Early Fundamentals of Coordinate Changes and Rotation Matrices for 3D Computer Vision

Ricardo Fabbri
Benjamin B. Kimia
Brown University, Division of Engineering, Providence RI 02912, USA
Based the first author's notes from 2007

March 21, 2012

1 Basics

A *coordinate system* for a vector space V is given by a basis of vectors plus an origin. Consider a base $A = \{a_1, a_2, \ldots, a_n\}$ of V. It forms a coordinate system if we attach origins O_A to it. Note that a_i and O_A are both elements of V.

The coordinates of a vector $v \in V$ relative to a base A is denoted by $\mathcal{X}_A(v)$. It is the unique set of scalars $x_1, \ldots, x_n$ such that $v = x_1 a_1 + \cdots + x_n a_n$. Remember: a vector space consists of the so-called vectors plus a scalar field K, taken here as the reals $\mathbb{R}$. Therefore the coordinates are given by an isomorphism between the vectors of a space of dimension n to the space of n-tuples of scalars (coordinates), K^n. If no change of coordinates is performed in an application, we may abuse notation and use the vector itself to denote its coordinates, i.e. we write v as a short form of $\mathcal{X}_A(v)$.

If we denote by 1 the coordinate system $\{A, O_A\}$, then the coordinate of a vector $v \in V$ in the coordinate system 1 is defined by:

$$\mathcal{X}_1(v) := \mathcal{X}_A(v - O_A) \tag{1.1}$$

Definition 1. Given a linear map $L : V \to V$ and two bases $A = \{a_1, a_2, \ldots, a_n\}$ and $B = \{b_1, b_2, \ldots, b_n\}$, the matrix of L relative to bases A and B is the unique matrix $\mathcal{M}_B^A(L)$ such that:

$$\mathcal{X}_B(L(v)) = \mathcal{M}_B^A(L) \cdot \mathcal{X}_A(v) \tag{1.2}$$

for any vector v in V.

Theorem 1. *The matrix $\mathcal{M}_B^A(L)$ is given by:*

$$\mathcal{M}_B^A(L) = \begin{bmatrix} M_B^A(L)(\mathcal{X}_A(a_1)) & M_B^A(L)(\mathcal{X}_A(a_2)) & \cdots & M_B^A(L)(\mathcal{X}_A(a_n)) \end{bmatrix} \tag{1.3}$$

That is, we first write each transformed basis vector a_i of A in terms of base B, then arrange them as columns of the matrix.

Proof. You can see this promptly since $\mathcal{X}_A(a_i)$ is given by $(0, \ldots, 1, \ldots, 0)$, where 1 is in the i^{th} entry. If you apply the matrix to this, the result is the i^{th} column of the matrix. □

Definition 2. (*Matrix of change of basis*). Consider two bases A and B for a vector space V. The matrix of change of basis from basis A to basis B is the matrix of the identy map id with respect to bases A and B, $\mathcal{M}_B^A(id)$. More pragmatically, it is no more than the unique matrix that takes coordinate vectors $\mathcal{X}_A(v)$ to coordinate vectors $\mathcal{X}_B(v)$ via multiplication: $\mathcal{X}_B(v) = \mathcal{M}_B^A(id)\mathcal{X}_A(v)$.

Note that, although the identity transformation id is such that $id(v) = v$ for v in V, the matrix of this map with respect to different bases A and B is *not* an identy map in the space of coordinates K^n.

$$\mathcal{M}_B^A(id) : \mathcal{K}^n(\text{coordinates}) \to \mathcal{K}^n(\text{coordinates}) \tag{1.4}$$

$$\mathcal{X}_A(v) \mapsto M_B^A(id)\mathcal{X}_A(v) = X_B(v) \tag{1.5}$$

1.1 Rotations

Motivation: We wish to find a practical way of defining the matrix of change of basis, given two bases differing by a rotation.

We all know in an elementary way what is an angle, and what does it mean to rotate a point counter-clockwise. We know by the same elementary way that rotation is a linear transformation: The rotation of a scaled vector is the rotated vector times the scaling, and the rotation of a sum of vectors is the sum of the rotated vectors. We can treat these as principles, and derive the matrix of the rotation from them. (The very basic principles would be that rotations do not change angles nor scale).

Since the rotation is a linear transformation (intuitively), the rotation of any vector can be determined once we know how to rotate the basis vectors. This is very useful in practice.

Problem 1.1. Given a set of 2D points, rotate them counter-clockwise about the origin.

Solution. First, denote the the coordinate sytem as having canonical basis E and origin O. We will rotate by an angle θ measuring from e_1 counterclockwise. To rotate the object, we just need to know how to rotate a point and then apply this to all the defining points of the object.

By drawing a diagram and by elemetary trigonometry we find that the rotation of e_1 is $(\cos\theta, \sin\theta)$, and the rotation of e_2 is $(-\sin\theta, \cos\theta)$. Therefore the matrix of rotation (as a linear map) is given by taking these vectors as the columns:

$$R_\theta = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

In my opinion this is much easier, intuitive, and simple to derive the rotation matrix than the traditional way (for those who know linear algebra). Note that

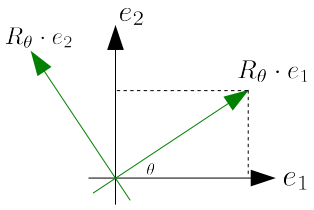

Figure 1: Deriving a 2D rotation matrix from basis vectors.

there is no rotation of the coordinate system, but just rotation of the points of the plane while keeping the coordinate system fixed.

Theorem 2. *The* columns *of a rotation matrix are the rotation of the basis vectors* $e_1, \ldots, e_n$. *The* rows *of the rotation matrix are the unit vectors that are taken to the basis vectors* $e_1, \ldots, e_n$ after *the rotation.*

One problem we face when defining a rotation matrix in 3D is merely how to define the direction of rotation for positive angles (*positive rotations*), and for negative angles (*negative rotations*).

Definition 3. (*orientation*) A 3D coordinate system is *right-handed* when positive rotations are such that, when looking from a positive axis toward the origin, a 90^o *counterclockwise* rotation will transform one positive axis into the other. In other words, in a right-handed coordinate system the positive rotations are given by the right-hand rule. (see Figure in P. 214, Foley). This table follows from this convention:

If axis of rotation is	Direction of positive rotation is
x	$y \to z$
y	$z \to x$
z	$x \to y$

Similarly, in a left-handed the positive rotations are *clockwise* when looking from a positive axis toward the origin. Whenever we are the ones modeling a problem and defining the coordinate systems, we try to choose every coordinate system to be right-handed. This way there is no reflexion involved in addition to the rotations.

1.2 Change of coordinates

Problem 1.2. (*Change of coordinates*) Given two coordinates $1 : \{A, O_A\}$ and $2 : \{B, O_B\}$ for V, determine the change of coordinates transformation from the first to the second. In other words, we want to find a matrix M and vector T such that:

$$\mathcal{X}_2(p) = M\mathcal{X}_1(p) + T \tag{1.6}$$

for every p in V.

Solution. Given a point p, from (1.1) we have:

$$\mathcal{X}_2(p) = \mathcal{X}_B(p - O_B) \tag{1.7}$$
$$= \mathcal{M}_B^A(id)\mathcal{X}_A(p - O_B) = \mathcal{M}_B^A(id)\mathcal{X}_A(p - O_A + O_A - O_B) \tag{1.8}$$
$$= \mathcal{M}_B^A(id)\mathcal{X}_A(p - O_A) - \mathcal{M}_B^A(id)\mathcal{X}_A(O_B - O_A) \tag{1.9}$$
$$= \mathcal{M}_B^A(id)\mathcal{X}_1(p) - \mathcal{M}_B^A(id)\mathcal{X}_1(O_B) \tag{1.10}$$

thus,

$$M = \mathcal{M}_B^A(id) \tag{1.11}$$
$$T = -M\mathcal{X}_1(O_B) \tag{1.12}$$

For example, the transformation from world coordinates to camera coordinates assumed in the definition of a projection matrix is defined in the same way. In this case, $M_B^A(id)$ will be a rotation matrix (since both bases have same units, both are orthogonal, and have the same orientation).

This change of coordinate system transformation is not linear anymore. Such a change of coordinates can be made linear by use of projective coordinates. In such case, we just calculate M and T as above and then assemble them into the same homogeneous matrix. This makes it easier to compose many coordinate transforms. If M is equal to a rotation R, the 3D transformation in homogeneous coordinates is:

$$\begin{bmatrix} R & T \\ 0^{\mathrm{T}} & 1 \end{bmatrix}$$

In terms of roll, pitch, and yaw angles, as in Figure 2, the rotation matrix $R = \mathcal{M}_B^A(id)$ can be decomposed as $R(\theta, \phi, \psi) = R_z(\theta)R_y(\phi)R_x(\psi)$, where

$$R_z(\theta) = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1.13}$$

$$R_y(\phi) = \begin{bmatrix} \cos\phi & 0 & -\sin\phi \\ 0 & 1 & 0 \\ \sin\phi & 0 & \cos\phi \end{bmatrix} \tag{1.14}$$

$$R_x(\psi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\psi & \sin\psi \\ 0 & -\sin\psi & \cos\psi \end{bmatrix} \tag{1.15}$$

And the form of $R(\theta, \phi, \psi) = R_z(\theta)R_y(\phi)R_x(\psi)$ would be:

$$\begin{bmatrix} \cos\theta\cos\phi & \cos\theta\sin\phi\sin\psi + \sin\theta\cos\psi & -\cos\theta\sin\phi\cos\psi + \sin\theta\sin\psi \\ -\sin\theta\cos\phi & -\sin\theta\sin\phi\sin\psi + \cos\theta\cos\psi & \sin\theta\sin\phi\cos\psi + \cos\theta\sin\psi \\ \sin\phi & -\cos\phi\sin\psi & \cos\phi\cos\psi \end{bmatrix} \tag{1.16}$$

Note that this is the inverse of the rotation for points while fixing the bases. Note also that the order of multiplication of the x, y, z rotation matrices makes a difference in the final rotation.

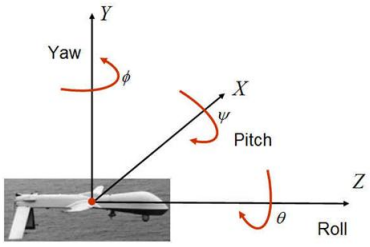

Figure 2: Orthogonal rotations

2 Perspective projection

2.1 Introduction

A camera is modeled as a device that maps a point in world coordinates to a point in image coordinates. In the case of a pinhole model, this is a composition of three coordinate changes:

1. 3D–3D rigid transformation (world to camera coordinates)
2. 3D–2D projection (camera to normalized image coordinates)
3. 2D–2D transformation (normalized image coordinates to final image coordinates).

Traditionally, the transformation from world homogeneous coordinates to image homogeneous coordinates are given by a 3×4 projection matrix P, in the case a of a pinhole camera model. This matrix can be decomposed as:

$$P = K\,[I\,|\,0] \begin{bmatrix} R & T \\ 0^{\mathrm{T}} & 1 \end{bmatrix} = K[R\,|\,T]\,, \tag{2.1}$$

Where:

- The matrix $\begin{bmatrix} R & T \\ 0^{\mathrm{T}} & 1 \end{bmatrix}$ (comprising the *extrinsic parameters*) transforms from 3D world coordinates to so-called 3D camera coordinates
- The matrix $[I\,|\,0]$ performs the perspective projection from 3D camera coordinates to 2D normalized image plane coordinates.
- The matrix K (*calibration matrix*, comprising the *intrinsic parameters*) transforms from normalized image plane coordinates to the (final) image coordinates.

Let us go through each of these in detail. The world coordinates are given by any fixed 3D coordinate system and is denoted by $W_c = \{O, W\}$, with $W = \{e_1, e_2, e_3\}$. A point in this system has coordinates denoted by (x^w, y^w, z^w).

The image coordinate system is defined as the pixel coordinates (u, v) centered in the bottom-left corner of the image, where u is the horizontal axis, and v is the vertical axis pointing upwards. As explained in Section **??**, we could use v as pointing down with little change in the formulation. Add Zisserman's notes

We think of a camera i as having its own orthonormal coordinate system $\{\boldsymbol{c}, B\}$, $B = \{b_1, b_2, b_3\}$, the coordinates indicated by x, y, z. The camera frame, shown in Figure **??**, is conveniently defined from the following properties:

- Its origin is at the center of the camera, $\boldsymbol{c}$.
- The $x - y$ plane is parallel to the image plane
- The z axis is normal to the image plane
- It is right-handed (assuming right-handed world coordinates will be used).
- Choose z such that it forms a right-handed coordinate system with the directions of u and v.
- The x axis is aligned with the u axis of the final image coordinate by convention.
- The y axis defined such that x, y, z is right-handed.
- Is has the same units as the world coordinates (e.g. milimiters).

We define the *principal point* as the projection of the camera center onto the image plane. The image plane on the camera coordinate system is expressed as (x, y, f), where f is the distance from $\boldsymbol{c}$ to the image plane, called *focal length*. The detector plane is at $(x, y, -f)$. By *normalized image coordinates* we mean the (x, y) coordinates in the $(x, y, z = 1)$ coordinate plane of the camera coordinate system. The normalized image plane is the $x - y$ plane translated to unit distance from the camera center. It is a virtual image having an orthonormal basis, origin at the principal point, and as if the camera had unit focal length.

The disposition of the camera coordinate system with respect to the world accounts for the orthogonal rotations roll, pitch, and yaw, as shown in Figure 2. Another convenient way to interpret these is as a unit focal vector $\boldsymbol{F}$ orthogonal to the image plane ($\boldsymbol{F} = (0, 0, 1)$ in camera coordinates) to represent pitch and yaw of the camera with respect to the world, plus an angle representing the rotation of the image plane around $\boldsymbol{F}$.

So, as an overview, in order to define camera parameters we:

- Start by identifying an image coordinate system and a world coordinate system
- Conveniently define a camera coordinate system as explained
- Find a rotation and translation in order to change from world to this camera coordinate system.

- Project points in camera coordinates to the normalized image plane, obtaining normalized image coordinates.
- Rescale, translate, and shear the normalized coordinates in order to get the final image coordinates.

In order to be able to transform between world and camera coodinates, we need a rotation and a translation term, the so-called extrinsic parameters. We then project that point into the normalized image plane. To get the final image coordinates, we also need to know:

- The focal length f
- The conversion factors s_u and s_v between world units and pixels measured in the u and v directions, resp.
- Position of the principal point relative to the bottom-left corner of the image
- The skew angle θ between u and v, since the u and v may not be orthogonal

It can be seen that, in principle, we need 6 intrinsic parameters. We will see that the focal length f, s_u, and s_v can be combined into parameters, as far as projecting 3D to 2D points goes. So in practice we need 5 intrinsic parameters, giving us a total of 11 parameters for the transformation performed by a pinhole camera. The *aspect ratio* is defined as s_u/s_v.

In the homogeneous notation of (2.1), the intrinsic parameters are enconded in the *calibration matrix* K:

$$K = \begin{bmatrix} \alpha_u & s & u_o \\ 0 & \alpha_v & v_o \\ 0 & 0 & 1 \end{bmatrix} \tag{2.2}$$

TODO: fix this

α_u	focal length $\times$ u pixels / unit length
α_v	focal length $\times$ y pixels / unit length
u_0	u coordinate of principal point
v_0	v coordinate of principal point
s_θ	skew factor

$$\alpha_u = \frac{f}{\text{width of a pixel in world measurement units}} \tag{2.3}$$

$$\alpha_v = \frac{f}{\text{height of a pixel in world measurement units}}. \tag{2.4}$$

Problem 2.1. Given a 3D point $p^w = (x^w, y^w, z^w)^\top$ in world coordinates, what is (u, v) ?

First, transform to camera coordinates $p = (x, y, z)^\top$:

$$p = Rp^w + T \tag{2.5}$$

where R and T are the extrinsic parameters, and are precisely defined in Section 1.2.

Now, project the point into the normalized image plane using similarity of triangles:

$$\begin{cases} \tilde{x} = \dfrac{x}{z} & (2.6) \\ \tilde{y} = \dfrac{y}{z} & (2.7) \end{cases}$$

After projection, to get image coordinates, the first step is to include focal length:

$$\begin{cases} \bar{x} = f\dfrac{x}{z} & (2.8) \\ \bar{y} = f\dfrac{y}{z} & (2.9) \end{cases}$$

Then convert to pixels:

$$\begin{cases} \bar{x}_{pix} = s_u f\dfrac{x}{z} & (2.10) \\ \bar{y}_{pix} = s_v f\dfrac{y}{z} & (2.11) \end{cases}$$

As we can see, f has effect similar to s_u and s_v. It would be sufficient to keep just two parameters, say f and aspect ratio, but lets us stick with three parameters for now. Next, translate to the bottom-left corner by, say, $(t_{\tilde{u}}, t_{\tilde{v}})$:

$$\begin{cases} \tilde{u} = s_u f\dfrac{x}{z} + t_{\tilde{u}} & (2.12) \\ \tilde{y} = s_v f\dfrac{y}{z} + t_{\tilde{v}} & (2.13) \end{cases}$$

And convert into the skewed coordinates:

$$\begin{cases} u = \tilde{u} - \tilde{v}\cot\theta & (2.14) \\ v = \tilde{v}\csc\theta & (2.15) \end{cases}$$

which gives

$$\begin{cases} u = \overbrace{s_u f}^{\alpha_u}\dfrac{x}{z} + \overbrace{(-\cot\theta\, s_v f)}^{s_\theta}\dfrac{y}{z} + \overbrace{t_{\tilde{u}} - \cot\theta\, t_{\tilde{v}}}^{u_0} & (2.16) \\ v = \underbrace{s_v f\csc\theta}_{\alpha_v}\dfrac{y}{z} + \underbrace{\csc\theta\, t_{\tilde{v}}}_{v_0} & (2.17) \end{cases}$$

Thus, the camera to image coordinate transformation can be ultimately given by the five parameters in the calibration matrix K. The above formula provides the interpretation for these parameters.

Problem 2.2. (*Backprojection*) Given (u, v), find the 3D point $\mathbf{\Gamma}^w = (x^w, y^w, z^w)$ in world coordinates, that projects to it.

DOF	Description
3	rotation
3	translation
2	scaling in x, y
1	focal length (may be incorporated as scaling)
2	principal point
1	skew
6	Total extrinsic
6	Total intrinsic
12	Total
11	Total unique (focal length as scaling)

Table 1: Summary of camera parameters

Solution. From 2.16, we can solve for (x, y, z) in the camera's coordinate system:

$$\begin{cases} \frac{x}{z} = \frac{u - u_0}{\alpha_u} - s\frac{v - v_0}{\alpha_u \alpha_v} & (2.18) \\ \frac{y}{z} = \frac{v - v_0}{\alpha_v} & (2.19) \end{cases}$$

So, taking $z = \lambda$, we have:

$$\mathbf{\Gamma}(\lambda) = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \lambda \begin{bmatrix} \frac{u-u_0}{\alpha_u} - s\frac{v-v_0}{\alpha_u \alpha_v} \\ \frac{v-v_0}{\alpha_v} \\ 1 \end{bmatrix} \tag{2.20}$$

We then apply a rotation and translation to the final solution:

$$\mathbf{\Gamma}^w(\lambda) = \begin{bmatrix} x^w \\ y^w \\ z^w \end{bmatrix} = R^\top \mathbf{\Gamma}(\lambda) - R^\top T \tag{2.21}$$

where R, T are the extrinsic parameters of the camera.

2.2 Examples

2.2.1 Turntables

Problem 2.3. Model a turntable camera configuration in detail.

Figure 3 illustrates this. Denote world coordinates as $I = \{W, O\}$, and camera coordinates at angle θ as $II(\theta) = \{B(\theta), \boldsymbol{c}(\theta)\}$. In a typical turntable configuration, after a process of calibration we start by having intrinsics for the first camera and its extrinsics relative to the world coordinates (e.g. centered on a calibration jig). This calibration can be performed using camera resectioning,

i.e. the process of estimating camera parameters from world 3D to image 2D feature correspondences.

In other words:

Data:

- Complete calibration for the first camera ($\theta = 0$):
  - $P(0)$
    - $\mathcal{M}^W_{B_0}(id) = R^W_{B_0}$.
    - $\boldsymbol{T}^I_{II_0}$ such that $\mathcal{X}_{II_0}(p) = R^W_{B_0}\mathcal{X}_I(p) + T^I_{II_0}$.
    - $K(0)$
  - $K(\theta) = K(0)$ for all θ.
- Rotation step $\Delta\theta$ (equally spaced camera positions).

Want:

- $P(\theta)$ (extrinsic part):
  - $\mathcal{M}^W_{B(\theta)}(id) = R^W_{B(\theta)}$.
  - $\boldsymbol{T}^I_{II(\theta)}$ such that $\mathcal{X}_{II(\theta)}(p) = R^W_{B(\theta)}\mathcal{X}_I(p) + T^I_{II(\theta)}$.

We first establish the direction of the basis vectors for each coordinate system.

- Assume given world coordinate system centered at C_R, so $O = C_R$, but its basis is not necessarily aligned with the bases of the cameras. (In practice, make sure this is where the world coordinate system really is).
- First, imagine cameras are looking at C_R.
- Rotation is going in the direction of $b_1(\theta)$, meaning it is the unit tangent vector to the camera path.
- If b_3 is going from $\boldsymbol{c}(\theta)$ to O_R, then b_2 is going into the plane.
- Assume rotation is positive with respect to the cameras' coordinate systems.

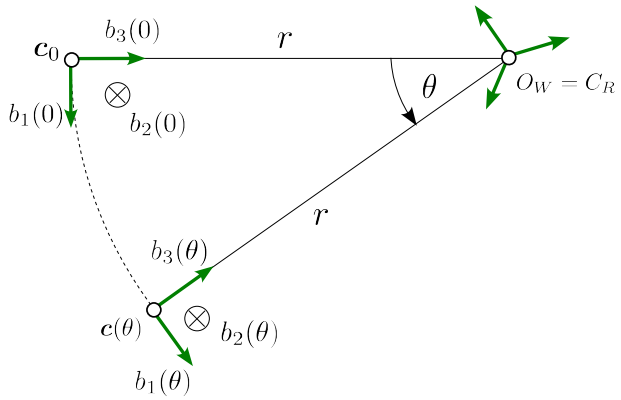

Figure 3: A turntable camera configuration

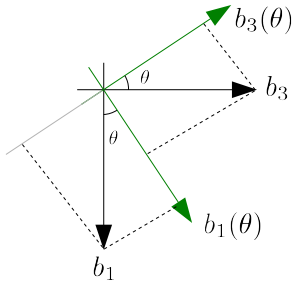

Figure 4: Determining the rotation matrix for the turntable example.

Rotation

$$R^W_{B(\theta)} = R^{B_0}_{B(\theta)} R^W_{B_0} \tag{2.22}$$

so we need to know how to write $R^{B_0}_{B(\theta)}$, i.e. we need to determine $R^{B_0}_{B(\theta)}(\mathcal{X}_{B_0}(b_i)) = \mathcal{X}_{B(\theta)}(b_i)$. From Figure 4, the column of the rotation matrix are:

$$R^{B_0}_{B(\theta)} \mathcal{X}_{B_0}(b_i)$$

That is, we write the vectors b_i (of base B_0) in base $B(\theta)$. The result is:

$$R^{B_0}_{B(\theta)} = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \tag{2.23}$$

Translation We now have to determine $T^I_{II(\theta)}$. From (1.12), we have:

$$T^I_{II(\theta)} = -R^W_{B(\theta)} \mathcal{X}_I(\boldsymbol{c}(\theta)) \tag{2.24}$$

Since the world basis W may not be aligned with the rotation plane, we try to write $\boldsymbol{c}(\theta)$ in the first cameras' coordinate system:

$$\mathcal{X}_I(\boldsymbol{c}(\theta)) = \mathcal{X}_W(\boldsymbol{c}(\theta) - O_W) = R_W^{B_0}\mathcal{X}_{B_0}(\boldsymbol{c}(\theta) - O_W) \tag{2.25}$$

We have all the information at $\theta = 0$, so we need to write the above in terms of that:

$$\mathcal{X}_{B_0}(\boldsymbol{c}(\theta) - O_W) = \mathcal{X}_{B_0}\left[Rot(\theta)(\boldsymbol{c}_0 - O_W)\right] = Rot_{B_0}(\theta)\mathcal{X}_{B_0}(\boldsymbol{c}_0 - O_W) \tag{2.26}$$

where

$$Rot_{B_0}(\theta) = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \tag{2.27}$$

and

$$\mathcal{X}_{B_0}(\boldsymbol{c}_0 - O_W) = R_{B_0}^W\mathcal{X}_W(\boldsymbol{c}_0 - O_W) = R_{B_0}^W\mathcal{X}_I(\boldsymbol{c}_0). \tag{2.28}$$

where $\mathcal{X}_I(\boldsymbol{c}_0)$ is obtained using (2.24):

$$\mathcal{X}_I(\boldsymbol{c}_0) = -\left[R_{B_0}^W\right]^{-1} T_{II_0}^I \tag{2.29}$$

with $\theta = 0$. Summarizing our solution:

$$T_{II(\theta)}^I = R_{B(\theta)}^W R_W^{B_0} Rot_{B_0}(\theta) R_{B_0}^W \left[R_{B_0}^W\right]^{-1} T_{II_0}^I \tag{2.30}$$

$$T_{II(\theta)}^I = R_{B(\theta)}^W R_W^{B_0} Rot_{B_0}(\theta) T_{II_0}^I \tag{2.31}$$

$$T_{II(\theta)}^I = R_{B(\theta)}^{B_0} Rot_{B_0}(\theta) T_{II_0}^I \tag{2.32}$$

while, from (2.24)

$$\mathcal{X}_I(\boldsymbol{c}(\theta)) = -\left[R_{B(\theta)}^W\right]^{-1} T_{II(\theta)}^I \tag{2.33}$$

$$= -\left[R_{B(\theta)}^W\right]^{-1} R_{B(\theta)}^{B_0} Rot_{B_0}(\theta) T_{II_0}^I \tag{2.34}$$

$$= R_W^{B_0} Rot_{B_0}(\theta) T_{II_0}^I \tag{2.35}$$

Problem 2.4. Same as in problem 2.3, but now imposing, as given data, that the rotation plane is the plane $x - z$ (normal is axis y) of the world coordinates, and the cameras' axis y are not necessarily aligned with this. See Figure 5.

Solution. We want:

$$\mathcal{M}_{B(\theta)}^W(id) = R_{B(\theta)}^W \tag{2.36}$$

$$T_{II(\theta)}^I = -R_{B(\theta)}^W\mathcal{X}_I(\boldsymbol{c}(\theta)) \tag{2.37}$$

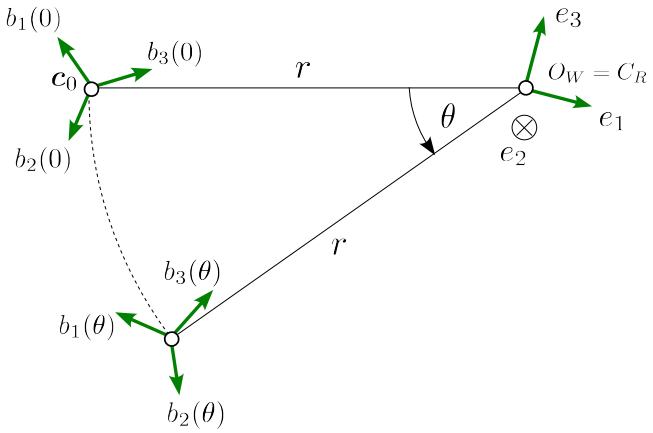

Figure 5

where

$$\mathcal{X}_I(\boldsymbol{c}(\theta)) = Rot_{axis_y}(-\theta)\mathcal{X}_I(\boldsymbol{c}(0)) \tag{2.38}$$

where

$$Rot_{axisy}(-\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \tag{2.39}$$

and

$$\mathcal{X}_I(\boldsymbol{c}(0)) = -\left[R^W_{B_0}\right]^{-1} T^I_{II_0} \tag{2.40}$$

Unlike the previous problem, the tricky part here is to determine the rotation matrix. The rotation is done around axis y of coordinate system I, and *not* necessarily around any of the orthogonal axes of the cameras. So we simply rotate the world coordinate system into a coordinate system $I(\theta)$ such that $M^{W(\theta)}_{B(\theta)}$ equals the given $R^W_{B(\theta)}$. So we first cange from I to $I(\theta)$ (easy) then from $I(\theta)$ to $B(\theta)$ (which is given).

To change from I to $I(\theta)$:

$$\mathcal{M}^I_{I(\theta)} = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \tag{2.41}$$

and the final solution is:

$$R^I_{II(\theta)} = R^{I(\theta)}_{II(\theta)} R^I_{I(\theta)} \tag{2.42}$$

2.2.2 Other examples

Problem 2.5. If the image is cropped after acquisition, change the calibration matrix in order to take that into account.

Problem 2.6. Given a point p in the camera coordinate system, what is the coordinate of the same point in world coordinates? What is the world coordinate of $p - \boldsymbol{c}$?

3 Structure and Motion

3.1 Reconstruction from Correspondences

The main problem to be solved is:

Problem 3.1. (Reconstruction for N views) Given N views and M point-correspondences $x_i^1, \ldots, x_i^N$ across all views, estimate camera intrinsic and extrinsic parameters.

Problem 3.2. (*Metric reconstruction of points from 2 views*) Given two cameras K^1, R^1, T^1, K^2, R^2, T^2, and given corresponding image points $\boldsymbol{\gamma}^1 = (u^1, v^1)$, $\boldsymbol{\gamma}^2 = (u^2, v^2)$, find $\boldsymbol{\Gamma} = (X, Y, Z)$, the 3D point projecting to them.

References

Lang, Linear Algebra.
Zisserman's Book.
Foley and Van Dam.
Prof. Joe Mundy's lecture notes.