Computer Vision

James L. Crowley

M2R MoSIG — Fall Semester
Lesson 6 — 19 Nov 2020

Homogeneous Coordinates and Projective Camera Models

<u>Lesson Outline</u>:

1 Homogeneous Coordinates and Tensor Notation

Homogeneous coordinates allow us to express translation, rotation, scaling, and projection all as matrix multiplications. The principle is to add an extra dimension to each vector.

For example, points on a plane are expressed as:

$$\vec{P} = \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Similarly, points in 3D space become

$$\vec{Q} = \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

The line equation, ax+by+c=0 can be expressed as a simple product:

$$\vec{L}^T \vec{P} = \begin{pmatrix} a & b & c \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = 0 \qquad \text{where} \qquad \vec{\mathrm{L}}^{\mathrm{T}} = \begin{pmatrix} a & b & c \end{pmatrix}$$

Similarly, for a plane equation: *ax+by+cz+d=1:*

$$\vec{S}^T \vec{P} = \begin{pmatrix} a & b & c & d \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = 0 \qquad \text{where} \qquad \vec{\mathrm{S}} = \begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix}$$

This is called a "homogeneous" equation because the all terms are first order.
Technically this is a "first order" homogeneous equation.

$ax^2+by^2+c=0$ would be a second order homogeneous equation.

Note that in Homogeneous coordinates, all scalar multiplications are equivalent.

$$a \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = b \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Any vector can be expressed in "canonical" form by normalizing the last coefficient to 1.

$$\begin{pmatrix} ax \\ ay \\ a \end{pmatrix} = \begin{pmatrix} ax/a \\ ay/a \\ a/a \end{pmatrix} = \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Using Homogeneous coordinates, we will construct a camera model as a 3 x 4 matrix

$$M_s^i = \begin{pmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & 1 \end{pmatrix}$$

such that the image point (x_i, y_i) is found from the scene point (x_s, y_s, z_s) by

$$\begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix} = \begin{pmatrix} q_1/q_3 \\ q_2/q_3 \\ 1 \end{pmatrix} = \begin{pmatrix} q_1 \\ q_2 \\ q_3 \end{pmatrix} = \vec{Q} = M_s^i \vec{P} = \begin{pmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & 1 \end{pmatrix} \begin{pmatrix} x_s \\ y_s \\ z_s \\ 1 \end{pmatrix}$$

Note that M_s^i is in canonical form. All coefficients have been divided by m_{34}.

$$M_s^i = \begin{pmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & 1 \end{pmatrix}$$

The camera model, M_s^i, has 11 coefficients and not 12. Calibrating M_s^i requires estimating 11 parameters.

1.1 Tensor Notation:

In tensor notation, the sign " $\rightarrow$ " is replaced by subscripts and superscripts.
A super-script signifies a column vector. For example the point $\vec{P}$ is P^i

$$P^i = \begin{pmatrix} p_1 \\ p_2 \\ p_3 \end{pmatrix}$$

The line $\vec{L}^T$ is $L_i = (l_1, l_2, l_3)$

A matrix is a line vector of column vector (or a column vector of line vectors)

$$M_i^j = \begin{pmatrix} m_1^1 & m_2^1 & m_3^1 \\ m_1^2 & m_2^2 & m_3^2 \\ m_1^3 & m_2^3 & m_3^3 \end{pmatrix} \qquad M_i = \begin{pmatrix} m_1 & m_2 & m_3 \end{pmatrix} \qquad M^j = \begin{pmatrix} m^1 \\ m^2 \\ m^3 \end{pmatrix}$$

When homogeneous coordinates are used to represent transforms, these indices can be used to indicate the reference frame.

For example: A transformation from the scene "s" to the image "i" is a 3 x 4 matrix:

$$M_s^i = \begin{pmatrix} m_1^1 & m_2^1 & m_3^1 & m_4^1 \\ m_1^2 & m_2^2 & m_3^2 & m_4^2 \\ m_1^3 & m_2^3 & m_3^3 & 1 \end{pmatrix}$$

The sub/super scripts indicate the source and destination reference frames.
M_s^i is a transformation from "s" (Scene) to "i" (image).

The summation symbol is implicit when a superscript and subscript have the same letter. This is called Einstein summation convention. For example:

$$L_i P^i = l_1 p^1 + l_2 p^2 + l_3 p^3$$

The product of a matrix and a vector gives a vector:

$$p^j = T_i^j p^i$$

This example transforms the point p^i in reference frame i to a point p^j in reference j.

2 Coordinate Transforms in 2D

Homogeneous coordinates allow us to unify projective transformations using only **matrix multiplication**. This includes both Affine and Projective transformations
Four popular classes of transformations are:

- Euclidean Transformations
- Isometries
- Affine Transformations
- Projective Transformations

Homogeneous transformations allow us to unify all of these as matrix multiplications. Below we review transformations in 2-space. These transformations are easily generalized to higher number of dimensions.

2D: points and lines on a plane
3D: points and planes in a volume
4D and up: points and hyper-planes in a hyper-space

2.1 Euclidean Transformation

A Euclidean transformation expresses 3 degrees of freedom: translation (t_x, t_y,) and rotation, θ.

$$Q^B = T_A^B P^A$$

$$\begin{pmatrix} q^1 \\ q^2 \\ q^3 \end{pmatrix} = \begin{pmatrix} Cos(\theta) & -Sin(\theta) & t_x \\ Sin(\theta) & Cos(\theta) & t_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} p^1 \\ p^2 \\ p^3 \end{pmatrix}$$

Note, in such transforms, the origin of the A (source) reference frame is mapped to the position (t_x, t_y) in the B (target) coordinate system.

Note that in classic notation this would be written:

$$\begin{pmatrix} x_B \\ y_B \\ 1 \end{pmatrix} = \begin{pmatrix} wx_B \\ wy_B \\ w \end{pmatrix} = \begin{pmatrix} Cos(\theta) & -Sin(\theta) & t_x \\ Sin(\theta) & Cos(\theta) & t_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_A \\ y_A \\ 1 \end{pmatrix}$$

which expresses:

$$x_B = x_A Cos(\theta) - y_A Sin(\theta) + t_x$$
$$y_B = x_A Sin(\theta) + y_A Cos(\theta) + t_y$$
$$w = 1$$

The transformation is invertible: $T_A^B = (T_B^A)^{-1}$

That is the translation (t_x, t_y) is the position of the origin of the source in the target.

2.2 Similitude

A similitude expresses 4 degrees of freedom: translation (t_x, t_y,), rotation, θ and scale, s.

We can translate, rotate and rescale an image with $Q^B = S_A^B P^A$

$$\begin{pmatrix} q^1 \\ q^2 \\ q^3 \end{pmatrix} = \begin{pmatrix} s \cdot Cos(\theta) & -s \cdot Sin(\theta) & t_x \\ s \cdot Sin(\theta) & s \cdot Cos(\theta) & t_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} p^1 \\ p^2 \\ p^3 \end{pmatrix}$$

this gives a reduction of scale by s: $x_2 = s\, x_1$ (x_2 is reduced by a scale factor s)
Alternatively we could write

$$\begin{pmatrix} q^1 \\ q^2 \\ q^3 \end{pmatrix} = \begin{pmatrix} Cos(\theta) & -Sin(\theta) & t_x \\ Sin(\theta) & Cos(\theta) & t_y \\ 0 & 0 & 1/s \end{pmatrix} \begin{pmatrix} p^1 \\ p^2 \\ p^3 \end{pmatrix}$$

in classic notation :

$$\begin{pmatrix} x_2 \\ y_2 \\ 1 \end{pmatrix} = \begin{pmatrix} s \cdot Cos(\theta) & -s \cdot Sin(\theta) & t_x \\ s \cdot Sin(\theta) & s \cdot Cos(\theta) & t_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \\ 1 \end{pmatrix}$$

In any case this reduces to a Euclidean transformation because all scalar multiples of homogeneous coordinates are equivalent.

If we replace *s* by *1/s* the space B is a larger scale copy of A.
If *s* is negative we obtain a reflection.

When s is the same for *x* and y, the scaling is isotropic (uniform). It is possible to express anisotropic (non-uniform) scaling with separate scale factors for x and y.

$$\begin{pmatrix} x_2 \\ y_2 \\ 1 \end{pmatrix} = \begin{pmatrix} s_x \cdot Cos(\theta) & -s_y \cdot Sin(\theta) & t_x \\ s_x \cdot Sin(\theta) & s_y \cdot Cos(\theta) & t_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \\ 1 \end{pmatrix}$$

An anisotropic projection captures 5 degrees of freedom: transalation (t_x, t_y,), rotation, θ and scale, (s_x, s_y).

2.3 Affine Transformations

An affine transformation has 6 dof: *a, b, c, d, e, f*

The complete affine transformation is $Q^b = A_a^b P^a$

$$\begin{pmatrix} q^1 \\ q^2 \\ q^3 \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} p^1 \\ p^2 \\ p^3 \end{pmatrix}$$

or

$$\begin{pmatrix} x_2 \\ y_2 \\ 1 \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \\ 1 \end{pmatrix}$$

The affine transform includes similitudes and isometries as a special cases, but also includes sheer.

2.4 Projection between two planes (Homography)

The projective transformation from one plane to another is called a homography.
A homography is bijective (reversible).

In tensor notation $Q^B = H_A^B P^A$

$$\begin{pmatrix} q^1 \\ q^2 \\ q^3 \end{pmatrix} = \begin{pmatrix} h_1^1 & h_2^1 & h_3^1 \\ h_1^2 & h_2^2 & h_3^2 \\ h_1^3 & h_2^3 & h_3^3 \end{pmatrix} \begin{pmatrix} p^1 \\ p^2 \\ p^3 \end{pmatrix}$$

$$x_B = \frac{q^1}{q^3} = \frac{h_1^1 p^1 + h_2^1 p^2 + h_3^1 p^3}{h_1^3 p^1 + h_2^3 p^2 + h_3^3 p^3} \qquad y_B = \frac{q^2}{q^3} = \frac{h_1^2 p^1 + h_2^2 p^2 + h_3^2 p^3}{h_1^3 p^1 + h_2^3 p^2 + h_3^3 p^3}$$

In classic notation:

$$\begin{pmatrix} x_b \\ y_b \\ 1 \end{pmatrix} = \begin{pmatrix} wx_b \\ wy_b \\ w \end{pmatrix} = \begin{pmatrix} h_1^1 & h_2^1 & h_3^1 \\ h_1^2 & h_2^1 & h_3^1 \\ h_1^3 & h_2^3 & 1 \end{pmatrix} \begin{pmatrix} x_a \\ y_a \\ 1 \end{pmatrix}$$

$$x_B = \frac{wx_B}{w} = \frac{h_{11}x_A + h_{12}y_A + h_{13}}{h_{31}x_A + h_{32}y_A + h_{33}}$$

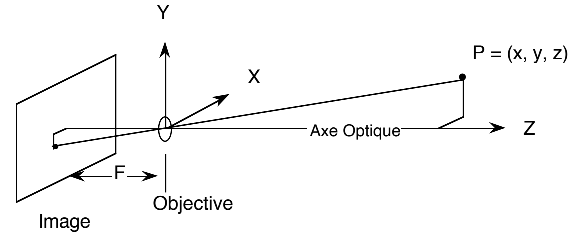
$$y_B = \frac{wy_B}{w} = \frac{h_{21}x_A + h_{22}y_A + h_{23}}{h_{31}x_A + h_{32}y_A + h_{33}}$$

3 The Camera Model

A "camera" is a closed box with an aperture (in latin: "camera obscura"). Photons are reflected from the world, and pass through the aperture to form an image on the retina. Thus the camera coordinate system is defined with the aperture at the origin.

The Z (or depth) axis runs perpendicular from the retina through the aperture.
The X and Y axes define coordinates on the plane of the aperture.

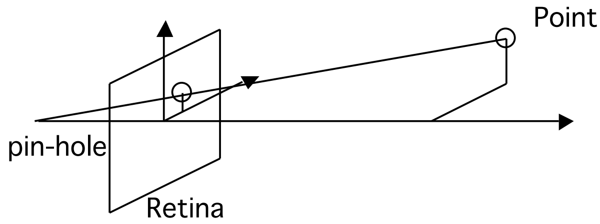
3.1 The Pinhole Camera



Points in the scene are projected to an "up-side down" image on the retina.

This is the "Pin-hole model" for the camera.

The scientific community of computer vision often uses the "Central Projection Model". In the Central Projection Model, the retina is placed in Front of the projective point.



We will model the camera as a projective transformation from scene coordinates, S, to image coordinates, i.

$$\vec{Q}^i = M_s^i \vec{P}^s$$

This transformation is expressed as a 3x4 matrix:

$$M_s^i = \begin{pmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \end{pmatrix}$$

composed from 3 transformations between 4 reference frames.

3.2 Extrinsic and Intrinsic camera parameters

The camera model can be expressed as a function of 11 parameters.
These are often separated into 6 "extrinsic" parameters and 5 "intrinsic" parameters:

Thus the "extrinsic" parameters of the camera describe the camera position and orientation in the scene. These are the six parameters:

Extrinsic Parameters = $(x, y, z, \theta, \varphi, \gamma)$

The intrinsic camera parameters express the projection to the retina, and the mapping to the image. These are :

F : The "focal" length of the camera
C_x, C_y : the image center (expressed in pixels).
D_x, D_y : The size of pixels (expressed in pixels/mm).

3.3 Coordinate Systems

This transformation can be decomposed into 3 basic transformations between 4 reference frames. The reference frames are:

Coordinate Systems:

Scene Coordinates:

Points in the scene: $P^s = (x_s, y_s, z_s, 1)^T$

Camera Coordinates:

Point in the scene: $P^c = (x_c, y_c, z_c, 1)^T$

Project on the retina: $Q^r = (x_r, y_r, 1)^T$

Image Coordaintes

Point in the image: $Q^i = (i, j, 1)^T$

The transformations are represented by Homogeneous projective transformations.

$$\vec{P}^c = T_s^c \, \vec{P}^s \qquad \vec{Q}^r = P_c^r \vec{P}^c \qquad \vec{Q}^i = C_r^i \vec{Q}^r$$

These express

1) A translation/rotation from scene to camera coordinates:
2) A projection from scene points in camera coordinates to the retina:
3) Sampling scan and A/D conversion of the retina to give an image:

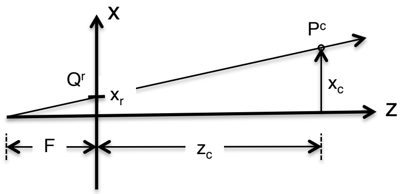
When expressed in homogeneous coordinates, these transformations are composed as matrix multiplications.

$$\vec{Q} = M_s^i \vec{P} = C_r^i P_c^r T_s^c \vec{P}$$

We use "tensor notation" to keep track of our reference frames.

3.4 Projective Transforms: from the scene to the retina

Consider the central projection model for a 1D camera:



The origin of this system is the retina, shown here as intersection of the x (1-D retina axis) and the z (depth) axes. These are referred to as "camera Coordinates".

The point $P^c = (x_c, z_c, 1)^T$ is a point in the scene, in camera coordinates.
The point $Q^r = (x_r, 1)^T$ is the projection of this point on the retina.
The projective (focal) point is a distance *F* behind the retina.

By similar triangles: $$\frac{x_r}{F} = \frac{x_c}{(F+z_c)} \Rightarrow x_r \frac{(F+z_c)}{F} = x_c$$

We can express the fraction $\frac{(F+z_c)}{F}$ as a coefficient w.

In matrix form this can be written as: $$\begin{pmatrix} x_r \\ 1 \end{pmatrix} = \begin{pmatrix} wx_r \\ w \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{F} & 1 \end{pmatrix} \begin{pmatrix} x_c \\ z_c \\ 1 \end{pmatrix}$$

If we extend to 3D we obtain: $$\begin{pmatrix} x_r \\ y_r \\ 1 \end{pmatrix} = \begin{pmatrix} wx_r \\ wy_r \\ w \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{1}{F} & 1 \end{pmatrix} \begin{pmatrix} x_c \\ y_c \\ z_c \\ 1 \end{pmatrix}$$

We represent this as the projective transformation P_c^r

$$Q^r = \begin{pmatrix} q_1 \\ q_2 \\ q_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{1}{F} & 1 \end{pmatrix} \begin{pmatrix} p^1 \\ p^2 \\ p^3 \\ 1 \end{pmatrix} = P_c^r \vec{P}^c$$

3.5 From Scene to Camera

The following matrix represents a translation $\Delta x, \Delta y, \Delta z$ and a rotation R.

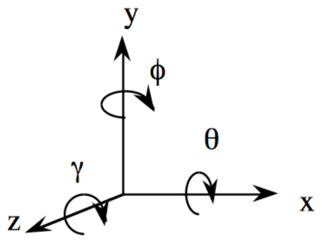
$$T_s^c = \begin{pmatrix} & R & & \Delta x \\ & & & \Delta y \\ & & & \Delta z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

The transformation is composed by expressing the position of the source reference frame in the destination reference frame.

The rotation part is a 3x3 matrix that can be decomposed into 3 smaller rotations using Euler Angles (rotation around each axis).

$$\mathbf{R} = \mathbf{R}_z(\gamma)\mathbf{R}_y(\varphi)\mathbf{R}_x(\theta)$$

En 3D



$\mathbf{R}_x(\theta)$ is a rotation around the x axis.

$$R_x(\theta) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{pmatrix}$$

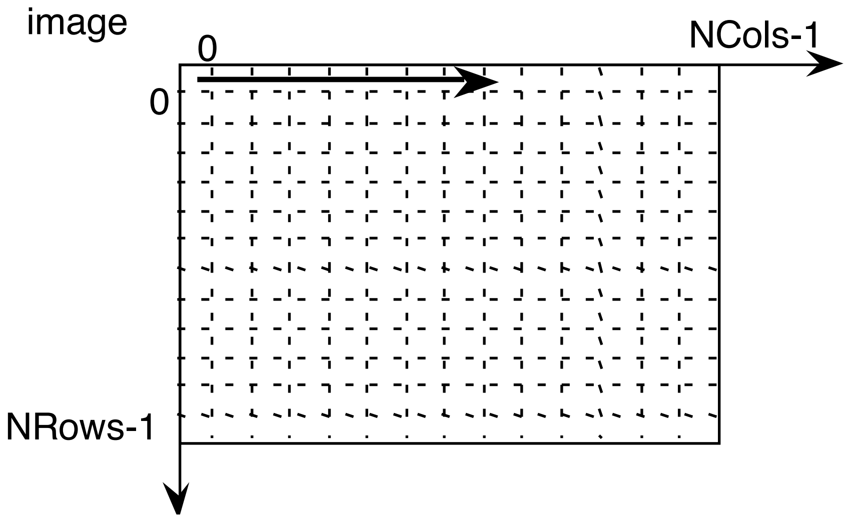
$$R_y(\varphi) = \begin{pmatrix} \cos(\varphi) & 0 & \sin(\varphi) \\ 0 & 1 & 0 \\ -\sin(\varphi) & 0 & \cos(\varphi) \end{pmatrix}$$

$$R_z(\gamma) = \begin{pmatrix} \cos(\gamma) & -\sin(\gamma) & 0 \\ \sin(\gamma) & \cos(\gamma) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

3.6 From the Retina to Digitized Image

The "intrinsic parameters of the camera are F and C_x, C_y, D_x, D_y

The image frame is composed of pixels (picture elements)



Note that pixels are not necessarily square.

The mapping from retina to image can be expressed with 4 parameters:

C_x, C_y : the image center (expressed in cols and rows).
D_x, D_y : The size of pixels expressed in cols/m and rows/mm.

$i = x_r D_i$ (mm · col/mm) + C_i (cols)

$j = y_r D_j$ (mm · row/mm) + C_j (rows)

Transformation from retina to image :

$$Q^i = \mathbf{C}_r^i \; Q^r$$

$$\begin{pmatrix} i \\ j \\ 1 \end{pmatrix} = \begin{pmatrix} D_i & 0 & C_i \\ 0 & -D_j & C_j \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_r \\ y_r \\ 1 \end{pmatrix}$$

3.7 The Complete Camera Model

$$\boxed{P^i = \boldsymbol{C}_r^i \ \boldsymbol{P}_c^r \ \boldsymbol{T}_s^c \ P^s = \boldsymbol{M}_s^i \ P^s}$$

$$Q^i = M_s^i P^s$$

$$\begin{pmatrix} wi \\ wj \\ w \end{pmatrix} = \begin{pmatrix} m_1^1 & m_2^1 & m_3^1 & m_4^1 \\ m_1^2 & m_2^2 & m_3^2 & m_4^2 \\ m_1^3 & m_2^3 & m_3^3 & m_4^3 \end{pmatrix} \begin{pmatrix} x_s \\ y_s \\ z_s \\ 1 \end{pmatrix}$$

and thus

$$i = \frac{wi}{w} = \frac{M_s^1 \cdot R^s}{M_s^3 \cdot R^s} = \frac{M_{11}x_s + M_{12}y_s + M_{13}z_s + M_{14}}{M_{31}x_s + M_{32}y_s + M_{33}z_s + 1}$$

$$j = \frac{wj}{w} = \frac{M_s^2 \cdot R^s}{M_s^3 \cdot R^s} = \frac{M_{21}x_s + M_{22}y_s + M_{23}z_s + M_{24}}{M_{31}x_s + M_{32}y_s + M_{33}z_s + 1}$$

3.8 Calibrating the Camera

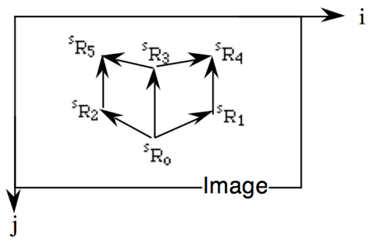
How can we obtain M_s^i ? By a process of calibration.

Observe a set of at least 6 non-coplanar points whose position in the world is known.

R_k^s for *k=0,1,2,3,4,5* (s are the scene coordinate axes s=1,2,3)

For each point, k, we observe the corresponding point in the image P_k^i

For example, we can use the corners of a cube. Define the lower front corner as the origin, and the edges as unit distances.



The matrix M_s^i is composed of 3x4=12 coefficients. However because, M_s^i is in homogeneous coordinates, the coordinate m_{34} can be set to 1.

Thus there are 12-1 = 11.

We can determine these coefficients by observing known points in the scene. (R_k^s).

Each point provides two coefficients. Thus, for 11 coefficients we need at least $5\frac{1}{2}$ points. With 6 points the system is over-constrained.

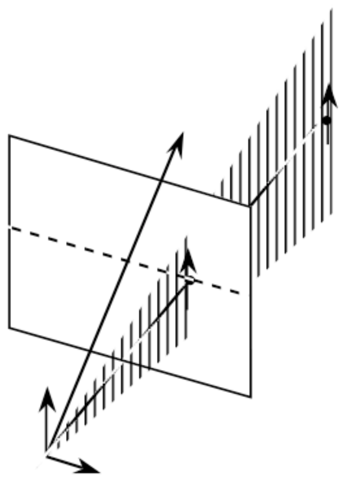
For each known calibration point R_k^s given its observed image position P_k^i, we can write:

$$i_k = \frac{w_k i_k}{w_k} = \frac{M_s^1 \cdot R_k^s}{M_s^3 \cdot R_k^s} \qquad j_k = \frac{w_k j_k}{w_k} = \frac{M_s^2 \cdot R_k^s}{M_s^3 \cdot R_k^s}$$

This gives 2 equations for each point.

$$M_s^1 \cdot R_k^s - i_k(M_s^3 \cdot R_k^s) = 0 \qquad M_s^2 \cdot R_k^s - j_k(M_s^3 \cdot R_k^s) = 0$$

Each pair of equations corresponds to the planes that pass though the image row and the image column of the observed image point P^s_k



The equation $M_s^1 \cdot R_k^s - i_k(M_s^3 \cdot R_k^s) = 0$ is the vertical plane that includes the projective center through the pixel $i=i_k$.

The equation $M_s^2 \cdot R_k^s - j_k(M_s^3 \cdot R_k^s) = 0$ is the horizontal plane that includes the projective center and the row $j=j_k$.

For each of k scene points, we know R_k^s by definition and we observe P_k^i.
We then use these pairs to solve for M_s^i.

given $P_k^i = \begin{pmatrix} i_k \\ j_k \\ 1 \end{pmatrix} = \begin{pmatrix} w_k i_k \\ w_k j_k \\ w_k \end{pmatrix}$ we write : $P_k^i = M_s^i R_k^s$

to obtain $M_s^1 \cdot R_k^s - i_k(M_s^3 \cdot R_k^s) = 0$ and $M_s^2 \cdot R_k^s - j_k(M_s^3 \cdot R_k^s) = 0$

For each pair of corresponding points (P_k^i, R_k^s) can write two equations

$$M_1^1 \cdot R_k^1 + M_2^1 \cdot R_k^2 + M_3^1 \cdot R_k^3 + M_4^1 \cdot 1 - i_k M_1^3 \cdot R_k^1 - i_k M_2^3 \cdot R_k^2 - i_k M_3^3 \cdot R_k^3 - i_k = 0$$
$$M_1^2 \cdot R_k^1 + M_2^2 \cdot R_k^2 + M_3^2 \cdot R_k^3 + M_4^2 \cdot 1 - j_k M_1^3 \cdot R_k^1 - j_k M_2^3 \cdot R_k^2 - j_k M_3^3 \cdot R_k^3 - j_k = 0$$

any 11 such equations we can solve for M_s^i (neglecting the coefficient $M_4^3 = 1$)

With 6 pairs of scene and image points we have 11 possible sets of 11 equations yielding 11 solutions. We could "average" the results.

Alternatively, we can set up all 12 equations and solve for a least squares solution that minimizes :

$$\mathbf{C} = \| \mathbf{A}\, \mathbf{M}_s^i \|$$

in matrix form this gives: $\mathbf{A_k}\, \mathbf{M}_s^i = 0$.

$$\begin{pmatrix} R_k^1 & R_k^2 & R_k^3 & 1 & 0 & 0 & 0 & 0 & -i_k R_k^1 & -i_k R_k^2 & -i_k R_k^3 & -i_k \\ 0 & 0 & 0 & 0 & R_k^1 & R_k^2 & R_k^3 & 1 & -j_k R_k^1 & -j_k R_0^2 & -j_0 R_k^3 & -j_k \end{pmatrix} \begin{pmatrix} M_1^1 \\ M_2^1 \\ M_3^1 \\ M_4^1 \\ M_1^2 \\ M_2^2 \\ M_3^2 \\ M_4^2 \\ M_1^3 \\ M_2^3 \\ M_3^3 \\ 1 \end{pmatrix} = 0$$

For example, give a cube with observed corners

$P^L_o = (101, 221)$ $P^L_1 = (144, 181)$ $P^L_2 = (22, 196)$
$P^L_3 = (105, 88)$ $P^L_4 = (145, 59)$ $P^L_5 = (23, 67)$

Least squares will give:

$$M_S^i = \begin{pmatrix} 55.88 & -79.29 & 1.27 & 101.91 \\ -22.29 & -17.87 & -134.34 & 221.30 \\ 0.100 & 0.038 & -0.008 & 1 \end{pmatrix}$$

Note that the center of the retina is at pixel (102, 221).