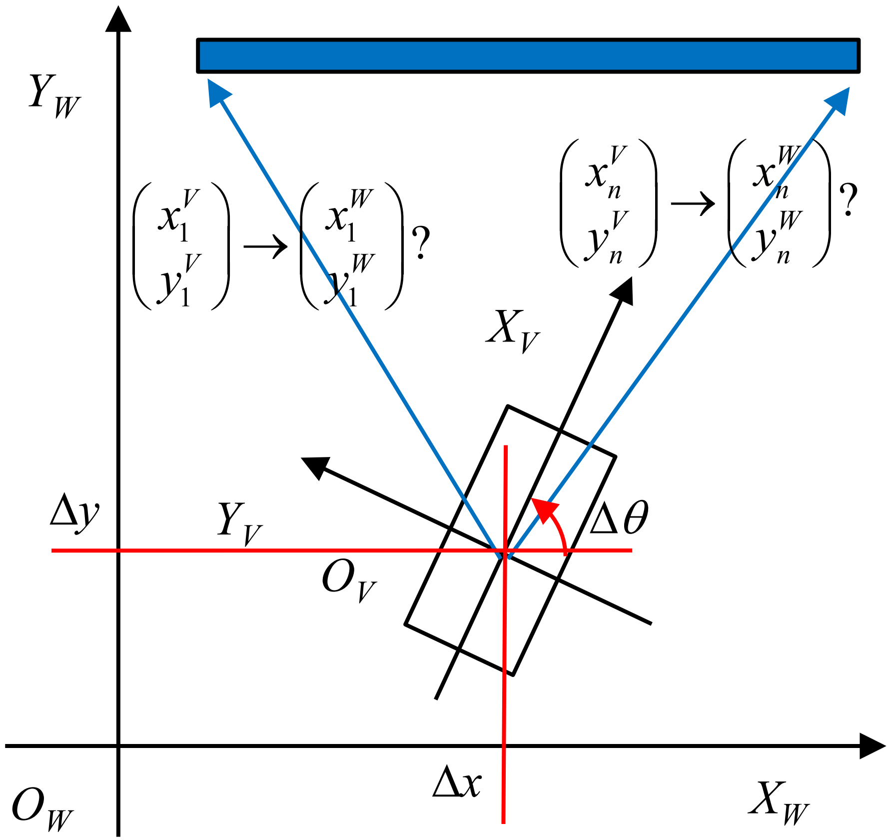
ICT03A: Advanced Robotics
#2 Frame Transformation

Keitaro Naruse
Email: naruse@u-aizu.ac.jp

2D frame transformation

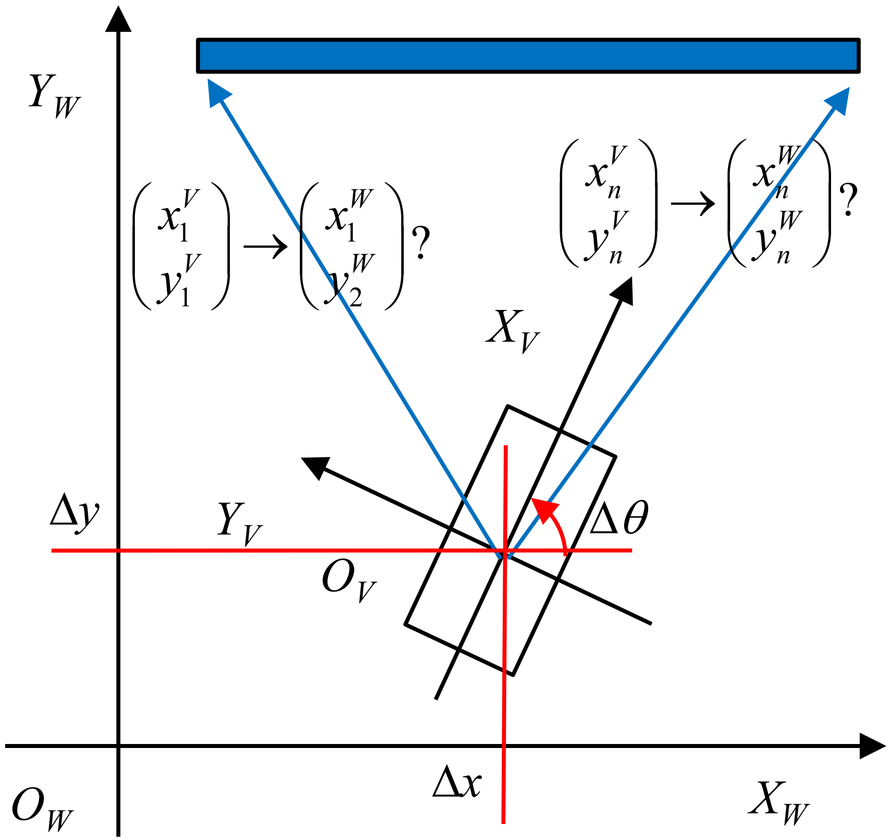
Motivation: Mapping
動機: 地図生成



- Frame := Coordinate system
  フレーム= 座標系
- Suppose we detect and measure an object in a robot frame
- How do we convert it to world frame in a single step?
- This is two-step: rotation（回転） and translation（並進）

$$\begin{pmatrix} x_i^W \\ y_i^W \end{pmatrix} = \begin{pmatrix} \cos(\Delta\theta) & -\sin(\Delta\theta) \\ \sin(\Delta\theta) & \cos(\Delta\theta) \end{pmatrix} \begin{pmatrix} x_i^V \\ y_i^V \end{pmatrix} + \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix}$$

2D Homogeneous Transformation Matrix
二次元同次変換行列



1-step transformation by homogeneous transformation matrix

$$\begin{pmatrix} x_i^W \\ y_i^W \\ \hline 1 \end{pmatrix} = \left(\begin{array}{cc|c} \cos(\Delta\theta) & -\sin(\Delta\theta) & \Delta x \\ \sin(\Delta\theta) & \cos(\Delta\theta) & \Delta y \\ \hline 0 & 0 & 1 \end{array}\right) \begin{pmatrix} x_i^V \\ y_i^V \\ \hline 1 \end{pmatrix}$$

$$\boldsymbol{x}_i^W = T\boldsymbol{x}_i^V$$

It is a special version of Affine transformation

- Only translation and rotation
- No scale and shear

because we consider only a rigid body

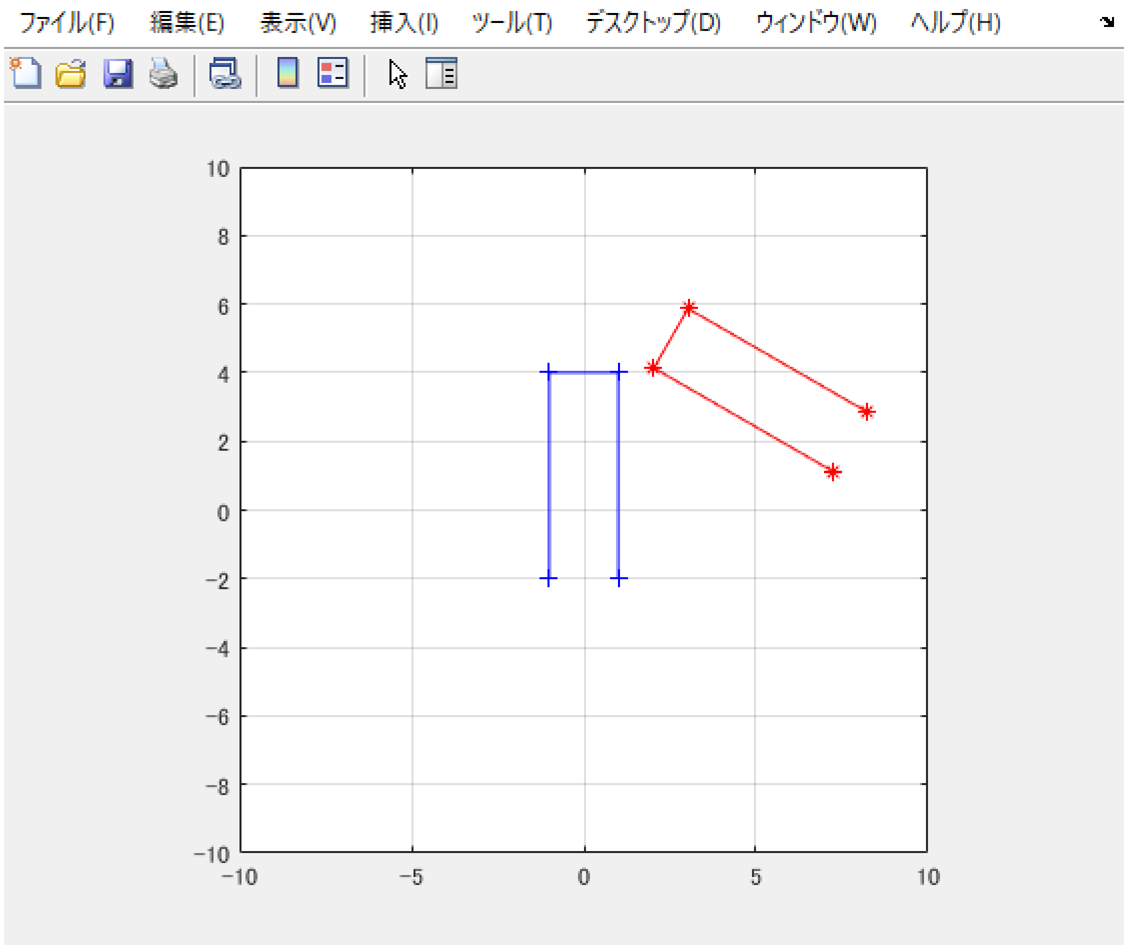
Matlab Sample Code of 2D Homogeneous Trans. Matrix

```
function [T] = T2D(x, y, q)
%T2D returns 2D homogeneous trannsformation matrix
%   from a target frame to a world frame
%   Input
%   - x: x coordinate of target x-origin in a world frame
%   - y: y coordinate of target y-origin in a world frame
%   - q: angle from a world to target frame
%   Output
%   - T: 3*3 matrix
T = [
    cos(q), -sin(q), x;
    sin(q), cos(q), y;
    0, 0, 1
    ];
end
```

Matlab Sample Code of 2D Homogeneous Trans. Matrix

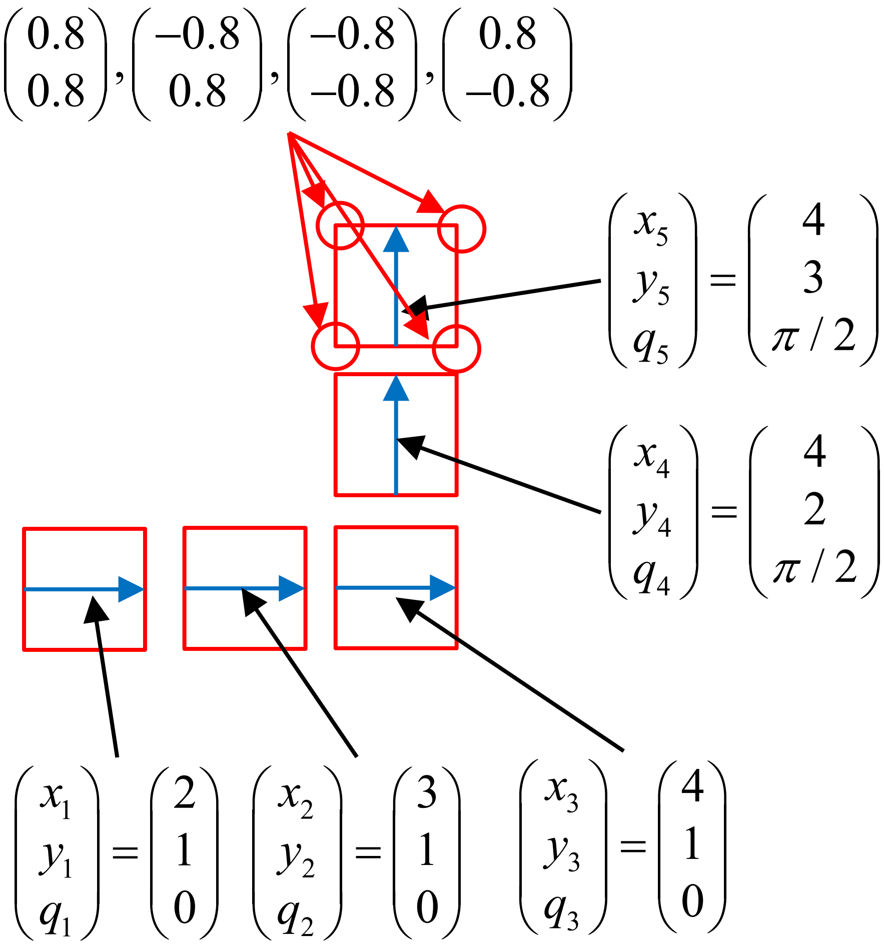
```
% Vehicle frame to world frame
X = 6.0; Y = 3.0; Q = deg2rad(60.0);
T = T2D(X, Y, Q);
% Homegeneous points in vehicle frame
pV = [
      1,  1, -1, -1; % x
     -2,  4,  4, -2; % y
      1,  1,  1,  1  % constant
    ];
% Homegeneous Points in world frame
pW = [ T*pV(:,1), T*pV(:,2), T*pV(:,3), T*pV(:,4)];
% Homegeneous points in vehicle frame
figure(1);
plot(pV(1,:), pV(2,:), 'b+-', pW(1,:), pW(2,:), 'r*-');
xlim([-10 10]); ylim([-10 10]); grid on; pbaspect([1 1 1]);
```

Results



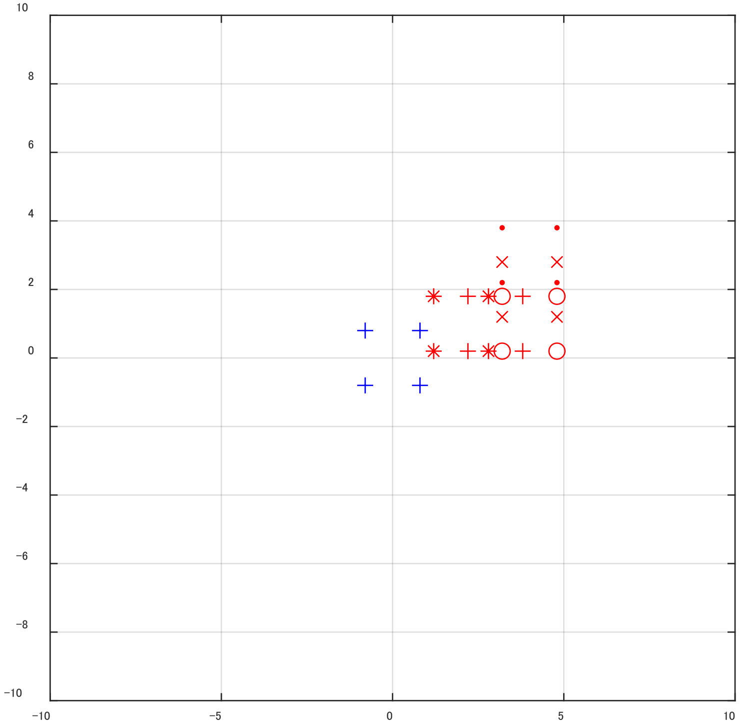
ファイル(F)
編集(E)
表示(V)
挿入(I)
ツール(T)
デスクトップ(D)
ウィンドウ(W)
ヘルプ(H)
10
8
6
4
2
0
−2
−4
−6
−8
−10
−10
−5
0
5
10

Quiz #1: Frame Transformation in 2D



- Suppose a vehicle moves as in the left figure
- At each of the positions, it measures the four points from its local frame
- Make an integrated map of sensed points with homogeneous transformation matrix

Example of Result

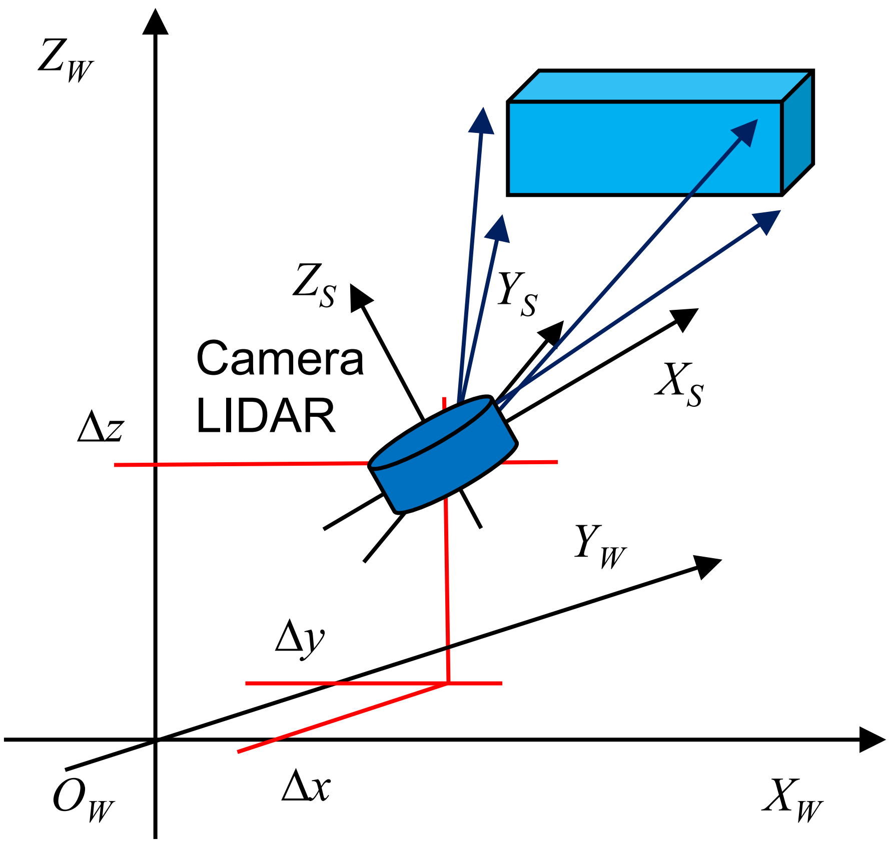


3D frame transformation

Motivation: Sensor in 3D Space

動機: 3次元空間でのセンシング

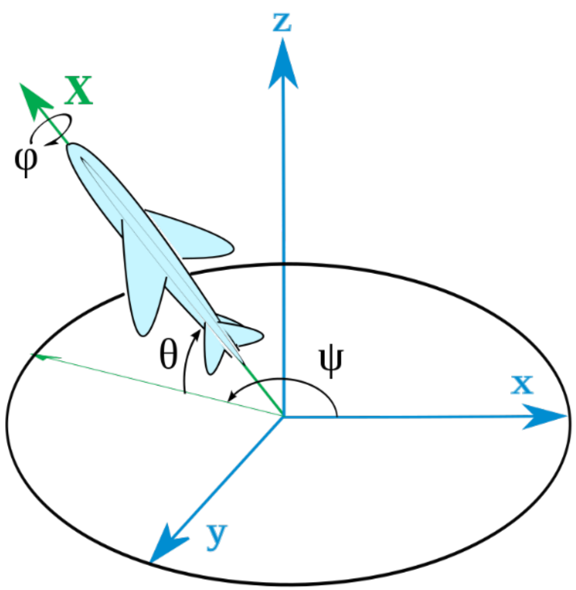
$$\begin{pmatrix} x_i^V \\ y_i^V \\ z_i^V \end{pmatrix} \rightarrow \begin{pmatrix} x_i^W \\ y_i^W \\ z_i^W \end{pmatrix} ?$$



- Translation in 3D is easy and no problem
- However, the orientation in 3D is not so easy
  - Roll, Pitch, Yaw angle or Euler angle
  - Rotation matrix
  - Quaternion
  - SO(3): Special Orthogonal Group of 3
- 3D Homogeneous transformation matrix
  3次元同次変換行列

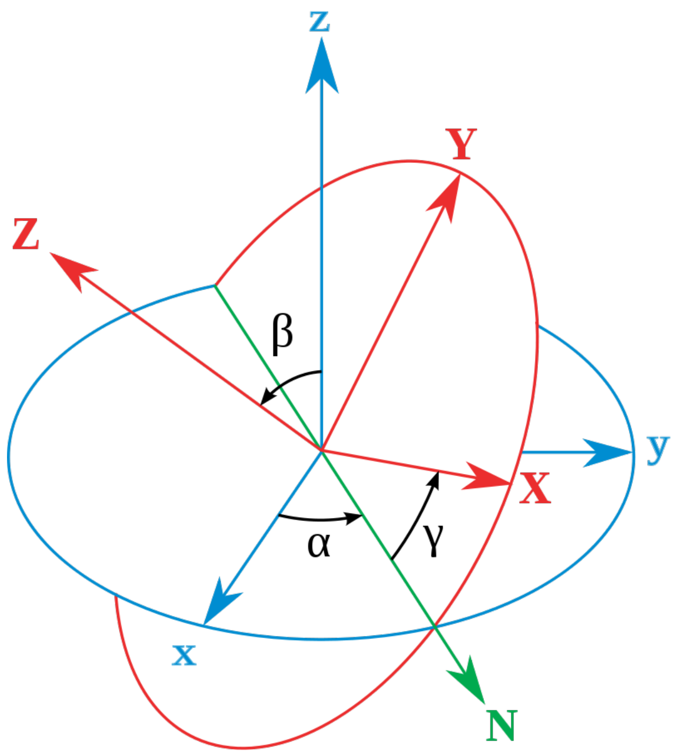
Roll, Pitch, and Yaw Angle (Variation of Euler Angle)

ロール角，ピッチ角，ヨー角（オイラー角）



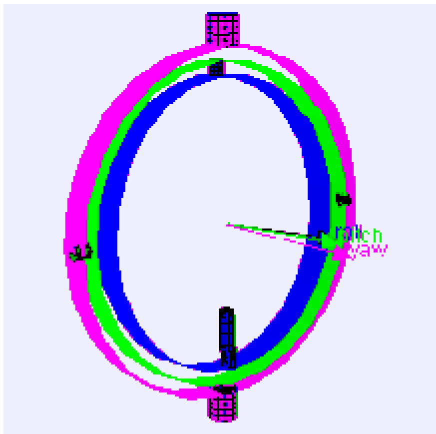
- Represent orientation of two frames by three angles
- Roll: ϕ represents a rotation around the x axis,
- Pitch: θ represents a rotation around the y axis,
- Yaw: ψ represents a rotation around the z axis
- Gimbal lock problem

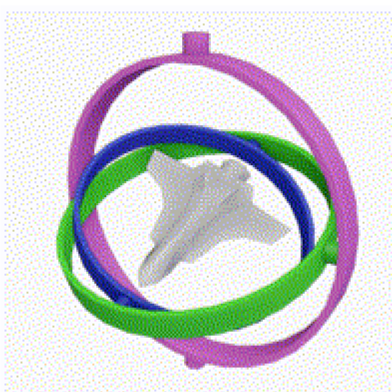
Euler Angle (Narrow Sense)
狭義のオイラー角



- Represent orientation of two frames by three angles
- α (or ϕ) represents a rotation around the z axis,
- β (or θ) represents a rotation around the x′ axis,
- γ (or ψ) represents a rotation around the z″ axis
- Gimbal lock problem

Problem of Euler Angle Representation: Gimbal Lock
オイラー角の問題:ジンバルロック





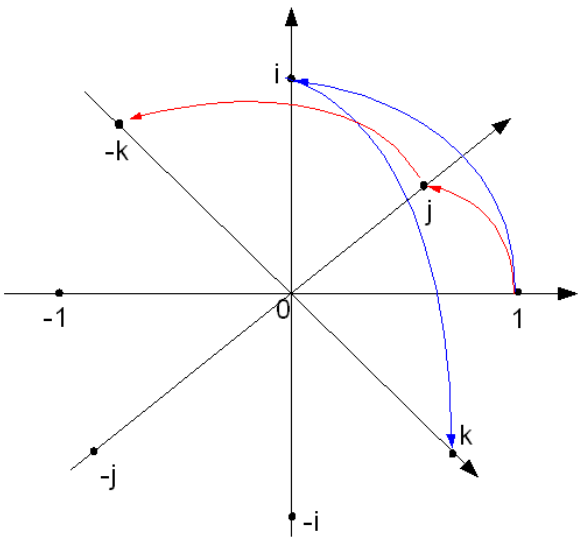
- If two axis rotates around the same axis, it loses one degree of freedom
- We cannot represent the orientation between the two frames
- (Numerical singular point)
  数値的な特異点

if $\cos\theta = 0$

we cannnot identify if $\theta = \frac{\pi}{2}$ or $\frac{-\pi}{2}$

we caanot calculate $\frac{1}{\cos\theta}$

Quaternion
クオータニオン・四元数



Graphical representation of quaternion units product as 90°-rotation in 4D-space

ij = k
ji = -k
ij = -ji

$$q = a + bi + cj + dk$$

- Represent three angles with one real and three imaginary numbers
- It is getting popular method and often used in CG
- Mathematically stable and no gimbal lock problem
- A bit redundant
- Not so intuitive

3D Rotation Matrix
3次元回転行列

Rotation around each axis

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

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

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

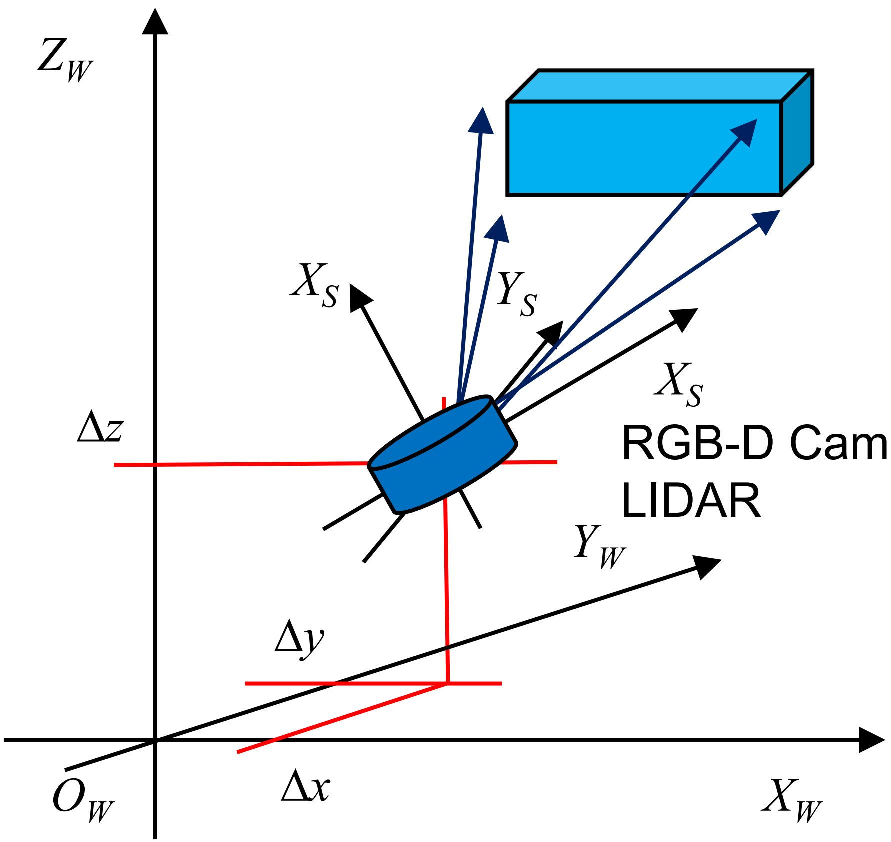
- Represent three angles with 3*3 matrix
- It is often used in robotics
- Mathematically stable and no gimbal lock problem
- Redundant representation

$\boldsymbol{n} = (n_x, n_y, n_z)^T$:unit axis

$$R_{\boldsymbol{n}}(\theta) = \begin{pmatrix} \cos\theta + n_x^2(1-\cos\theta) & n_x n_y(1-\cos\theta) - n_z\sin\theta & n_z n_x(1-\cos\theta) + n_y\sin\theta \\ n_x n_y(1-\cos\theta) + n_z\sin\theta & \cos\theta + n_y^2(1-\cos\theta) & n_y n_z(1-\cos\theta) - n_x\sin\theta \\ n_z n_x(1-\cos\theta) - n_y\sin\theta & n_y n_z(1-\cos\theta) + n_x\sin\theta & \cos\theta + n_z^2(1-\cos\theta) \end{pmatrix}$$

3D Homogeneous Transformation Matrix

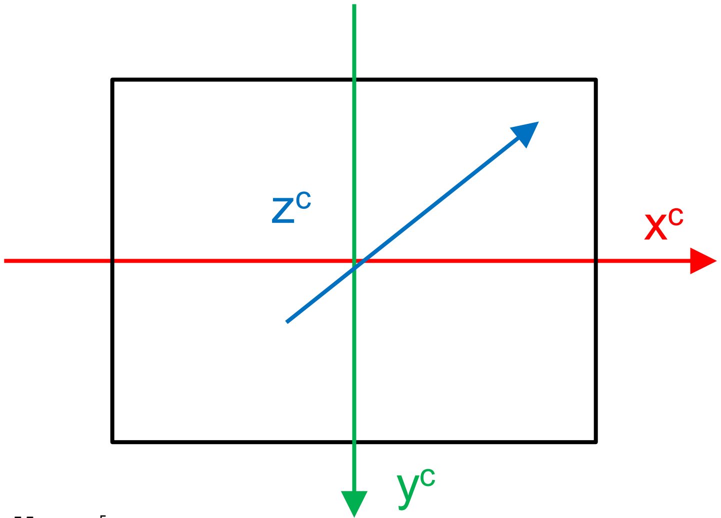
3次元同次変換行列



$$\begin{pmatrix} x_i^W \\ y_i^W \\ z_i^W \\ \hline 1 \end{pmatrix} = \left(\begin{array}{ccc|c} a_x^X & a_x^Y & a_x^Z & \Delta x \\ a_y^X & a_y^Y & a_y^Z & \Delta y \\ a_z^X & a_z^Y & a_z^Z & \Delta z \\ \hline 0 & 0 & 0 & 1 \end{array}\right) \begin{pmatrix} x_i^V \\ y_i^V \\ z_i^V \\ \hline 1 \end{pmatrix}$$

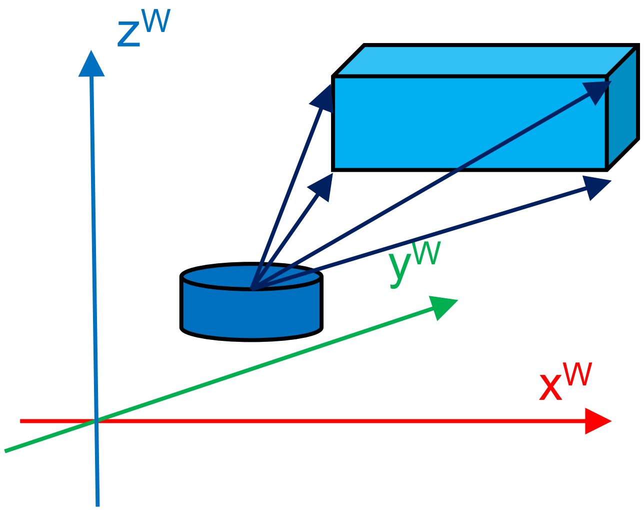
$$\boldsymbol{x}_i^W = T\boldsymbol{x}_i^V$$

Camera Frame




```
pV = [
      0.32, 0.32, -0.32, -0.32; % u = x
     -0.24, 0.24,  0.24, -0.24; % v = -y
      1.00, 1.00,  1.00,  1.0;  % z = depth
      1,    1,     1,     1  % constant
     ];
```

World Frame



Camera frame to world one
Rotate around x with 90 deg

Matlab Sample Code: Main

```
% Vehicle frame to world frame
Dx = 0.0; Dy = 0.0; Dz = 0.0;
R = deg2rad(90.0);
P = deg2rad( 0.0);
Y = deg2rad( 0.0);
% Transformatio matrix from camera to world frame
T = HTTrans([Dx; Dy; Dz]) * HTRotZ(Y) * HTRotY(P) * HTRotX(R);

% Homegeneous points in camera frame
pC = [
     0.32, 0.32, -0.32, -0.32; % u = x
    -0.24, 0.24,  0.24, -0.24; % v = -y
     1.00, 1.00,  1.00,  1.0;  % z = depth
     1,    1,     1,     1  % constant
    ];

% Homegeneous Points in world frame
pW = [ T*pC(:,1), T*pC(:,2), T*pC(:,3), T*pC(:,4)];

% Display points in vehicle and world frame at the same window
figure(1);
plot3(pC(1,:), pC(2,:), pC(3,:),'b+-', pW(1,:), pW(2,:), pW(3,:),'r*-')
xlim([-3 3]); ylim([-3 3]); zlim([-3 3]);
grid on; pbaspect([1 1 1]);
```

Matlab Sample Code: Functions

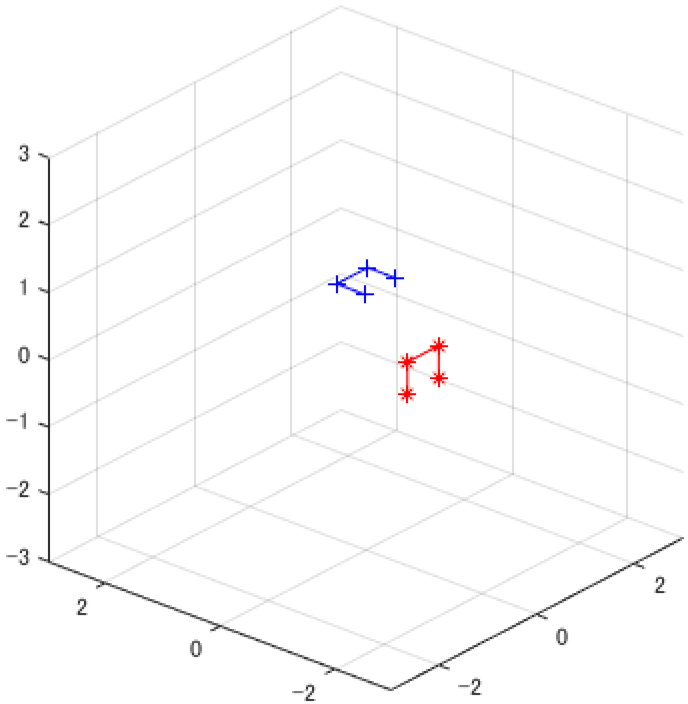
```
function [m] = HTRotX(q)
% HTRotX(q): returns a homegeneous transformation matirx of rotating an angle of q
around Y-axis
    m = [1, 0,       0,       0;...
         0, cos(q),-sin(q), 0;...
         0, sin(q), cos(q), 0;...
         0, 0,       0,       1];
end

function [m] = HTRotY(q)
% HTRotY(q): returns a homegeneous transformation matirx of rotating an angle of q
around Y-axis
    m = [ cos(q), 0, sin(q), 0;...
          0,       1, 0,       0;...
         -sin(q), 0, cos(q), 0;...
          0,       0, 0,       1];
end

function [m] = HTRotZ(q)
% HTRotZ(q): returns a homegeneous transformation matirx of rotating an angle of q
around Z-axis
    m = [cos(q), -sin(q), 0, 0;...
         sin(q),  cos(q), 0, 0;...
         0,        0,       1, 0;...
         0,        0,       0, 1];
end
```

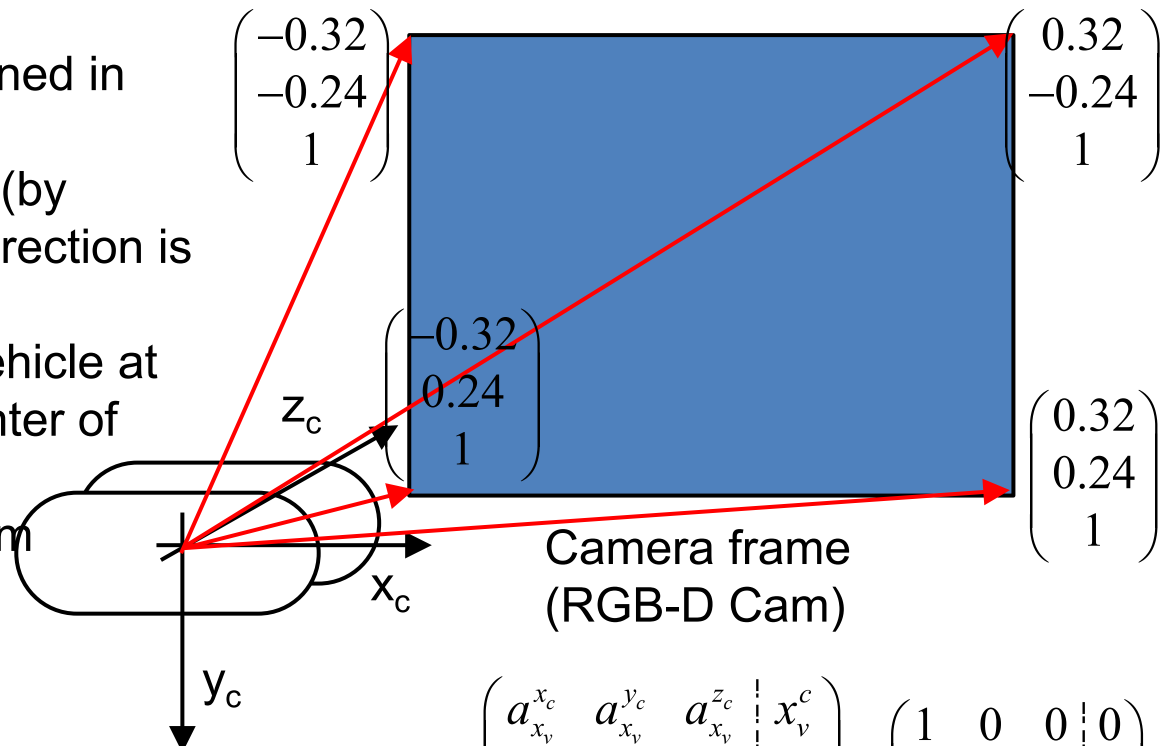
```
function [m] = HTTrans(t)
% HTTrans(t): returns a homegeneous transformation matirx of translation
% movetion with a column vector of t
    m = [1, 0, 0, t(1);...
         0, 1, 0, t(2);...
         0, 0, 1, t(3);...
         0, 0, 0, 1];
end
```

Result

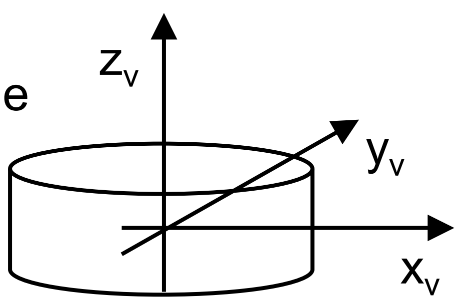


Quiz #2: Frame Transformation in 3D: Frame Conversion from Camera to Vehicle

- Camera has its frame (defined in camera SDK)
- Vehicle has another frame (by convention, e.g., moving direction is x)
- Camera has attached to vehicle at a height of 1.0 m in the center of vehicle
- How do we represent T from camera to vehicle frame

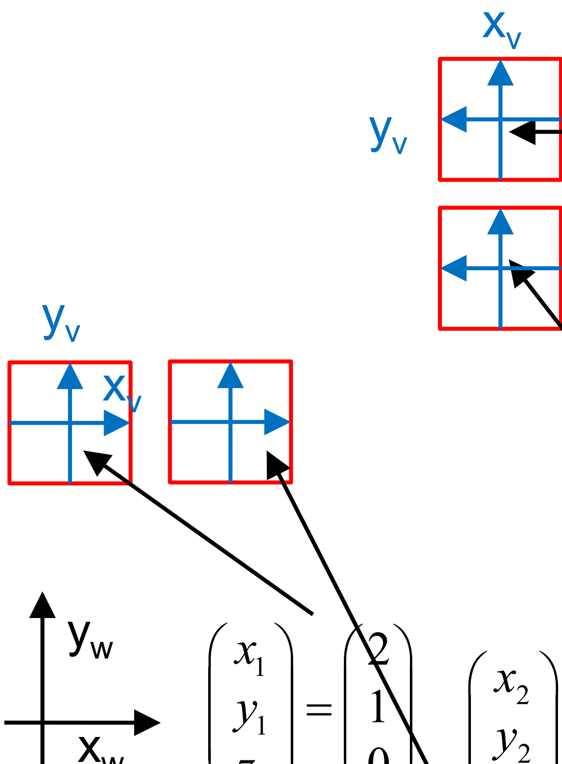


Vehicle frame
(Turtlebot)



$$T_v^c = \left(\begin{array}{ccc|c} a_{x_v}^{x_c} & a_{x_v}^{y_c} & a_{x_v}^{z_c} & x_v^c \\ a_{y_v}^{x_c} & a_{y_v}^{y_c} & a_{y_v}^{z_c} & y_c^c \\ a_{z_v}^{x_c} & a_{z_v}^{y_c} & a_{z_v}^{z_c} & z_v^v \\ \hline 0 & 0 & 0 & 1 \end{array}\right) = \left(\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 1 \\ \hline 0 & 0 & 0 & 1 \end{array}\right)$$

Quiz #2: Frame Transformation in 3D: Frame Conversion from Vehicle to World



$$\begin{pmatrix} x_4 \\ y_4 \\ z_4 \end{pmatrix} = \begin{pmatrix} 5 \\ 4 \\ 0 \end{pmatrix}, \begin{pmatrix} r_4 \\ p_4 \\ y_4 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \pi/2 \end{pmatrix}$$

$$\begin{pmatrix} x_3 \\ y_3 \\ z_3 \end{pmatrix} = \begin{pmatrix} 5 \\ 3 \\ 0 \end{pmatrix}$$

$$\begin{pmatrix} r_3 \\ p_3 \\ y_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \pi/2 \end{pmatrix}$$

$$\begin{pmatrix} x_1 \\ y_1 \\ z_1 \end{pmatrix} = \begin{pmatrix} 2 \\ 1 \\ 0 \end{pmatrix} \quad \begin{pmatrix} x_2 \\ y_2 \\ z_2 \end{pmatrix} = \begin{pmatrix} 3 \\ 1 \\ 0 \end{pmatrix}$$

$$\begin{pmatrix} r_1 \\ p_1 \\ y_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \quad \begin{pmatrix} r_2 \\ p_2 \\ y_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

- Vehicle moves in world frame
- Make a homogeneous transformation matrix from vehicle to world in each vehicle position

$$T_w^{v4} = \left(\begin{array}{ccc|c} a_{x_w}^{x_v} & a_{x_w}^{y_v} & a_{x_w}^{z_v} & x_w^v \\ a_{y_w}^{x_v} & a_{y_w}^{y_c} & a_{y_v}^{z_v} & y_v^w \\ a_{z_w}^{x_v} & a_{z_w}^{y_c} & a_{z_w}^{z_v} & z_v^w \\ \hline 0 & 0 & 0 & 1 \end{array}\right) = \left(\begin{array}{ccc|c} 0 & -1 & 0 & 5 \\ 1 & 0 & 0 & 4 \\ 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right)$$

Quiz #2: Frame Transformation in 3D: Problem Setting

Sensor measures the followings at
vehicle position 1, 2, 3, 4

$$p^c = \begin{pmatrix} -0.32 \\ -0.24 \\ 1 \end{pmatrix}, \begin{pmatrix} 0.32 \\ -0.24 \\ 1 \end{pmatrix}, \begin{pmatrix} -0.32 \\ 0.24 \\ 1 \end{pmatrix}, \begin{pmatrix} -0.32 \\ 0.24 \\ 1 \end{pmatrix}$$

Make an integrated 3D map in world frame from the measured points
at the five vehicle positions: camera frame -> vehicle frame -> world frame

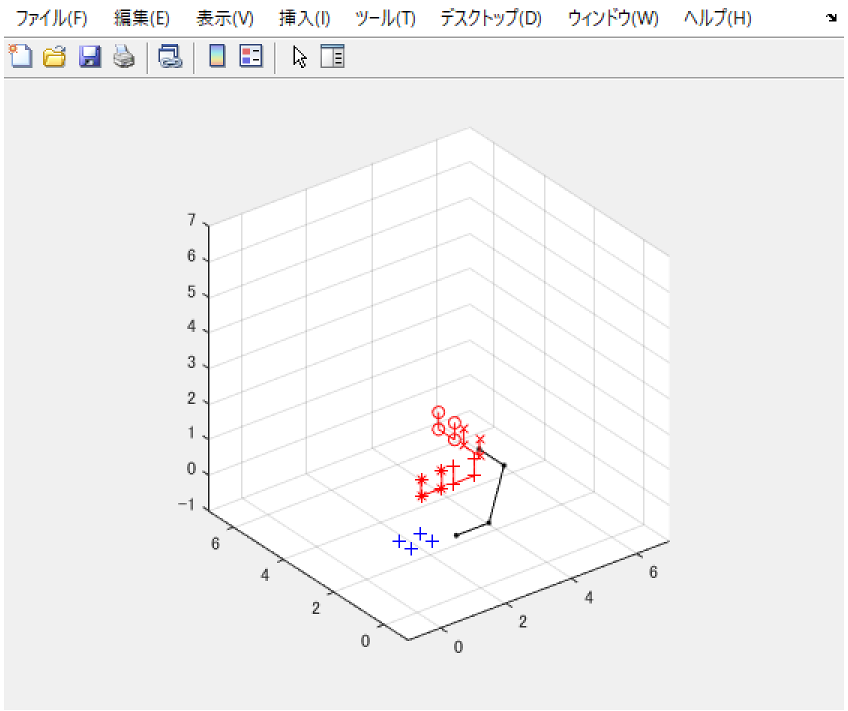
For example,
Measurement at vehicle position 1

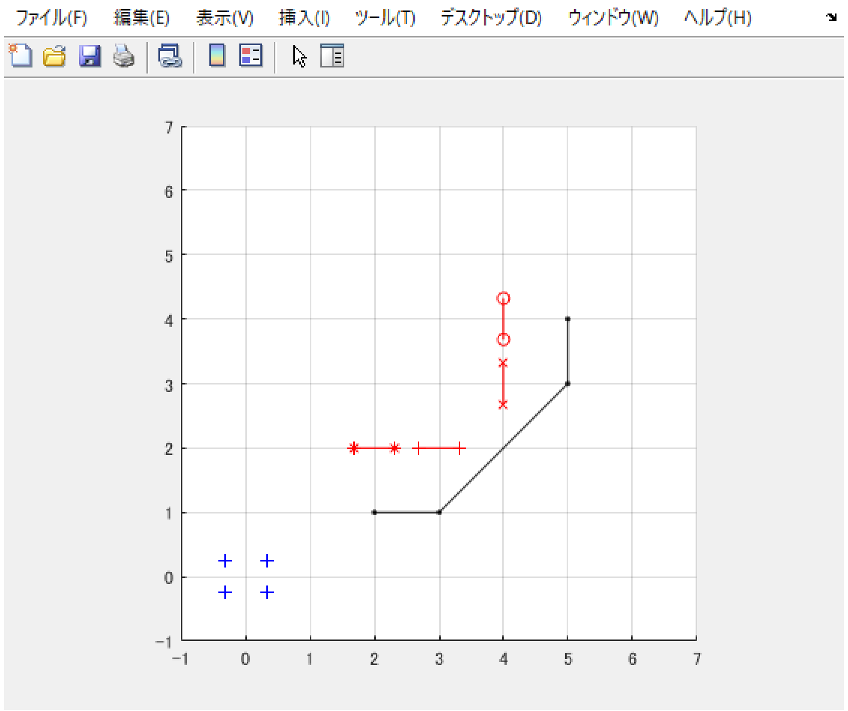
$$T_v^c = \left(\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 1 \\ \hline 0 & 0 & 0 & 1 \end{array}\right) \quad T_w^v == \left(\begin{array}{ccc|c} 1 & 0 & 0 & 2 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right)$$

Measurement at vehicle position 4s

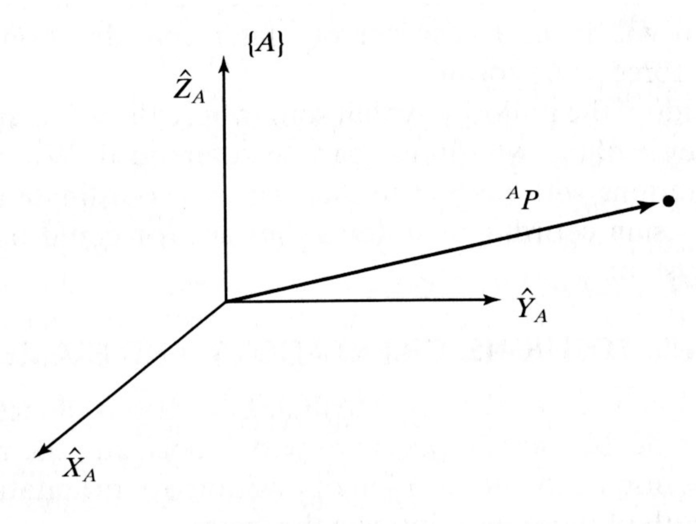
$$T_v^c = \left(\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 1 \\ \hline 0 & 0 & 0 & 1 \end{array}\right) \quad T_w^v == \left(\begin{array}{ccc|c} 0 & -1 & 0 & 4 \\ 1 & 0 & 0 & 3 \\ 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right)$$

Sample Result



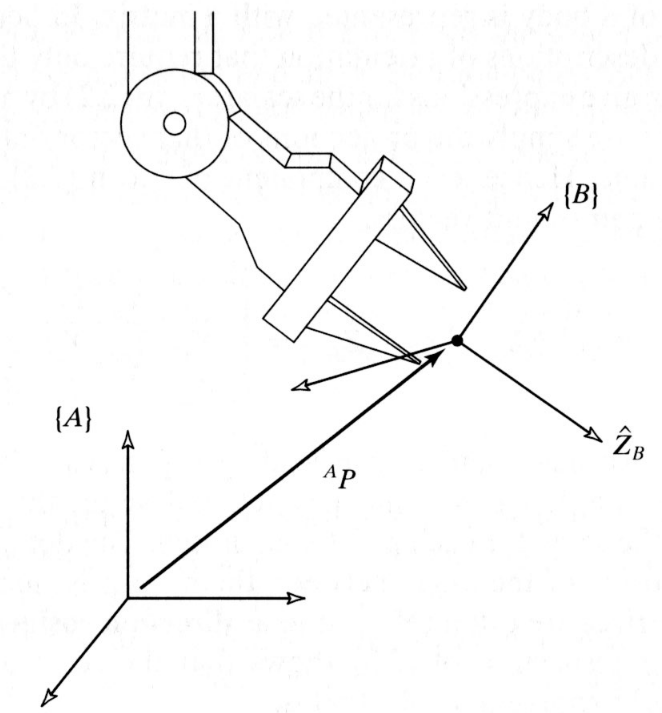


Description of Position



$$ {}^{A}P = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} $$

Description of Orientation



$\hat{X}_B, \hat{Y}_B, \hat{Z}_B$ Unit vectors of frame {B}

$^A\hat{X}_B, ^A\hat{Y}_B, ^A\hat{Z}_B$ Unit vectors of frame {B} written in frame {A}

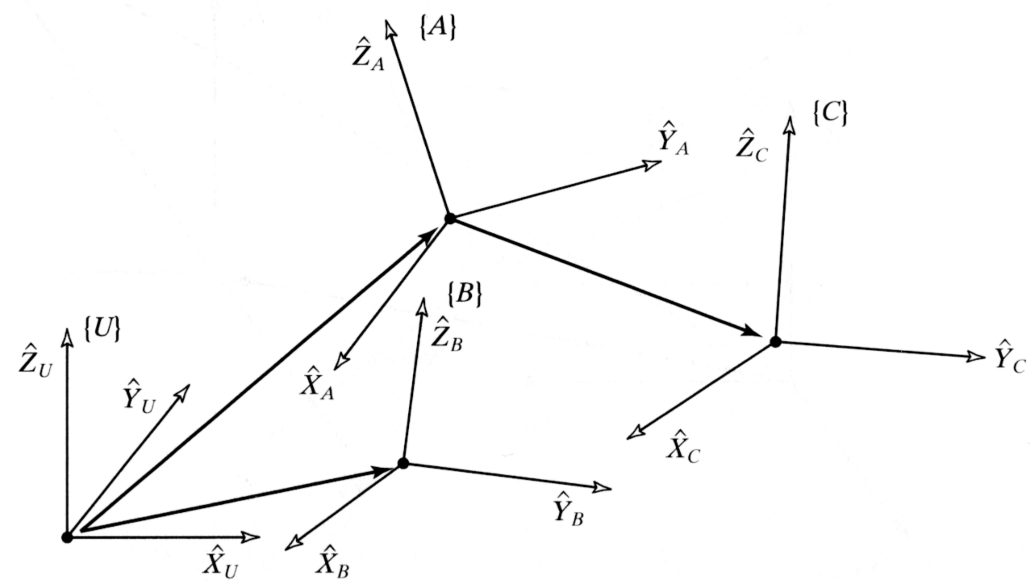
Rotation matrix (from B to A) (B in A)

$$
{}^A_BR = \left[{}^A\hat{X}_B, {}^A\hat{Y}_B, {}^A\hat{Z}_B \right]
= \begin{bmatrix} \hat{X}_B \bullet \hat{X}_A & \hat{Y}_B \bullet \hat{X}_A & \hat{Z}_B \bullet \hat{X}_A \\ \hat{X}_B \bullet \hat{Y}_A & \hat{Y}_B \bullet \hat{Y}_A & \hat{Z}_B \bullet \hat{Y}_A \\ \hat{X}_B \bullet \hat{Z}_A & \hat{Y}_B \bullet \hat{Z}_A & \hat{Z}_B \bullet \hat{Z}_A \end{bmatrix}
$$

$$
{}^A_BR = {}^B_AR^{-1} = {}^B_AR^T
$$

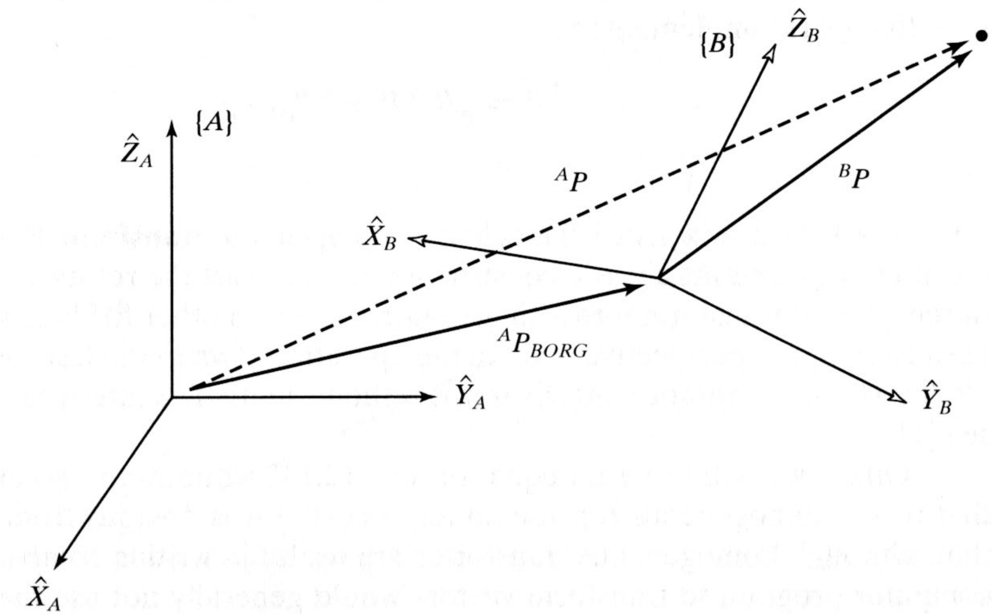
Description of Frame

$$\{B\} = \{{}^A_B R, {}^A P_{B\ ORG}\}$$ Frame {B} is described relative to Frame {A}



$$\{A\} = \{{}^U_A R, {}^U P_{A\ ORG}\}, \{B\} = \{{}^U_B R, {}^U P_{B\ ORG}\}, \{C\} = \{{}^A_C R, {}^A P_{C\ ORG}\}$$

Homogeneous Transformation



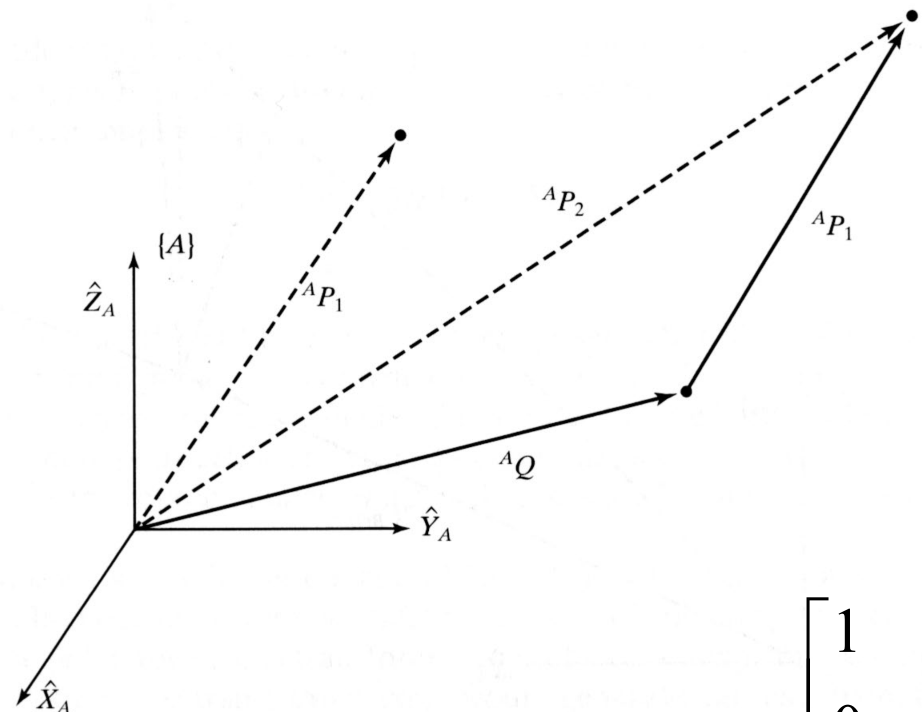
$$ {}^{A}P = {}^{A}_{B}R\,{}^{B}P + {}^{A}P_{B\ ORG} $$

$$ \begin{bmatrix} {}^{A}P \\ 1 \end{bmatrix} = \left[\begin{array}{ccc|c} & {}^{A}_{B}R & & {}^{A}P_{B\ ORG} \\ \hline 0 & 0 & 0 & 1 \end{array} \right] \begin{bmatrix} {}^{B}P \\ 1 \end{bmatrix} $$

Modify it conceptually in the form

$$ {}^{A}P = {}^{A}_{B}T\,{}^{B}P $$

Translational Operator

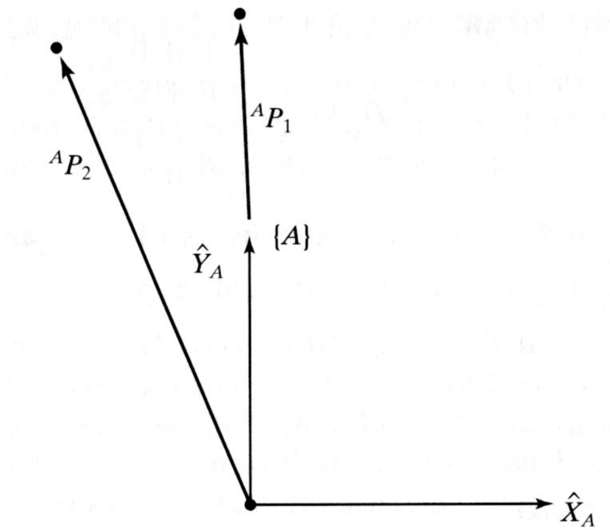


$$^AP_2 = {}^AP_1 + {}^AQ$$

$$^AP_2 = D_Q(q)\ {}^AP_1$$

$$D_Q(q) = \left[\begin{array}{ccc|c} 1 & 0 & 0 & q_x \\ 0 & 1 & 0 & q_y \\ 0 & 0 & 1 & q_z \\ \hline 0 & 0 & 0 & 1 \end{array}\right]$$

Rotational Operator



Rotation about Z axis

$$R_z(\theta)=\left[\begin{array}{ccc|c}\cos\theta & -\sin\theta & 0 & 0\\ \sin\theta & \cos\theta & 0 & 0\\ 0 & 0 & 1 & 0\\ \hline 0 & 0 & 0 & 1\end{array}\right]$$

Rotation about Y axis

$$R_Y(\theta)=\left[\begin{array}{ccc|c}\cos\theta & 0 & \sin\theta & 0\\ 0 & 1 & 0 & 0\\ -\sin\theta & 0 & \cos\theta & 0\\ \hline 0 & 0 & 0 & 1\end{array}\right]$$

Rotation about X axis

$$R_X(\theta)=\left[\begin{array}{ccc|c}1 & 0 & 0 & 0\\ 0 & \cos\theta & -\sin\theta & 0\\ 0 & \sin\theta & \cos\theta & 0\\ \hline 0 & 0 & 0 & 1\end{array}\right]$$

Matlab Sample Code

A function returning a 4*4 homogeneous transformation matrix of rotating around z axis with an angle of q

Function definition

```
function [m] = HTRotZ(q)
m = [cos(q), -sin(q), 0, 0;
         sin(q), cos(q), 0, 0;
         0, 0, 1, 0;
         0, 0, 0, 1];
end
```

Test code

```
p = [1; 2; 3; 0];
m = HTRotZ( deg2rad(30) );
m*p
```

Matlab Sample Code

A function returning a 4*4 homogeneous transformation matrix of rotating around Y axis with an angle of q

Function definition

```
function [m] = HTRotY(q)
m = [cos(q), 0, sin(q), 0;
        0, 1, 0, 0;
        -sin(q), 0, cos(q), 0;
         0, 0, 0, 1];
end
```

Test code

```
p = [1; 2; 3; 0];
m = HTRotY( deg2rad(30) );
m*p
```

Matlab Sample Code

A function returning a 4*4 homogeneous transformation matrix of rotating around x axis with an angle of q

Function definition

```
function [m] = HTRotX(q)
m = [1, 0, 0, 0;
         0, cos(q), -sin(q), 0;
         0, sin(q), cos(q), 0;
         0, 0, 0, 1];
end
```

Test code

```
p = [1; 2; 3; 0];
m = HTRotX( deg2rad(30) );
m*p
```

Matlab Sample Code

A function returning a 4*4 homogeneous transformation matrix of translating a vector x of t = [x; y; z]

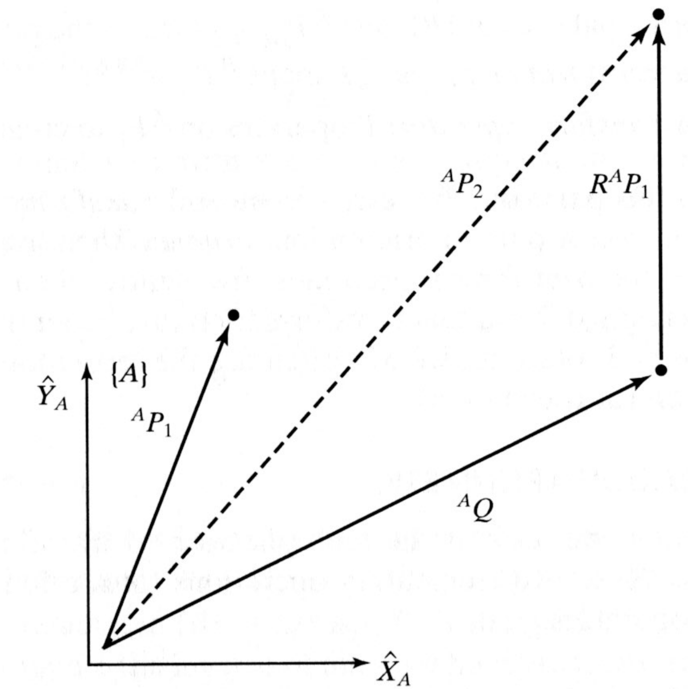
Function definition

```
function [m] = HTTrans(t)
m = [1, 0, 0, t(1);
        0, 1, 0, t(2);
        0, 0, 1, t(3);
        0, 0, 0, 1];
end
```

Test code

```
t = [100; 100, 0];
p = [1; 2; 3; 0];
m = HTTrans(t);
m*p
```

Transformation Operator



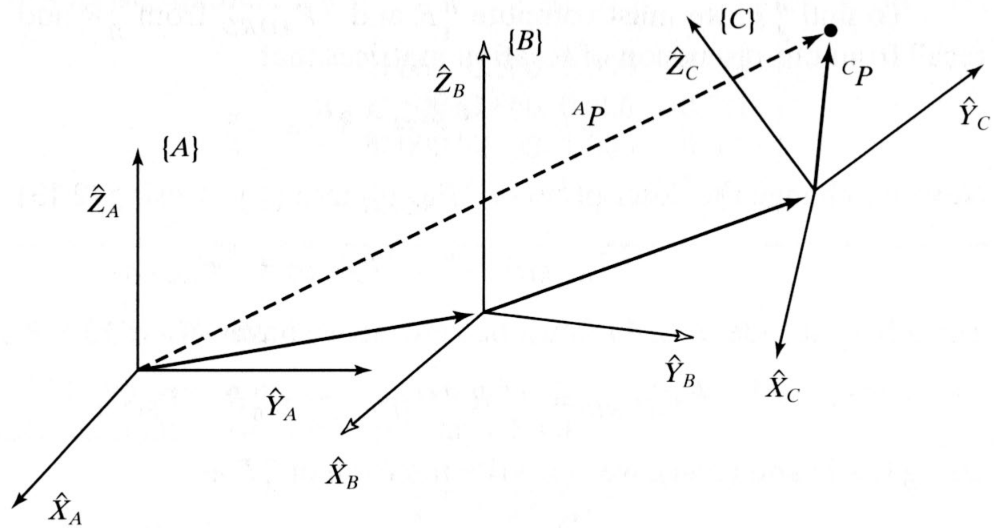
Rotate about Z by 30 degree
and translate it 10 in X_A, 5 in Y_A

$$T = \left[\begin{array}{ccc|c} 0.866 & -0.500 & 0.000 & 10.0 \\ 0.500 & 0.866 & 0.000 & 5.0 \\ 0.000 & 0.000 & 1.000 & 0.0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right]$$

$$T = Trans(t)\,RotZ(\theta)$$

$$= \begin{pmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos 30 & -\sin 30 & 0 & 0 \\ \sin 30 & \cos 30 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} \cos 30 & -\sin 30 & 0 & x \\ \sin 30 & \cos 30 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

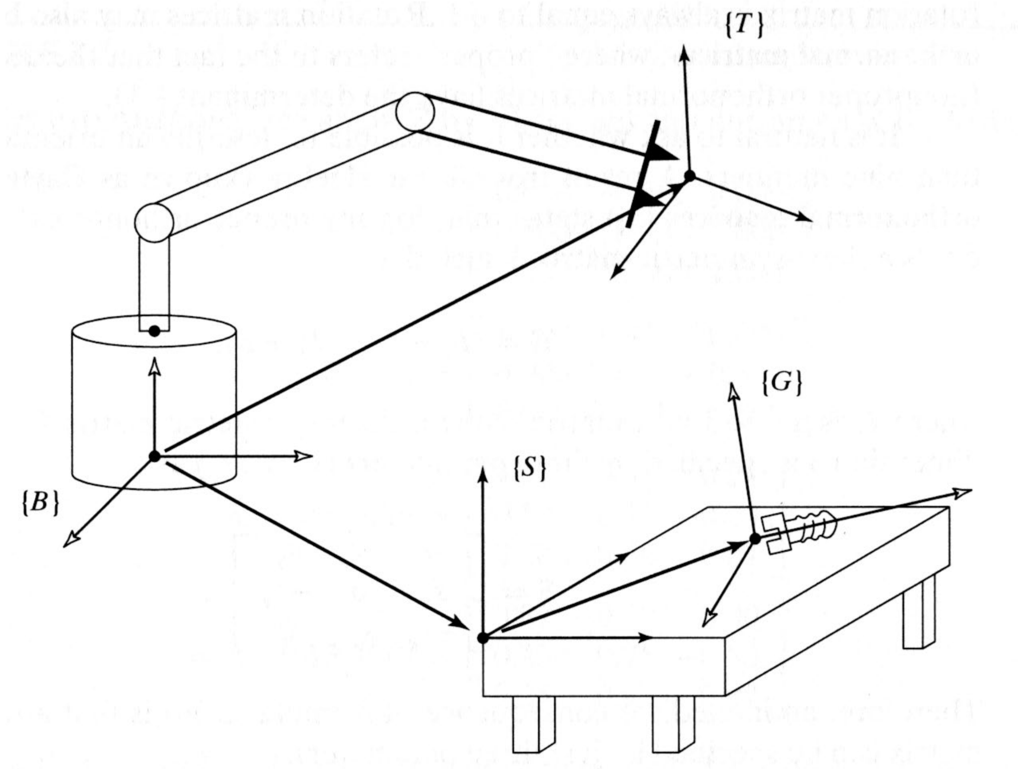
Compound and Invert Transformation



$$ {}^A_C T = {}^A_B T \, {}^B_C T \qquad {}^A_C T = \left[\begin{array}{ccc|c} & {}^A_B R \, {}^B_C R & & {}^A_B R \, {}^B P_{C\ ORG} + {}^A P_{B\ ORG} \\ \hline 0 & 0 & 0 & 1 \end{array} \right] $$

$$ {}^B_A T = {}^A_B T^{-1} \qquad {}^B_A T = \left[\begin{array}{ccc|c} & {}^A_B R^{\,T} & & -{}^A_B R^T \, {}^A P_{B\ ORG} \\ \hline 0 & 0 & 0 & 1 \end{array} \right] $$

Example



$$
{}^{T}_{G}T = {}^{T}_{B}T \; {}^{B}_{S}T \; {}^{S}_{G}T
$$

$$
= {}^{B}_{T}T^{-1} \; {}^{B}_{S}T \; {}^{S}_{G}T
$$

B: Base
T: Tool (Gripper)
S: Workspace (Table)
G: Object (Bolt)