\ P_y \\ P_z \\ 1 \end{bmatrix}$$

Wrapping up

- The translation is not the position of the camera
- The final coordinate of $\boldsymbol{P}_h$ must be 1.

Projection matrix:

$$\boldsymbol{q} = \boldsymbol{\mathcal{P}} \boldsymbol{P}_h = \boldsymbol{K} \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \end{bmatrix} \boldsymbol{P}_h$$

The matrix $\boldsymbol{\mathcal{P}}$ is known as the projection matrix

(don't call it the camera matrix)

Exercise information

- Use Python interactively
  - Jupyter notebook
  - VS Code
  - Spyder
  - etc.
- Makes it easier to debug

Exercise information: Storing points on the computer

- Storing multiple one-dimensional vectors happens frequently
- Matrices are ideal for this
- We always operate on column vectors, so these matrices should be $3 \times n$ for a 3D vector (for example)
- Matrix multiplication lets you project many points at once
  - `ph = P.dot(Ph)` or even shorter
  - `ph = P@Ph`

Comment about exercises

- If you need for-loops, you're probably not doing it the easy way.
  - No exercises today need for-loops (except the provided function)
- Converting from homogeneous coordinates to regular
  - `p = ph[:-1]/ph[-1]`
- Ask the TAs

Learning objectives

After this lecture you should be able to:

- explain homogeneous coordinates
- convert to and from homogeneous coordinates
- perform relevant coordinate transformations
- explain the pinhole camera model

Exercise time!