1 Pose Description

Pose: position and orientation

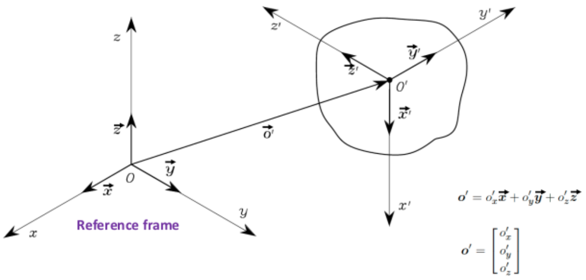

Figure 1: pose smaple

We can desctibe the new frame (’) with the following projection:

$$\vec{x'} = {x'}_x\vec{X} + {x'}_y\vec{Y} + {x'}_z\vec{Z}$$
$$\vec{y'} = {y'}_x\vec{X} + {y'}_y\vec{Y} + {y'}_z\vec{Z}$$
$$\vec{z'} = {z'}_x\vec{X} + {z'}_y\vec{Y} + {z'}_z\vec{Z}$$

It describes the (’) frame completely.

1.1 Rotation Matrix

$$R = \begin{bmatrix} X' & Y' & Z' \end{bmatrix} = \begin{bmatrix} x^{'}_x & y^{'}_x & z^{'}_x \\ x^{'}_y & y^{'}_y & z^{'}_y \\ x^{'}_z & y^{'}_z & z^{'}_z \end{bmatrix} = \begin{bmatrix} {x'}_x\vec{X} & {x'}_y\vec{Y} & {x'}_z\vec{Z} \\ {y'}_x\vec{X} & {y'}_y\vec{Y} & {y'}_z\vec{Z} \\ {z'}_x\vec{X} & {z'}_y\vec{Y} & {z'}_z\vec{Z} \end{bmatrix} \tag{1}$$

reminder:

$$\vec{x'}^T\vec{y'} = 0 \qquad \vec{y'}^T\vec{z'} = 0 \qquad \vec{z'}^T\vec{x'} = 0$$
$$\vec{x'}^T\vec{x'} = 1 \qquad \vec{y'}^T\vec{y'} = 1 \qquad \vec{z'}^T\vec{z'} = 1$$

therefore, rotation matrix properties:

$$R^TR = I$$
$$R^T = R^{-1}$$
$$|\det(R)| = 1$$
$$R \in SO(m)$$

Special Orthonormal group: real m×m matrices with orthonormal columns and determinant=1

1.2 Rotation along the axis - Elementry Rotations

Rotation matrices

$$R_z(\alpha) = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

$$R_y(\beta) = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \tag{3}$$

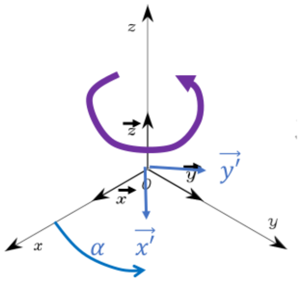

Figure 2: Rotation along z axis

$$R_x(\gamma) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{bmatrix} \tag{4}$$

Rotation in the "opposite" direction is equivalent to the inverse R matrix which is the transpose rotation matrix:

$$R_k(-\vartheta) = \{R_k^{-1}(\vartheta)\} = R_k^T(\vartheta) \quad ; \quad k = x, y, z$$

(*) see rotation matrix properties

1.3 Representation of a vector

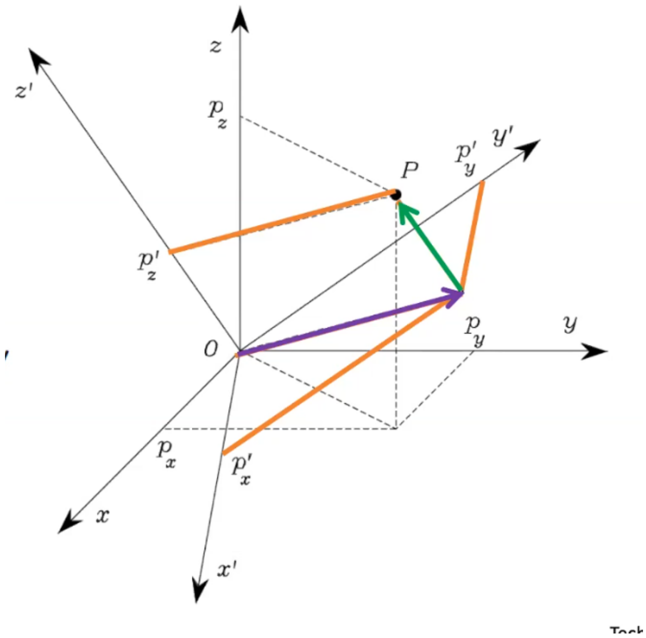

Figure 3: vector representation

$$o^{'} = O$$

$$P = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} \quad P^{'} = \begin{bmatrix} p'_x \\ p'_y \\ p'_z \end{bmatrix}$$

$$P = p'_x x' + p'_y y' + p'_z z' = \begin{bmatrix} x' & y' & z' \end{bmatrix} p'$$

$$p = Rp'$$

$$p' = R^T p$$

R Matrix represents:

1. The orientation of O' w.r.t O
2. The transformation of vector from O' to O
3. Rotation of a vector (in the same frame)

1.4 Composition of Rotation matrices

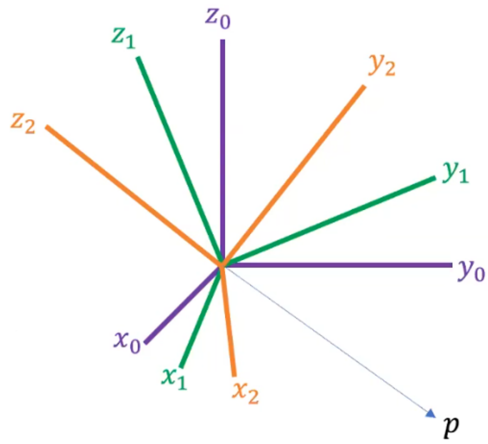

Figure 4: composition

- Consider 3 frames with same origin
- A point represented in each frame: p^0, p^1, p^2

Transformation of O_2 w.r.t O_1: $p^1 = R_2^1 p^2$

$$p^0 = R_1^0 p^1$$
$$p^0 = R_2^0 p^2 \quad \Longrightarrow \quad R_2^0 = R_1^0 R_2^1$$

(*) from left to right from 0 to 1 and than from 1 to 2 - "current frame" method

we can transform from 0 to 2 in two steps:

- First rotate the given frame according to R_1^0, so as to align it with frame $O_{x_1y_1z_1}$
- Then rotate the frame, now aligned with frame $O_{x_1y_1z_1}$ according to R_2^1 so as to align it with frame $O_{x_2y_2z_2}$

The order matters - rotation transformation do not cummute!

So we can see that we can reverse the transformation using:

$$R_i^j = (R_j^i)^{-1} = (R_i^j)^T$$

1.5 Composition Current Frame vs. Fixed Frame

- **Current Frame** Consider the following sequence of rotations:
  1. Rotate around z_0
  2. Rotate around x_1
- **Fixed Frame** Consider the following sequence of rotations:
  1. Rotate around z_0
  2. Rotate around x_0

It is obvious that the rotation gives us different rotation (current vs. fixed)
We cant use the same formula!

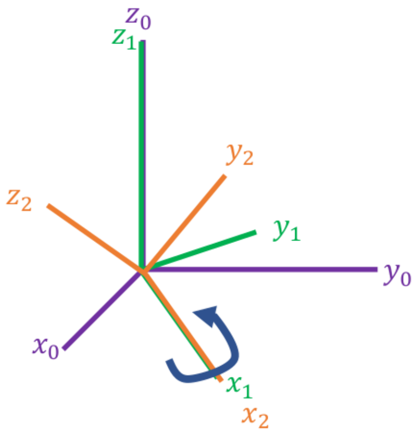

Figure 5: composition

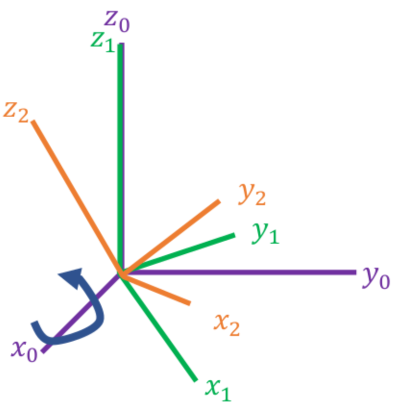

Figure 6: composition

1.6 Fixed Frame

Define:

- R_1^0: rotation for O_1 w.r.t O_0
- R: rotation of O_1 w.r.t O_0 to obtain O_2

So: $R_2^0 = R_1^0 R_2^1$: rotation for O_2 w.r.t O_0 but we know it is nto the transformation we want so... $R_2^0 \neq R_1^0 R$

What we really want is:

$$R_2^0 = R_1^0 \left[(R_1^0)^{-1} R R_1^0 \right]$$

after opening the brackets: $(R_1^0 (R_1^0)^{-1} = R_1^0 (R_1^0)^T = I)$

$$R_2^0 = R R_1^0$$

Fix frame recipe:

1. Rotate O_1 to align with O_0
2. Apply R (by definition)
3. Undo the rotation in (1)

For rotation w.r.t fixed-frame - multiply the rotation matrices in reverse order!

2 Parameterization of rotations

- Rotaion matrix has 9 elements
- The minimum parameters needed to define arbitrary rotation - 3
- there are several parameterization options.

2.1 Euler angles

- Eulre angles $(ZYZ)(\phi, \theta, \psi)$
  1. Rotate about Z axis by an angle ϕ, $\quad R_z(\phi)$
  2. Rotate about **current** y axis by an angle θ, $\quad R_y(\theta)$
  3. Rotate about **current** z axis by an angle ψ, $\quad R_z(\psi)$

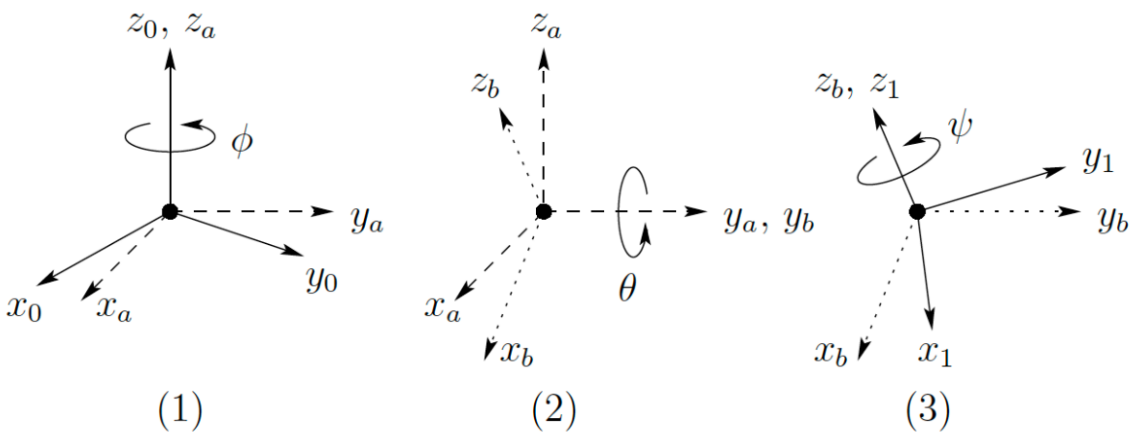

Figure 7: Eulre angles (ϕ, θ, ψ)

$$R_1^0 = R_{z,\phi} R_{y,\theta} R_{z,\psi} = \begin{bmatrix} C_\phi & -S_\phi & 0 \\ S_\phi & C_\phi & -0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} C_\theta & 0 & S_\theta \\ 0 & 1 & 0 \\ -S_\theta & 0 & C_\theta \end{bmatrix} \begin{bmatrix} C_\psi & -S_\psi & 0 \\ S_\psi & C_\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} =$$

$$= \begin{bmatrix} C_\phi C_\theta C_\psi - S_\phi S_\psi & -C_\phi C_\theta S_\psi - S_\phi C_\psi & C_\phi S_\theta \\ S_\phi C_\theta C_\psi + C_\phi S_\psi & -S_\phi C_\theta S_\psi + C_\phi C_\psi & S_\phi S_\theta \\ -S_\theta C_\psi & S_\theta S_\psi & C_\theta \end{bmatrix}$$

4. Roll-Pitch-yaw (fixed frame) angles: (ϕ, θ, ψ)
   - Rotate about **fixed** x axis by an angle ϕ, $\quad R_x(\phi)$
   - Rotate about **fixed** y axis by an angle θ, $\quad R_y(\theta)$
   - Rotate about **fixed** z axis by an angle ψ, $\quad R_z(\psi)$
     (*) Fixed frame(!) multiply in reverse order.

$$R_1^0 = R_{z,\phi} R_{y,\theta} R_{x,\psi} = \begin{bmatrix} C_\phi & -S_\phi & 0 \\ S_\phi & C_\phi & -0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} C_\theta & 0 & S_\theta \\ 0 & 1 & 0 \\ -S_\theta & 0 & C_\theta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & C_\psi & -S_\psi \\ 0 & S_\psi & C_\psi \end{bmatrix} =$$

$$= \begin{bmatrix} C_\phi C_\theta & -S_\phi C_\psi + C_\phi S_\theta S_\psi & S_\phi S_\psi + C_\phi S_\theta C_\psi \\ S_\phi C_\theta & C_\phi C_\psi + S_\phi S_\theta S_\psi & -C_\phi S_\psi + S_\phi S_\theta C_\psi \\ -S_\theta & C_\theta S_\psi & C_\theta C\psi \end{bmatrix}$$

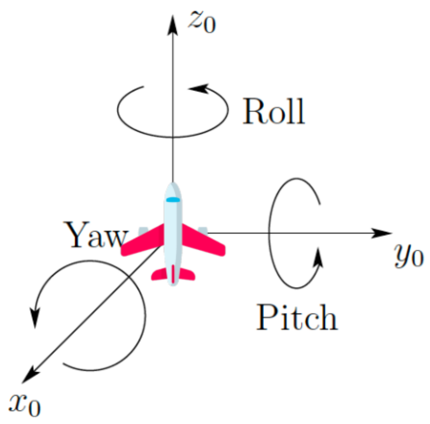

Figure 8: roll-pitch-yaw (bad axis tags... usually the x axis is along the heading of the plane) (ϕ, θ, ψ)

2.1.1 Euler angles - inverse problem

1. Rotate about **fixed** x by an angle ϕ
2. Rotate about **fixed** y by an angle θ
3. Rotate about **fixed** z by an angle ψ

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \rightarrow (\phi, \theta, \psi)$$

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} = \begin{bmatrix} C_\phi C_\theta & -S_\phi C_\psi + C_\phi S_\theta S_\psi & S_\phi S_\psi + C_\phi S_\theta C_\psi \\ S_\phi C_\theta & C_\phi C_\psi + S_\phi S_\theta S_\psi & -C_\phi S_\psi + S_\phi S_\theta C_\psi \\ -S_\theta & C_\theta S_\psi & C_\theta C\psi \end{bmatrix}$$

$$\begin{cases} C_\theta S_\psi = r_{32} \\ C_\theta C_\psi = r_{13} \end{cases} = \begin{cases} C_\theta^2 S_\psi^2 = r_{32}^2 \\ C_\theta^2 C_\psi^2 = r_{13}^2 \end{cases} + \tag{5}$$

$$C_\theta^2 (S_\psi^2 + C_\psi^2) = r_{32}^2 + r_{33}^2$$

$$C_\theta = \pm\sqrt{r_{32}^2 + r_{33}^2}$$

In case of positive θ

$$\begin{cases} C_\phi C_\theta = r_{11} \\ S_\pi C_\theta = r_{21} \end{cases} \implies \frac{S_\phi}{C_\phi} = \frac{r_{21} C_\theta}{r_{11} C_\theta} \tag{6}$$

$$\phi = Atan2(r_{21}, r_{11})$$

In case of negative θ

$$\phi = Atan2(-r_{21}, r_{11})$$

$$\begin{cases} C_\theta S_\psi = r_{32} \\ C_\theta C_\psi = r_{13}) \end{cases} \rightarrow \psi = Atan2(r_{32}, r_{33}) \tag{7}$$

(*) all calculation are correct only if $C_\theta \neq 0$

In case of $C_\theta = 0$

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} = \begin{bmatrix} \cancelto{0}{C_\phi C_\theta} & -S_\phi C_\psi + C_\phi \cancelto{1}{S_\theta} S_\psi & S_\phi S_\psi + C_\phi \cancelto{1}{S_\theta} C_\psi \\ \cancelto{0}{S_\phi C_\theta} & C_\phi C_\psi + S_\phi \cancelto{1}{S_\theta} S_\psi & -C_\phi S_\psi + S_\phi \cancelto{1}{S_\theta} C_\psi \\ \cancelto{1}{-S_\theta} & \cancelto{0}{C_\theta S_\psi} & \cancelto{0}{C_\theta C_\psi} \end{bmatrix} =$$

$$\begin{bmatrix} 0 & \cancelto{S_{\psi-\phi}}{-S_\phi C_\psi + C_\phi S_\psi} & \cancelto{C_{\psi-\phi}}{S_\phi S_\psi + C_\phi C_\psi} \\ 0 & \cancelto{C_{\psi-\phi}}{C_\phi C_\psi + S_\phi S_\psi} & \cancelto{-S_{\psi-\phi}}{-C_\phi S_\psi + S_\phi C_\psi} \\ 1 & 0 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & S_{\psi-\phi} & C_{\psi-\phi} \\ 0 & C_{\psi-\phi} & -S_{\psi-\phi} \\ 1 & 0 & 0 \end{bmatrix}$$

(*) If $x = (\psi - \phi)$ we can see that we can only find the diff. between the angles.

Summary: $for \theta \in (-\frac{\pi}{2}, \frac{\pi}{2})$ $(C_\theta > 0)$

$$\phi = Atan2(r_{21}, r_{11})$$

$$\theta = Atan2(-r_{31}, \sqrt{r_{32}^2) + r_{33}^2})$$

$$\psi = Atan2(r_{32}, r_{33})$$

For $\theta \in (\frac{\pi}{2}, \frac{3\pi}{2})$ $(C_\theta < 0)$ there is another solution...
Solution for $C_\theta = 0$ degenerate. Only possible to determine the sum or difference of ϕ, ψ (**Gimbal lock**, Euler angle issue)
(*) Limitation - dont rotate twice in a sequence by the same axis e.g XXY...
so we have 12 valid configuration.

Atan2 function

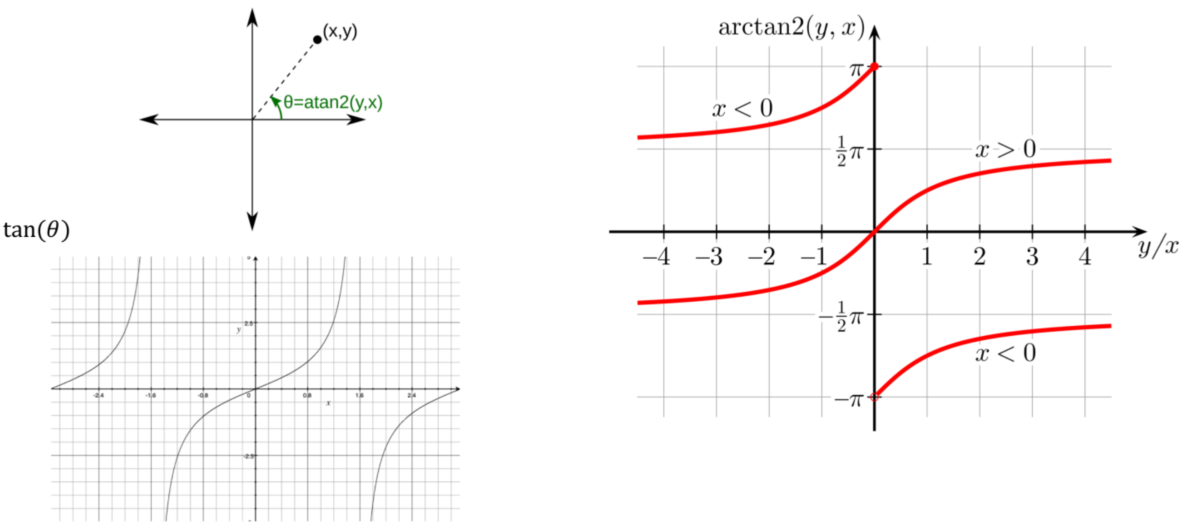

Figure 9: atan2 - definition

2.2 Angle and axis representation

- **Non0minimal representation** $R(\vec{r},\ \nu)$
- Unit vector $\vec{r} = (r_x,\ r_y,\ r_z)$
- Rotation angle ν about $\vec{r}$

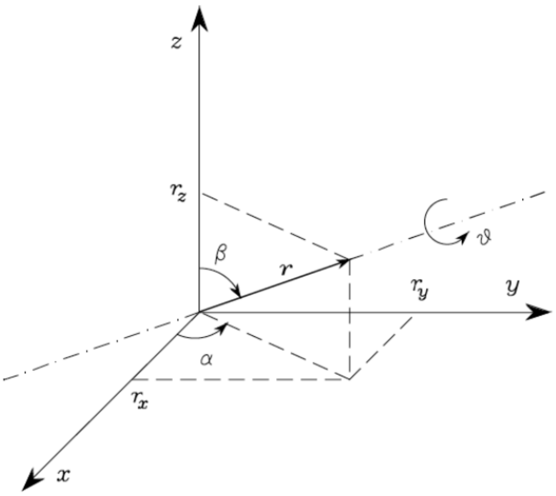

Figure 10: axis-angle representation

Matrix representation:

1. Alight $\vec{r}$ with $\vec{z} \to R_y(-\beta)R_z(-\alpha)$
2. Apply $\nu \to R_z(\nu)$
3. Re-aligh with $\vec{r} \to R_z(\alpha)R_y(\beta)$

$$R(\nu, r) = R_z(\alpha)R_y(\beta)R_z(\nu)R_y(-\beta)R_z(-\alpha)$$

$$R(\nu, r) = \begin{bmatrix} r_x^2(1-C_\nu)+C_\nu & r_xr_y(1-C\nu)-r_zS)\nu & r_xr_z(1-C_\nu)+r_yS_\nu \\ r_xr_y(1-C_\nu)+r_zS_\nu & r_y^2(1-C_\nu)+C_\nu & r_yr_z(1-C_\nu)-r_xS\nu \\ r_xr_z(1-C_\nu)-r_yS_\nu & r_yr_z(1-C_\nu)+r_xS_\nu & r_z^2(1-C_\nu)+C_\nu \end{bmatrix}$$

Note: $R(-\nu, -r) = R(\nu, r)$ While $\nu = 0$ there is singularity, all R matrix is zero.

2.2.1 Inverse problem

$$\nu = \arccos\left(\frac{r_{11}+r_{22}+r_{33}+1}{2}\right)$$

$$r = \frac{1}{2S_\nu}\begin{bmatrix} r_{32}-r_{23} \\ r_{13}-r_{31} \\ r_{21}-r_{12} \end{bmatrix}, \quad for\ S_\nu \neq 0$$

for $\nu = 0$ vertor $\vec{r}$ is arbitrary (singularity).

2.3 Quaternions

- Less intuitive than Euler/axis-angle
- Unique inverse
- No gimbal lock
- Fast, stable implementation

2.3.1 2D Rotation and Complex Numbers

- Comlex number is a tuple: $a + bi$
- Where: $i^2 = -1$
- Addition: $(a + bi) + (c + di) = (a + c) + (b + d)i$
- Multiply: $(a + bi)(c + di) = ac + adi + bci + bd\overset{-1}{\cancel{i^2}} = (ac - bd) + (ad + bc)i$
- Euler formula: $e^{i\theta} = (\cos\theta + i\sin\theta)$
  rotation by θ:
  $e^{i\theta}(x + iy) = (C_\theta + iS_\theta)(x + yi) = (xC_\theta - yS_\theta) + i(xS_\theta - yC_\theta)$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} C_\theta & -S_\theta \\ S_\theta & C_\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

  Complex multiplication=rotation(!)

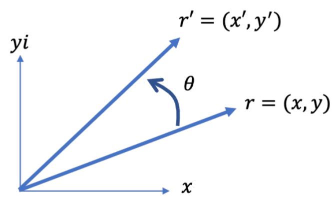

Figure 11: complex multiplication

2.3.2 Quaternions

- Quaternion is a 4-tuple $q_0 + q_1 i + q_2 j + q_3 k$
- Where:

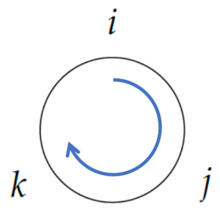

Figure 12: quaternion multiplication order

$$i^2 = j^2 + k^2 = -1$$
$$ij = k, \;\; ji = -k$$
$$jk = i, \;\; kj = -i$$
$$ki = j, \;\; ik = -j$$

- Addition:

$$(q_0 + q_1 i + q_2 j + q_3 k) + (p_0 + p_1 i + p_2 j + p_3 k) =$$
$$= (q_0 + p_0) + (q_1 + p_1)i + (q_2 + p_2)j + (q_3 + p_3)k$$

- Multiplication:

$$\begin{gathered}(q_0 + q_1 i + q_2 j + q_3 k)(p_0 + p_1 i + p_2 j + p_3 k) = \\ = (q_0 p_0 + q_1 p_1 i^2 + q_2 p_2 j^2 + q_3 p_3 k^2)+ \\ (q_0 p_1 i + q_1 p_0 i + q_2 p_3 jk + q_3 p_2 kj)+ \\ (q_0 p_2 j + q_2 p_0 j + q_1 p_3 ik + q_3 p_1 ki) \\ (q_0 p_3 k + q_3 p_0 k + q_1 p_2 ij + q_2 p_1 ji) = \\ = (q_0 p_0 - q_1 p_1 + q_2 p_2 + q_3 p_3)+ \\ (q_0 p_1 + q_1 p_0 + q_2 p_3 - q_3 p_2)i+ \\ (q_0 p_2 + q_2 p_0 - q_1 p_3 + q_3 p_1)j \\ (q_0 p_3 + q_3 p_0 + q_1 p_2 - q_2 p_1)k\end{gathered}$$

- Quaternion Conjugate:

$$q = q_0 + Q_1 i + q_2 j + q_3 k$$
$$q* = q_0 - Q_1 - +q_2 j - q_3 k$$

- Quaternion Norm:
  $|q| = \sqrt{q_0^2 + q_1^2 + q_2^2 + q_3^2}$
- Quaternion inverse: $qq* = q_0^2 + q_1^2 + q_2^2 + q_3^2 \implies (Norm)^2$

$$q^-1 = \frac{q*}{|q|^2}$$

- Quaternions for rotations
  - Vector (x, y, z) is a pure quaternion: $0 + xi + yj + zk$
  - Rotation = Unit quaternion $q_R : |q_R| = 1$
  - Rotation from frame B to frame A:

$$q_b = q_R q_A q_R^*$$

  While q_R is a unit quaternion
  q_A q_B are pure quaternions.

$$\begin{gathered}q_R q_A q_R^* = (q_0 + q_1 i + q_2 j + q_3 k)(xi + yj + zk)(q_0 - q_1 i - q_2 j - q_3 k) = \\ (x(q_0^2 + q_1^2 - q_2^2 - q_3^2) + 2y(q_1 q_2 - q_0 q_3) + 2z(q_0 q_2 + q_1 q_3))i+ \\ (2x(q_0 q_3 + q_1 q_2) + y(q_0^2 - q_1^2 + q_2^2 - q_3^2) + 2z(q_2 q_3 - q_0 q_1))j+ \\ (2x(q_1 q_3 - q_0 q_2) + 2y(q_0 q_1 + q_2 q_3) + z(q_0^2 - q_1^2 - q_2^2 + q_3^2))k\end{gathered}$$

  - Matrix notation for $q_R q_A q_R^*$: $M \cdot (x, y, z)^T$:

$$M = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1 q_2 - q_0 q_3) & 2(q_0 q_2 + q_1 q_3) \\ 2(q_0 q_3 + q_1 q_2) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2 q_3 - q_0 q_1) \\ 2(q_1 q_3 - q_0 q_2) & 2(q_0 q_1 + q_2 q_3) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix}$$

  - Simplification of M matrix:

$$M = 2 \cdot \begin{bmatrix} q_0^2 + q_1^2 - 0.5 & q_1 q_2 - q_0 q_3 & q_0 q_2 + q_1 q_3 \\ q_0 q_3 + q_1 q_2 & q_0^2 + q_2^2 - 0.5 & q_2 q_3 - q_0 q_1 \\ q_1 q_3 - q_0 q_2 & q_0 q_1 + q_2 q_3 & q_0^2 + q_3^2 - 0.5 \end{bmatrix}$$

- Rotation Matrix to Quaternion:

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} = 2 \cdot \begin{bmatrix} q_0^2 + q_1^2 - 0.5 & q_1 q_2 - q_0 q_3 & q_0 q_2 + q_1 q_3 \\ q_0 q_3 + q_1 q_2 & q_0^2 + q_2^2 - 0.5 & q_2 q_3 - q_0 q_1 \\ q_1 q_3 - q_0 q_2 & q_0 q_1 + q_2 q_3 & q_0^2 + q_3^2 - 0.5 \end{bmatrix}$$

$$q_0 = \frac{1}{2}\sqrt{r_{11} + r_{22} + r_{33} + 1} \text{ (We will take the positive result only)}$$

$$\begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} sgn(r_{32} - r_{23})\sqrt{r_{11} - r_{22} - r_{33} + 1} \\ sgn(r_{13} - r_{31})\sqrt{r_{22} - r_{33} - r_{11} + 1} \\ sgn(r_{21} - r_{12})\sqrt{r_{33} - r_{11} - r_{22} + 1} \end{bmatrix} \text{, No singularity!}$$

- Axis/Angle to Quaternion
  - Scalar part: $q_0 = \cos(\nu/2)$
  - Vector port $(q_1, q_2, q_3) = \sin(\nu/2)\vec{r}$

$$q_1 = r_x \sin(\nu/2)$$

$$...$$

$$q_0^2 + q_1^2 + q_2^2 + q_3^2 = 1$$

  - Same quaternion for $(-\vec{r}, -\nu), (\vec{r}, \nu)$:**unique!**
  - $\nu \in [-\pi, \pi] : q_0 \geq 0$

2.3.3 Homogeneous Transformations

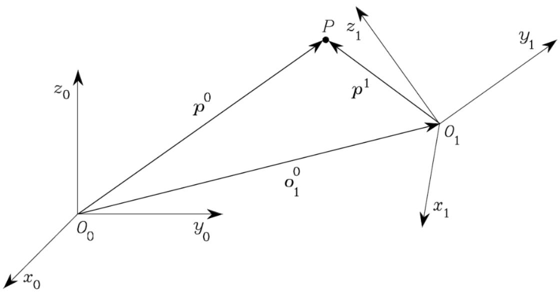

Figure 13: Rotation and translation

The normal way to do it is:

$$p^0 = o_1^0 + R_1^0 p^1$$
$$p^1 = -R_1^{0T} o_1^0 + R_1^{0T} p^0 \; ; \; R_1^{0T} = R_0^1$$
$$p^1 = -R_0^1 o_1^0 + R_0^1 p^0$$

As you can see it is not so comfortable, so the following trick is needed:
Homogeneous, we will define:

$$\tilde{p} = \begin{bmatrix} p\,[3 \times 1] \\ 1 \end{bmatrix} \; ; \; A_1^0 = \begin{bmatrix} R_1^0\,[3 \times 3] & o_1^0\,[3 \times 1] \\ 0^T\,[3 \times 1] & 1 \end{bmatrix}$$

so the transformation can be defined as:

$$\tilde{p}^0 = A_1^0 \tilde{p}^1$$

$$A_1^0\tilde{p}^1 = \begin{bmatrix} R_1^0 & o_1^0 \\ 0^T & 1 \end{bmatrix} \begin{bmatrix} p^1 \\ 1 \end{bmatrix} = \begin{bmatrix} (R_1^0 p^1 + o_1^0) \\ 1 \end{bmatrix} = \tilde{p}^0$$

So, all transformation is now very easy to define:

$$\tilde{p}^0 = A_1^0\tilde{p}^1 \Longleftrightarrow \tilde{p}^1 = A_0^1\tilde{p}^0 = (A_1^0)^{-1}\tilde{p}^0$$

very important to notice $A^-1 \neq A^T$
Using that trick we can now see a Sequence of transformations as the following:

$$\tilde{p}^0 = A_1^0 A_2^1 ... A_n^{n-1}\tilde{p}^n \tag{8}$$

A_i^{i-1} : homogeneuos transformation for point in frame i to the description of the same point in frame $i-1$

3 Camera Calibration and 3d reconstruct

Camera calibration is the process of estimating the parameters of camera, parameters and coefficients that determine an accurate relationship between **3D point** in the real world and its corresponding **2D projection** (pixel) in the image captured by that calibrated camera.
Two sets of parameters:

- **Intrinsic Parameters** - camera/lens system:
  - Focal length (x-axis/y-axis)
  - Optical center (center pixel, along the optical axis)
  - Radial distortion coefficients of the lens.
- **External parameters** - orientation (rotation and translation) of the camera with respect to some world coordinate system.

3.1 Intrinsic parameters

$$K = \begin{bmatrix} f_x & \gamma & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

f_x, f_y - x, y focal lengths (typically the same), in pixels or mm
c_x, c_y - x, y coordinates of the optical axis in the image plane (pixels)
γ - skew between the axes, usually 0.

4 DH Recipe

The DH operating recipe is as follows:

1. Find and number consecutively the joint axes; set the directions of axes z_0, ...,z_{n-1}.
2. Choose frame 0 by locating the origin on axis z_0; axes x_0 and y_0 are chosen so as to obtain a right-handed frame. If feasible, it is worth choosing frame 0 to coincide with the base frame.

   Execute steps 3-5 for $n = 1, \dots, n-1$
3. In order to locate the origin O_i:
   (a) If axes z_i and z_{i-1} are parallel:
      i. If joint i is revolute, locate O_i so that $d_i = 0$
      ii. If joint i is prismatic, locate O_i t a reference position for the joint range, e.g., a mechanical limit.

   (b) Otherwise, locate the origin O_i at the intersection of z_i with the common normal to axes z_{i-1} and z_i
4. Choose axis $x_i$along the common normal to axes z_{i-1} and z_i Pointing towards the end-effector.
5. Choose axis y_i so as to obtain a right-handed frame.
   To complete:
6. Choose frame n:
   (a) If joint n is revolute, align $z_n$with z_{n-1}
   (b) If joint n is prismatic, choose z_n arbitrarily.
   Axis x_n s set according to step 4.
7. For $i = 1, ..., n$ form the table of parameters $a_i, d_i, \alpha_i, \vartheta_i$
8. Compute the homogeneous transformation matrices A_{i-1}^{i} for $i = 1, ..., n$
9. Compute $T_0^n = A_1^0 A_2^1 ... A_n^{n-1}(q_n)$
10. Given T_0^b and T_e^n, compute the direct kinematics function $T_e^b(q) = T_0^b T_n^0(q) T_e^n$

a_i	Distance between O_i and $O_{i'}$
d_i	Coordinade of $O_{i'}$ along z_{i-1}
α_i	Angle between axes z_{i-1} and z_i about axis x_i to be taken positive when rotation is made counterclockwise (right handed)
ϑ_i	Angle between axes x_{i-1} and x_i about axis z_{i-1} to be taken positive when rotation is made counterclockwise (right handed)

* $O_{i'}$: the intersection of the common normal to z_i and z_{i-1} with z_{i-1}

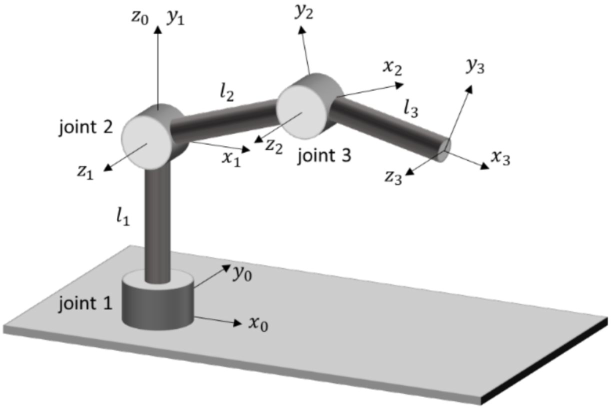

Figure 14: sample Robot

5 DH Helper

Link	a_i	α_i	d_i	ϑ_i
1	0	$\pi/2$	l_1	ϑ_1
2	l_1	0	0	ϑ_2
3	l_2	0	0	ϑ_3

Each line on the table is transformed to that matrix:

$$\begin{bmatrix} C\vartheta_i & -S\vartheta_i C\alpha_i & S\vartheta_i S\alpha_i & a_i C\vartheta_i \\ S\vartheta_i & C\vartheta_i C\alpha_i & -C\vartheta_i S\alpha_i & a_i S\vartheta_i \\ 0 & S\alpha_i & C\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

1. Link l_i is between O_{i-1} and O_i
2. The lines $_i$ on D-H table are per **link!**
3. Common normal is between z_{i-1} and z_i (right handed),