Camera calibration

How to recover 3-dimensional structure of a scene from its images.

pixel → full metric reconstruction

- position of the camera with respect to the world coordinate frame
- orientation of the camera with respect to the world coordinate frame

external parameters

- how the camera maps, via perspective projection, points in the world on the image plane

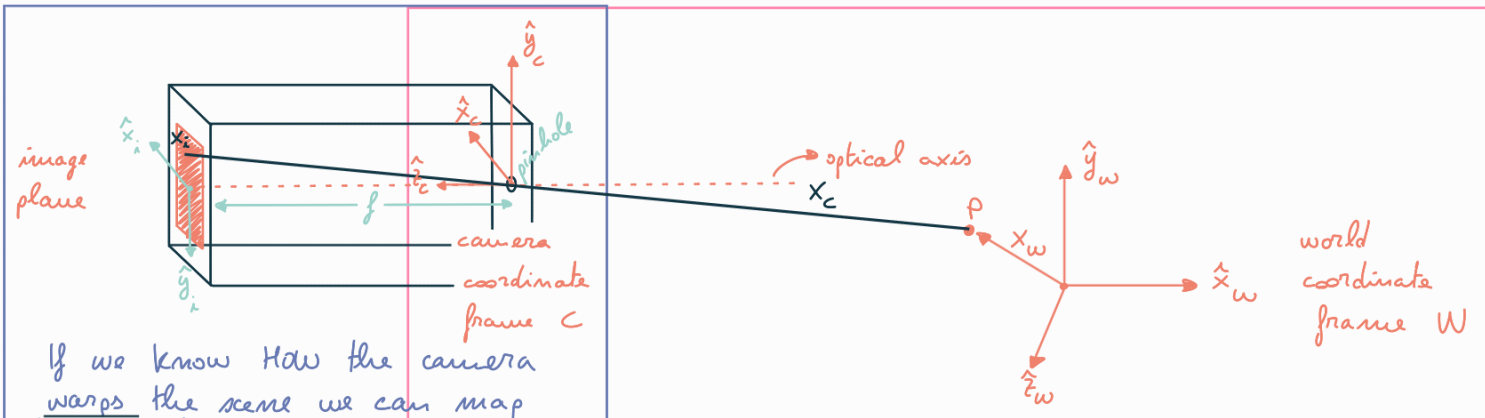
internal parameters

Determining internal and external parameters is what is referred to as camera calibration.

Before developing a method to find a camera's internal and external parameters we need a camera model.

A camera model takes you from a point in 3D to it's projection in pixels in the image. We want the model to be linear (easier). The linear camera model is called projection matrix.

How to develop a model of the camera (find the projection matrix)?



If we know HOW the camera warps the scene we can map the point from camera coordinates to its projection x_i in the image frame.

lens distortion

If we know position and orientation of the camera frame with respect to the world coordinate frame then we can write an expression that takes us from the point P in the world coordinate frame to its projection x_c in camera coordinates

Start from a point in world coordinate frame $x_w = \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix}$

Model its transformation to the camera coordinate frame $x_c = \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix}$

3D to 3D transformations

Once we have x_c we can apply perspective projection to end up with x_i which is 2D $x_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix}$

$x_w = \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix}$ — world coordinates

coordinate transformation

$x_c = \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix}$ — camera coordinates

perspective projection

$x_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix}$ — image coordinates

Perspective projection

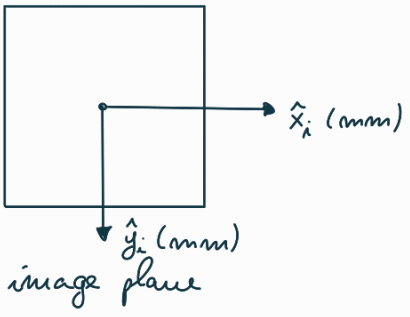
$x_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix}$ ← perspective projection — $x_c = \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix}$ ← coordinate transformation — $x_w = \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix}$

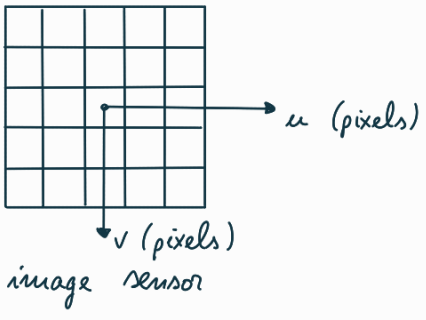
image coordinates | camera coordinates | world coordinates

projection matrix

We know that $\frac{x_i}{f} = \frac{x_c}{z_c}$ and $\frac{y_i}{f} = \frac{y_c}{z_c}$

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





We need to find a way to map from mm to pixel

If m_x and m_y are the pixel densities (pixel/mm) in x and y directions, respectively, then pixel coordinates are:

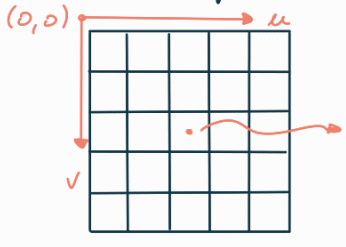
$$u = m_x x_i = m_x f \frac{x_c}{z_c}$$
$$v = m_y y_i = m_y f \frac{y_c}{z_c}$$
} pixels

pixels may be rectangular

Furthermore, the origin of the image is shifted on the upper left corner, so we need to translate

$$u = m_x x_i = m_x f \frac{x_c}{z_c} + o_x$$
$$v = m_y y_i = m_y f \frac{y_c}{z_c} + o_y$$

unknowns



(o_x, o_y) is the principal point, where the optical axis pierces the sensor

$$f_x = m_x f$$
$$f_y = m_y f$$
} (f_x, f_y) are the focal lengths in pixels in the x and y directions.

Cameras typically have only one focal lenght, this is only a way to accomodate for differents pixel densities

(f_x, f_y, o_x, o_y) are the intrinsic parameters of the camera. They represent the camera's internal geometry.

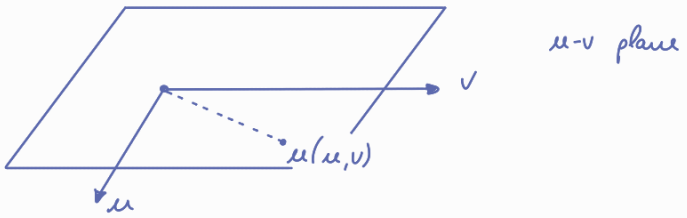
$$\left.\begin{aligned} u &= f_x \frac{x_c}{z_c} + o_x \\ v &= f_y \frac{y_c}{z_c} + o_y \end{aligned}\right\}$$ is a non-linear model; we want a linear model. It is convenient to express the perspective projection equations as linear equations.

We can use homogeneous coordinates. A homogeneous representation of a 2D point $u = (u, v)$ is a 3D point $\tilde{u} = (\tilde{u}, \tilde{v}, \tilde{w})$. The third component $\tilde{w} \neq 0$ is fictitious such that:

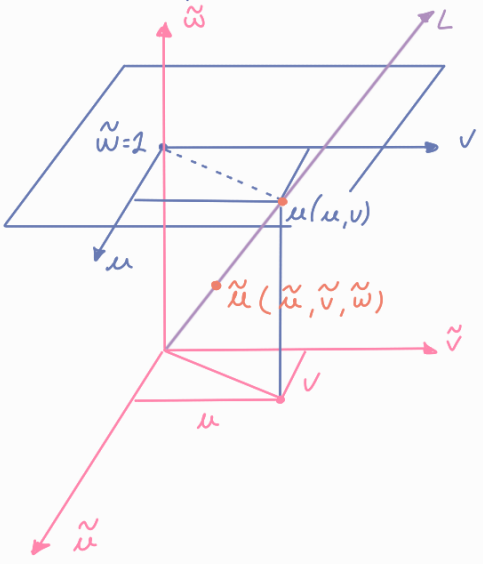
$$u = \frac{\tilde{u}}{\tilde{w}} \qquad v = \frac{\tilde{v}}{\tilde{w}}$$

$$u = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \tilde{w}\,u \\ \tilde{w}\,v \\ \tilde{w} \end{bmatrix} = \begin{bmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{bmatrix} = \tilde{u}$$

this can be any number



we can define another coordinate frame $(\tilde{u}, \tilde{v}, \tilde{w})$ such that the u-v plane lies on $\tilde{w} = 1$



All the points from the origin (not including it) through the point u-v in L are homogeneous coordinates of (u,v)

The same thing holds for points $\in \mathbb{R}^3 \longrightarrow$ homogeneous coordinates in $\mathbb{R}^4$.

We can express the perspective projection equations with homogeneous coordinates

$$\begin{vmatrix} u \\ v \\ 1 \end{vmatrix} = \begin{vmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{vmatrix} = \begin{vmatrix} \tilde{w}\,u \\ \tilde{w}\,v \\ \tilde{w} \end{vmatrix} = \begin{vmatrix} z_c\,u \\ z_c\,v \\ z_c \end{vmatrix}$$

$\tilde{w} = z_c$

Express u, v in homogeneous coordinates

This way:

$$\begin{vmatrix} z_c\,u \\ z_c\,v \\ z_c \end{vmatrix} = \begin{vmatrix} z_c(f_x\,x_c/z_c + o_x) \\ z_c(f_y\,y_c/z_c + o_y) \\ z_c \end{vmatrix} = \begin{vmatrix} f_x\,x_c + z_c\,o_x \\ f_y\,y_c + z_c\,o_y \\ z_c \end{vmatrix}$$

The reason we do this is that we can further simplify the computation by using a matrix:

$$\begin{vmatrix} f_x\,x_c + z_c\,o_x \\ f_y\,y_c + z_c\,o_y \\ z_c \end{vmatrix} = \begin{Vmatrix} f_x & 0 & 0 & 0 \\ 0 & f_y & 0 & 0 \\ 0 & 0 & 1 & 0 \end{Vmatrix} \begin{Vmatrix} x_c \\ y_c \\ z_c \\ 1 \end{Vmatrix}$$

camera calibration matrix

We now have a linear model for perspective projection. This matrix is called intrinsic matrix and it contains all the internal parameters. This is used to map 3D camera points into 2D image points.

$$\tilde{u} = [K\,|\,0]\,\tilde{x}_c = M_{int}\,\tilde{x}_c$$

$\tilde{u}$: point in image frame (pixel coordinates)

$\tilde{x}_c$: point in camera frame

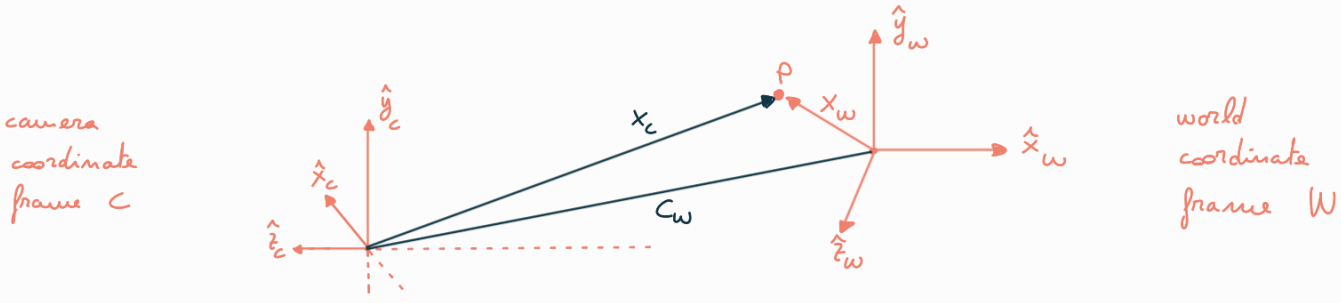
Coordinate transformation

$x_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix}$ ← M_{int} (perspective projection) — $x_c = \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix}$ ← (coordinate transformation) — $x_w = \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix}$

image coordinates | camera coordinates | world coordinates

To map a point from world coordinates to camera coordinates (3D to 3D) we need position (c_w) and orientation (R) of the camera in the world coordinate frame w. These are known as the camera's extrinsic parameters.

c_w is a vector that gives us the position
R is a matrix that gives us the orientation



$$R = \begin{vmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{vmatrix}$$

Row 1: direction of $\hat{x}_c$ in world coordinate frame
Row 2: direction of $\hat{y}_c$ in world coordinate frame
Row 3: direction of $\hat{z}_c$ in world coordinate frame

Orientation / Rotation matrix R is orthonormal

Two vectors u, v are orthonormal iff:
① $\text{dot}(u,v) = u^T v = 0$ orthonormality
② $u^T u = v^T v = 1$ unit length

A square matrix R whose row (or column) vectors are orthonormal is a matrix such that:

① $R^{-1} = R^T$
② $R^T R = R R^T = I$

Given the extrinsic parameters (R, C_w) of the camera, the camera-centric location of the point P in the world coordinate frame is:

$$X_c = R(X_w - C_w) \longrightarrow \text{point } P \text{ in camera coordinate frame}$$

$$X_c = RX_w - RC_w = RX_w + t \qquad t = -RC_w$$

$$X_c = \begin{vmatrix} x_c \\ y_c \\ z_c \end{vmatrix} = \begin{vmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{vmatrix} \begin{vmatrix} x_w \\ y_w \\ z_w \end{vmatrix} + \begin{vmatrix} t_x \\ t_y \\ t_z \end{vmatrix}$$

point in camera coordinate frame — point in world coordinate frame

Putting all together (with homogeneous coordinates)

$$\tilde{X}_c = \begin{vmatrix} x_c \\ y_c \\ z_c \\ 1 \end{vmatrix} = \begin{vmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{vmatrix} \begin{vmatrix} x_w \\ y_w \\ z_w \\ 1 \end{vmatrix} = \begin{vmatrix} R_{3\times3} & t \\ 0_{1\times3} & 1 \end{vmatrix} = M_{ext}$$

We have a single 4×4 matrix that includes rotation and translation. This matrix is called extrinsic matrix.

$$\tilde{X}_c = M_{ext}\, \tilde{X}_w$$

$x_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix}$ ← M_{int} (perspective projection) — $x_c = \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix}$ ← M_{ext} (coordinate transformation) — $x_w = \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix}$

image coordinates — camera coordinates — world coordinates

camera to pixel

$$\begin{vmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{vmatrix} = \begin{vmatrix} f_x & 0 & 0 & 0 \\ 0 & f_y & 0 & 0 \\ 0 & 0 & 1 & 0 \end{vmatrix} \begin{vmatrix} x_c \\ y_c \\ z_c \\ 1 \end{vmatrix}$$

$$\tilde{u} = M_{int}\, \tilde{X}_c$$

world to camera

$$\begin{vmatrix} x_c \\ y_c \\ z_c \\ 1 \end{vmatrix} = \begin{vmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{vmatrix} \begin{vmatrix} x_w \\ y_w \\ z_w \\ 1 \end{vmatrix}$$

$$\tilde{X}_c = M_{ext}\, \tilde{X}_w$$

Combining the above two equations, we get the full projection matrix P:

$$\tilde{u} = M_{int}\, M_{ext}\, \tilde{X}_w = P\, \tilde{X}_w$$

P is a 3×4 matrix called projection matrix. To calibrate the camera we need to find these 12 numbers

$$\begin{vmatrix} u \\ v \\ 1 \end{vmatrix} = \begin{vmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{vmatrix} \begin{vmatrix} x_w \\ y_w \\ z_w \\ 1 \end{vmatrix}$$

Scale of projection matrix

Projection matrix acts on homogeneous coordinates
We know that:

$$\begin{vmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{vmatrix} = k \begin{vmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{vmatrix} \qquad (k \neq 0 \text{ is any constant})$$

Thus:

$$\begin{vmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{vmatrix} \begin{vmatrix} x_w \\ y_w \\ z_w \\ 1 \end{vmatrix} = k \begin{vmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{vmatrix} \begin{vmatrix} x_w \\ y_w \\ z_w \\ 1 \end{vmatrix}$$

P and kP produce the same homogeneous pixel coordinates. The projection matrix is only defined up to a scale factor. This means that if we scale simultaneously the scene and the camera, the resulting image is the same. We can set k arbitrarily.

How can we estimate the projection matrix?

We do this by using an object of known geometry.

1. Capture an image of an object with known geometry
2. Identify correspondances between 3D scene points and image points
   We end up with a set of correspondance points (3D to 2D)
3. For each corresponding point i in scene and image

$$\begin{vmatrix} u^{(i)} \\ v^{(i)} \\ 1 \end{vmatrix} = \begin{vmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{vmatrix} \begin{vmatrix} x_w^{(i)} \\ y_w^{(i)} \\ z_w^{(i)} \\ 1 \end{vmatrix}$$

known unknown known

We have two linear methods to compute P

- $Ax = b$
- $Ax = 0$

Homogeneous system $AP = 0$

Expanding the matrix as linear equations:

$$u^{(i)} = \frac{p_{11} x_w^{(i)} + p_{12} y_w^{(i)} + p_{13} z_w^{(i)} + p_{14}}{p_{31} x_w^{(i)} + p_{32} y_w^{(i)} + p_{33} z_w^{(i)} + p_{34}}$$

$$u^{(i)}(p_{31} x_w^{(i)} + p_{32} y_w^{(i)} + p_{33} z_w^{(i)} + p_{34}) = p_{11} x_w^{(i)} + p_{12} y_w^{(i)} + p_{13} z_w^{(i)} + p_{14}$$

$$u^{(i)} p_{34} = p_{11} x_w^{(i)} + p_{12} y_w^{(i)} + p_{13} z_w^{(i)} + p_{14} - u^{(i)} p_{31} x_w^{(i)} - u^{(i)} p_{32} y_w^{(i)} - u^{(i)} p_{33} z_w^{(i)}$$

since homogeneous coordinates are defined up to a scale, we could set $k = p_{34}$, thus $u^{(i)} p_{34} = u^{(i)}$

$$v^{(i)} = \frac{p_{21} x_w^{(i)} + p_{22} y_w^{(i)} + p_{23} z_w^{(i)} + p_{24}}{p_{31} x_w^{(i)} + p_{32} y_w^{(i)} + p_{33} z_w^{(i)} + p_{34}}$$

$$u^{(i)} p_{34} = p_{11} x_w^{(i)} + p_{12} y_w^{(i)} + p_{13} z_w^{(i)} + p_{14} - u^{(i)} p_{31} x_w^{(i)} - u^{(i)} p_{32} y_w^{(i)} - u^{(i)} p_{33} z_w^{(i)}$$

$$v^{(i)} p_{34} = p_{21} x_w^{(i)} + p_{22} y_w^{(i)} + p_{23} z_w^{(i)} + p_{24} - v^{(i)} p_{31} x_w^{(i)} - v^{(i)} p_{32} y_w^{(i)} - v^{(i)} p_{33} z_w^{(i)}$$

$$u^{(i)} = p_{11} x_w^{(i)} + p_{12} y_w^{(i)} + p_{13} z_w^{(i)} + p_{14} + 0 + 0 + 0 + 0 - u^{(i)} p_{31} x_w^{(i)} - u^{(i)} p_{32} y_w^{(i)} - u^{(i)} p_{33} z_w^{(i)}$$

$$v^{(i)} = 0 + 0 + 0 + 0 \; p_{21} x_w^{(i)} + p_{22} y_w^{(i)} + p_{23} z_w^{(i)} + p_{24} - v^{(i)} p_{31} x_w^{(i)} - v^{(i)} p_{32} y_w^{(i)} - v^{(i)} p_{33} z_w^{(i)}$$

$$u^{(i)} = \begin{pmatrix} x_w^{(i)} & y_w^{(i)} & z_w^{(i)} & 1 & 0 & 0 & 0 & 0 & -u^{(i)} x_w^{(i)} & -u^{(i)} y_w^{(i)} & -u^{(i)} z_w^{(i)} \end{pmatrix} \begin{pmatrix} p \end{pmatrix}^T$$

$$v^{(i)} = \begin{pmatrix} 0 & 0 & 0 & 0 & x_w^{(i)} & y_w^{(i)} & z_w & 1 & -v^{(i)} x_w^{(i)} & -v^{(i)} y_w^{(i)} & -v^{(i)} z_w^{(i)} \end{pmatrix} \begin{pmatrix} p \end{pmatrix}^T$$

$$0 = \begin{pmatrix} x_w^{(i)} & y_w^{(i)} & z_w^{(i)} & 1 & 0 & 0 & 0 & 0 & -u^{(i)} x_w^{(i)} & -u^{(i)} y_w^{(i)} & -u^{(i)} z_w^{(i)} & -u^{(i)} \end{pmatrix} \begin{pmatrix} p \end{pmatrix}^T$$

$$0 = \begin{pmatrix} 0 & 0 & 0 & 0 & x_w^{(i)} & y_w^{(i)} & z_w & 1 & -v^{(i)} x_w^{(i)} & -v^{(i)} y_w^{(i)} & -v^{(i)} z_w^{(i)} & -v^{(i)} \end{pmatrix} \begin{pmatrix} p \end{pmatrix}^T$$

We have a pair of equations for each 3D-2D correspondance. We can put all of these equations together in a matrix in the unknown values p_{11} to p_{34} of the projection matrix P

$$\begin{vmatrix} x_w^{(1)} & y_w^{(1)} & z_w^{(1)} & 1 & 0 & 0 & 0 & 0 & -u_1 x_w^{(1)} & -u_1 y_w^{(1)} & -u_1 z_w^{(1)} & -u_1 \\ 0 & 0 & 0 & 0 & x_w^{(1)} & y_w^{(1)} & z_w^{(1)} & 1 & -v_1 x_w^{(1)} & -v_1 y_w^{(1)} & -v_1 z_w^{(1)} & -v_1 \\ \vdots & & & & & & & & & & & \\ x_w^{(i)} & y_w^{(i)} & z_w^{(i)} & 1 & 0 & 0 & 0 & 0 & -u_i x_w^{(i)} & -u_i y_w^{(i)} & -u_i z_w^{(i)} & -u_i \\ 0 & 0 & 0 & 0 & x_w^{(i)} & y_w^{(i)} & z_w^{(i)} & 1 & -v_i x_w^{(i)} & -v_i y_w^{(i)} & -v_i z_w^{(i)} & -v_i \\ \vdots & & & & & & & & & & & \\ x_w^{(m)} & y_w^{(m)} & z_w^{(m)} & 1 & 0 & 0 & 0 & 0 & -u_m x_w^{(m)} & -u_m y_w^{(m)} & -u_m z_w^{(m)} & -u_m \\ 0 & 0 & 0 & 0 & x_w^{(m)} & y_w^{(m)} & z_w^{(m)} & 1 & -v_m x_w^{(m)} & -v_m y_w^{(m)} & -v_m z_w^{(m)} & -v_m \end{vmatrix} \begin{vmatrix} p_{11} \\ p_{12} \\ p_{13} \\ p_{14} \\ p_{21} \\ p_{22} \\ p_{23} \\ p_{24} \\ p_{31} \\ p_{32} \\ p_{33} \\ p_{34} \end{vmatrix} = \begin{vmatrix} 0 \\ \vdots \\ 0 \end{vmatrix}$$

A — Known

P — unknown

We have an homogeneous system $Ap = 0$
Apply constrained optimization

we won't have $Ap = 0$ thus we want to go as close as possible to this

if we don't set constraints, the result will be $p = 0$

Define a cost function:

$$E = |Ap|^2$$

and a constraint; $\|p\|^2 = 1$ is convenient; the constraint prevents p from becoming a zero vector.
We want Ap as close as possible to 0 and $\|p\|^2 = 1$:

$$\min_p \|Ap\|^2 \text{ such that } \|p\|^2 = 1$$

We can solve this with a Lagrange multiplier $\lambda > 0$

constraint

$$E(p, \lambda) = \|Ap\|^2 - \lambda(\|p\|^2 - 1)$$

$$= (Ap)^T(Ap) - \lambda(p^T p - 1)$$

$$\hat{p} = \arg\min_p E(p, \lambda)$$

$$= \arg\min_p (Ap)^T(Ap) - \lambda(p^T p - 1)$$

$$\frac{\partial E}{\partial p} = 0 \rightarrow 2A^T Ap - 2\lambda p = 0$$

$$A^T Ap = \lambda p$$

eigenvalue problem
eigenvector p with the smallest eigenvalue λ of matrix $A^T A$ minimizes the loss function $L(p)$

considerations

$$p^T A^T A p = \lambda p^T p$$

pre-multiplying both sides by p^T

$$(Ap)^T(Ap) = \lambda$$

$$\|Ap\|^2 = \lambda$$

this is the same expression of the cost function. This means that minimizing $\|Ap\|^2$ is equal to minimizing λ.

Now that we have p (eigenvector with the smallest eigenvalue) we can rearrange it to form the projection matrix.

Non-homogeneous system $AP = b$

Setting $P_{34} = 1$, the i-th point x_i is written as:

$$u^{(i)} = \begin{pmatrix} x_w^{(i)} & y_w^{(i)} & z_w^{(i)} & 1 & 0 & 0 & 0 & 0 & -u^{(i)}x_w^{(i)} & -u^{(i)}y_w^{(i)} & -u^{(i)}z_w^{(i)} \end{pmatrix} \begin{pmatrix} P \end{pmatrix}^T$$

$$v^{(i)} = \begin{pmatrix} 0 & 0 & 0 & 0 & x_w^{(i)} & y_w^{(i)} & z_w & 1 & -v^{(i)}x_w^{(i)} & -v^{(i)}y_w^{(i)} & -v^{(i)}z_w^{(i)} \end{pmatrix} \begin{pmatrix} P \end{pmatrix}^T$$

and putting together all the points we get:

$$\begin{vmatrix} x_w^{(1)} & y_w^{(1)} & z_w^{(1)} & 1 & 0 & 0 & 0 & 0 & -u_1 x_w^{(1)} & -u_1 y_w^{(1)} & -u_1 z_w^{(1)} \\ 0 & 0 & 0 & 0 & x_w^{(1)} & y_w^{(1)} & z_w^{(1)} & 1 & -v_1 x_w^{(1)} & -v_1 y_w^{(1)} & -v_1 z_w^{(1)} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_w^{(i)} & y_w^{(i)} & z_w^{(i)} & 1 & 0 & 0 & 0 & 0 & -u_i x_w^{(i)} & -u_i y_w^{(i)} & -u_i z_w^{(i)} \\ 0 & 0 & 0 & 0 & x_w^{(i)} & y_w^{(i)} & z_w^{(i)} & 1 & -v_i x_w^{(i)} & -v_i y_w^{(i)} & -v_i z_w^{(i)} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_w^{(m)} & y_w^{(m)} & z_w^{(m)} & 1 & 0 & 0 & 0 & 0 & -u_m x_w^{(m)} & -u_m y_w^{(m)} & -u_m z_w^{(m)} \\ 0 & 0 & 0 & 0 & x_w^{(m)} & y_w^{(m)} & z_w^{(m)} & 1 & -v_m x_w^{(m)} & -v_m y_w^{(m)} & -v_m z_w^{(m)} \end{vmatrix} \begin{vmatrix} P_{11} \\ P_{12} \\ P_{13} \\ P_{14} \\ P_{21} \\ P_{22} \\ P_{23} \\ P_{24} \\ P_{31} \\ P_{32} \\ P_{33} \end{vmatrix} = \begin{vmatrix} -u_1 \\ -v_1 \\ \vdots \\ -u_i \\ -v_i \\ \vdots \\ -u_m \\ -v_m \end{vmatrix}$$

A — Known — $(2m \times 11)$

P — unknown — (11×1)

b — Known — $(2m \times 1)$

Considering a cost function $E = \| AP - b \|^2$, the projection matrix is obtained as follows:

$$\hat{P} = \underset{P}{\operatorname{argmin}}\, E = \underset{P}{\operatorname{argmin}}\, \| AP - b \|^2 = \underset{P}{\operatorname{argmin}}\, (AP-b)^T(AP-b)$$

$\frac{\partial E}{\partial P} = 0$

$$\rightarrow A^T(AP - b) = 0$$

$$A^TAP - A^Tb = 0$$

$$A^TAP = A^Tb$$

$$P = (A^TA)^{-1}A^Tb$$

P can be estimated if A^TA is invertible

A square matrix A is invertible iff it has a non-zero determinant. When a matrix is invertible, it exists an inverse matrix A^{-1} such that

$$A A^{-1} = A^{-1} A = I$$

If the matrix A is not invertible ($\det(A) \neq 0$) then we solve $AP = 0$

Decomposing the camera matrix

$$P = \begin{bmatrix} \boxed{\begin{matrix} p_{11} & p_{12} & p_{13} \\ p_{21} & p_{22} & p_{23} \\ p_{31} & p_{32} & p_{33} \end{matrix}} & \begin{matrix} p_{14} \\ p_{24} \\ p_{34} \end{matrix} \end{bmatrix} = \begin{bmatrix} f_x & 0 & 0 & 0 \\ 0 & f_y & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

M_{int} M_{ext}

$$\begin{bmatrix} p_{11} & p_{12} & p_{13} \\ p_{21} & p_{22} & p_{23} \\ p_{31} & p_{32} & p_{33} \end{bmatrix} = \begin{bmatrix} f_x & 0 & 0 \\ 0 & f_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} = KR$$

K → calibration matrix

Given that K is an Upper Right Triangular matrix and R is an Orthonormal matrix it is possible to uniquely decouple K and R from their product with a process called QR factorization.
In other words, knowing P we can estimate all the internal parameters (K) and the rotation matrix (R).

To find the translation:

$$\begin{bmatrix} p_{14} \\ p_{24} \\ p_{34} \end{bmatrix} = \begin{bmatrix} f_x & 0 & 0 \\ 0 & f_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} = Kt$$

$$t = K^{-1} \begin{bmatrix} p_{14} \\ p_{24} \\ p_{34} \end{bmatrix}$$

To find the camera center c:

$$Pc = 0$$

Single Value Decomposition of P