Spatial Descriptors and Transformations

Descriptions:
- **Position**
- **Orientation**
- **Frames**

A position vector P on a frame A can be represented as ${}^{A}P$

$$ {}^{A}P = \begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix} $$

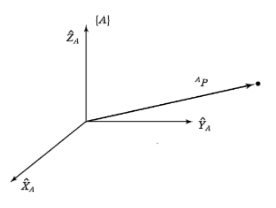
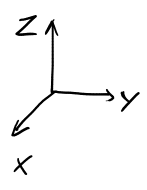
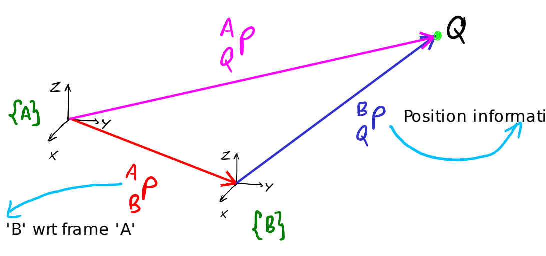


FIGURE 2.1: Vector relative to frame (example).



Mapping involving Translated Frames:



Position information of point 'Q' wrt frame 'B'

Position information of frame 'B' wrt frame 'A'

* There are two frames {A} and {B} each consit of three orthonormal vectors 'X', 'Y' and 'Z'
* There is a point Q in 3D space, which can be described from frame {B} as a positional vector of point 'Q' wrt 'B' ${}^{B}_{Q}P$
* The positional information of {B} wrt {A} can be represented as ${}^{A}_{B}P$

Now, the positional information of 'Q' wrt {A} can be represented as ${}^{A}_{Q}P = {}^{B}_{Q}P + {}^{A}_{B}P$

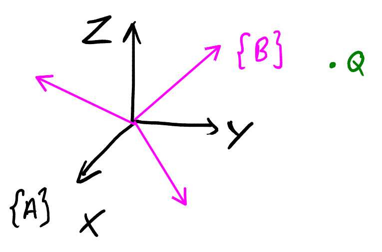
Description of Orientation

Orientation describes how the axes (XYZ) of one frame is aligned wrt other frame (XYZ)

$$ {}^{A}_{B}R = \left[\, {}^{A}\hat{X}_B \;\; {}^{A}\hat{Y}_B \;\; {}^{A}\hat{Z}_B \,\right] = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} $$

$$ {}^{A}_{B}R = \left[\, {}^{A}\hat{X}_B \;\; {}^{A}\hat{Y}_B \;\; {}^{A}\hat{Z}_B \,\right] = \begin{bmatrix} \hat{X}_B \cdot \hat{X}_A & \hat{Y}_B \cdot \hat{X}_A & \hat{Z}_B \cdot \hat{X}_A \\ \hat{X}_B \cdot \hat{Y}_A & \hat{Y}_B \cdot \hat{Y}_A & \hat{Z}_B \cdot \hat{Y}_A \\ \hat{X}_B \cdot \hat{Z}_A & \hat{Y}_B \cdot \hat{Z}_A & \hat{Z}_B \cdot \hat{Z}_A \end{bmatrix} $$

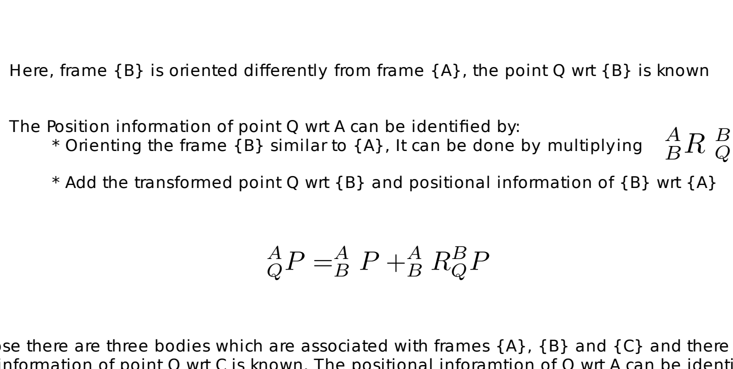
Mapping involving Rotated Frame:



* Two frames {A} and {B} having no translation but oriented differently
* The positional information of point 'Q' wrt 'B' is known, the poisitonal info of 'Q' wrt 'A' can be represented as

${}^{A}_{Q}P = {}^{A}_{B}R\,{}^{B}_{Q}P$ where, ${}^{A}_{B}R$ is the Rotation information of frame {B} wrt {A}

Mapping involving Translated Rotated Frames:



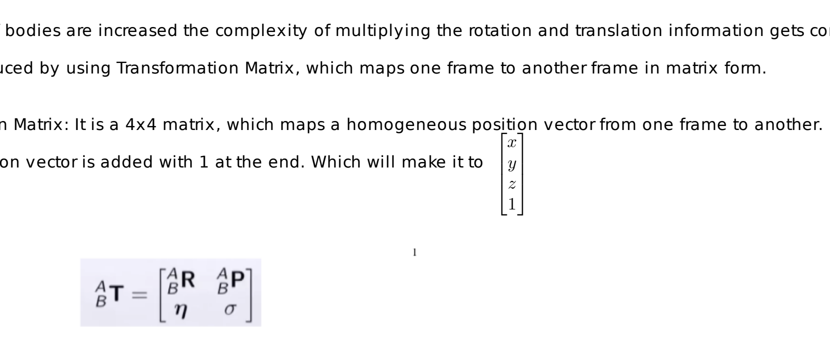
Here, frame {B} is oriented differently from frame {A}, the point Q wrt {B} is known

The Position information of point Q wrt A can be identified by:
* Orienting the frame {B} similar to {A}, It can be done by multiplying ${}^{A}_{B}R\;{}^{B}_{Q}P$
* Add the transformed point Q wrt {B} and positional information of {B} wrt {A}

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

Exercise: Suppose there are three bodies which are associated with frames {A}, {B} and {C} and there is a point Q in space.
The positional information of point Q wrt C is known. The positional inforamtion of Q wrt A can be identified as

$$ {}^{A}_{Q}P = {}^{A}_{B}P + {}^{A}_{B}R\;{}^{B}_{C}P + {}^{A}_{B}R\;{}^{B}_{C}R\;{}^{C}_{Q}P $$



Note: Here, if the number of bodies are increased the complexity of multiplying the rotation and translation information gets complex.

This complexity can be reduced by using Transformation Matrix, which maps one frame to another frame in matrix form.

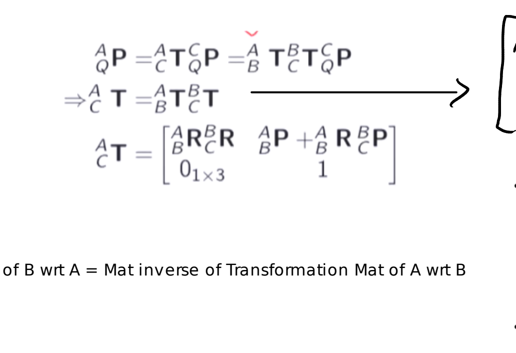
Homogeneous Transformation Matrix: It is a 4x4 matrix, which maps a homogeneous position vector from one frame to another.

Where, homogeneous position vector is added with 1 at the end. Which will make it to $\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$

$$ {}^{A}_{B}\mathbf{T} = \begin{bmatrix} {}^{A}_{B}\mathbf{R} & {}^{A}_{B}\mathbf{P} \\ \eta & \sigma \end{bmatrix} $$

Where,
η *is the Prespective vector*, the prespective vector is used if the transformation is non-orthogonal otherwise [0, 0, 0] for orthogonal projection
σ *is the scaling factor, normally it will be set to* 1

Compound Transformation:



$$ {}^{A}_{Q}\mathbf{P} = {}^{A}_{C}\mathbf{T}\,{}^{C}_{Q}\mathbf{P} = {}^{A}_{B}\mathbf{T}\,{}^{B}_{C}\mathbf{T}\,{}^{C}_{Q}\mathbf{P} $$
$$ \Rightarrow {}^{A}_{C}\mathbf{T} = {}^{A}_{B}\mathbf{T}\,{}^{B}_{C}\mathbf{T} \longrightarrow \begin{bmatrix} {}^{A}_{B}R & {}^{A}_{B}P \\ 0 & 1 \end{bmatrix} \begin{bmatrix} {}^{B}_{C}R & {}^{B}_{C}P \\ 0 & 1 \end{bmatrix} $$
$$ {}^{A}_{C}\mathbf{T} = \begin{bmatrix} {}^{A}_{B}\mathbf{R}\,{}^{B}_{C}\mathbf{R} & {}^{A}_{B}\mathbf{P} + {}^{A}_{B}\mathbf{R}\,{}^{B}_{C}\mathbf{P} \\ 0_{1\times 3} & 1 \end{bmatrix} $$

- **SO(3) (Special Orthogonal Group in 3D):**
  - Refers to the group of **3D rotations** about the origin.
  - It consists of all **orthogonal 3×3 rotation matrices** R with determinant $\det(R) = 1$:

    $$ SO(3) = \{R \in \mathbb{R}^{3\times 3} \mid R^T R = I,\ \det(R) = 1\} $$
  - Represents pure rotation without any translation.
- **SE(3) (Special Euclidean Group in 3D):**
  - Refers to the group of **rigid transformations** (rotation + translation) in 3D space.
  - It combines SO(3) for rotation and $\mathbb{R}^3$ for translation:

    $$ SE(3) = \left\{ \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \mid R \in SO(3), t \in \mathbb{R}^3 \right\} $$
  - Encodes both the orientation (rotation) and the position (translation) of an object.

Properties of Transformation Matrix:

Matrix Inverse: Transformation Mat of B wrt A = Mat inverse of Transformation Mat of A wrt B

$$ {}^{A}_{B}T = {}^{B}_{A}T^{-1} $$

Euler Rotation (X, Y, Z):
To Transform the orientation from Frame {A} to Frame {B}, perform the below three steps:
- Rotate {A} about X axis to an angle γ = $\{A^1\}$
- Rotate $\{A^1\}$ about Y axis to an angle β = $\{A^2\}$
- Rotate $\{A^2\}$ about Z axis to an angle α [Equation] = {B}

$$ {}^{A}_{B}R = {}^{A2}_{B}R_Z(\alpha)\ \ {}^{A1}_{A2}R_Y(\beta)\ \ {}^{A}_{A1}R_X(\gamma) $$

$$ {}^{A}_{B}\mathbf{R} = R_Z(\alpha)\; R_Y(\beta)\; R_X(\gamma) \qquad + $$
$$ = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{bmatrix} $$

$$ {}^{A}_{B}\mathbf{R} = \begin{bmatrix} \cos\alpha\cos\beta & \cos\alpha\sin\beta\sin\gamma - \sin\alpha\cos\gamma & \cos\alpha\sin\beta\sin\gamma + \sin\alpha\sin\gamma \\ \sin\alpha\cos\beta & \sin\alpha\sin\beta\sin\gamma + \cos\alpha\cos\gamma & \sin\alpha\sin\beta\cos\gamma - \cos\alpha\sin\gamma \\ -\sin\beta & \cos\beta\sin\gamma & \cos\beta\cos\gamma \end{bmatrix} $$
$$ = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} $$

$$ \beta = \operatorname{atan2}\left(-r_{31}, \sqrt{r_{11}^2 + r_{21}^2}\right) $$
$$ \alpha = \operatorname{atan2}\left(\frac{r_{21}}{\cos\beta}, \frac{r_{11}}{\cos\beta}\right) $$
$$ \gamma = \operatorname{atan2}\left(\frac{r_{32}}{\cos\beta}, \frac{r_{33}}{\cos\beta}\right) $$

Derivative of a Rotation Matrix (Use of Skew Symmetric Matrix):

Derivative of Rotation matrix

$$ \mathbf{R}\mathbf{R}^T = \mathbf{I} $$

Differentiating w.r.t. time

$$ \dot{\mathbf{R}}\mathbf{R}^T + \mathbf{R}\dot{\mathbf{R}}^T = \mathbf{0} $$
$$ \dot{R}R^T + (\dot{R}R^T)^T = 0 $$
$$ S + S^T = 0 $$
$$ \boxed{S = \dot{R}R^T} $$

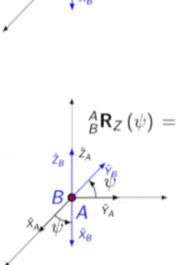
Skew symmetric matrix

$$ S^T = -S $$
$$ S + S^T = 0 $$
$$ \dot{R} = S(R^T)^{-1} $$
$$ \dot{R}R^T = S $$
$$ \dot{R} = S(R^T)^{-1} $$
$$ \dot{R} = SR $$

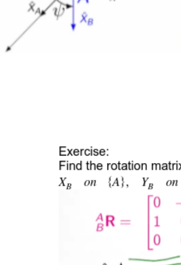
Derivation of Rotation Matrices:

$$ {}^{A}_{B}R = \left[\, {}^{A}\hat{X}_B \;\; {}^{A}\hat{Y}_B \;\; {}^{A}\hat{Z}_B \,\right] = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} $$

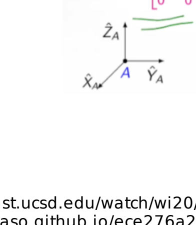
$$ {}^{A}_{B}\mathbf{R}_X(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix} $$



$$ {}^{A}_{B}\mathbf{R}_Y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} $$



$$ {}^{A}_{B}\mathbf{R}_Z(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} $$



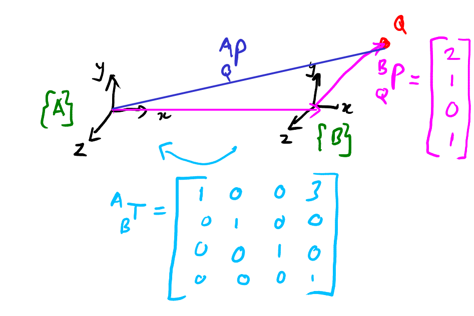
Exercise:
Find the rotation matrix between {B} and {A} => Project
X_B *on* $\{A\}$, Y_B *on* $\{A\}$, Z_B *on* $\{A\}$

$$ {}^{A}_{B}\mathbf{R} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} $$

Lecture to check:
https://podcast.ucsd.edu/watch/wi20/ece276a_a00/13/audio ## UC SanDiego Lecture on Robotics
https://natanaso.github.io/ece276a2020/schedule.html

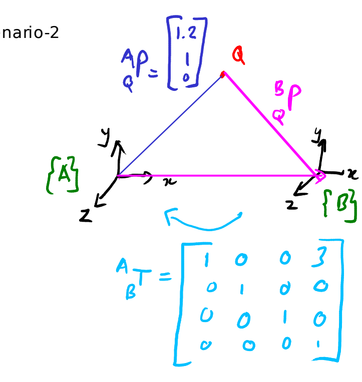
Some examples of Transformation matrices:

Scenario-1




```
In [28]: T_AA
Out[28]:
array([[1., 0., 0., 0.],
       [0., 1., 0., 0.],
       [0., 0., 1., 0.],
       [0., 0., 0., 1.]])

In [29]: T_BA
Out[29]:
array([[1., 0., 0., 3.],
       [0., 1., 0., 0.],
       [0., 0., 1., 0.],
       [0., 0., 0., 1.]])

In [30]: P_QB
Out[30]: array([2, 1, 0, 1])

In [31]: P_QA = T_BA @ P_QB

In [32]: P_QA
Out[32]: array([5., 1., 0., 1.])
```

Scenario-2




```
In [33]: T_AA
Out[33]:
array([[1., 0., 0., 0.],
       [0., 1., 0., 0.],
       [0., 0., 1., 0.],
       [0., 0., 0., 1.]])

In [34]: T_BA
Out[34]:
array([[1., 0., 0., 3.],
       [0., 1., 0., 0.],
       [0., 0., 1., 0.],
       [0., 0., 0., 1.]])

In [35]: P_QA = np.array([1.2, 1, 0, 1])

In [36]: P_QB = np.linalg.inv(T_BA) @ P_QA

In [37]: P_QB
Out[37]: array([-1.8,  1. ,  0. ,  1. ])
```

There are two frames A and B, Frame B is translated by 3 units in X direction (no rotation) Consider a point Q represented in B frame as P_QB. Now the point P_QB can be written in A frame by

$$^{A}_{Q}P = {}^{A}_{B}T \; {}^{B}_{Q}P$$

Consider the second scenario, it is same as scenario-1 except the position is represented wrt A frame P_QA. Now the point P_QB can be written as

$$^{B}_{Q}P = \left({}^{A}_{B}T\right)^{-1} {}^{A}_{Q}P = {}^{B}_{A}T \; {}^{A}_{Q}P$$

Read both of them from right to left. Start from Q traverse till A (scenario-1) and B(scenario-2) if the rotation direction changes perfrom transformation inverse (in case of scenario-2)

Additional information related to rotation (Group Theory Aspect)

1. 🔄 Complex Numbers and U(1) Rotation

Concept:

- Complex numbers can rotate vectors in 2D.
- The **unit circle** in the complex plane is the group **U(1)**.

Definition:

U(1) = { z ∈ ℂ : |z| = 1 }

These are complex numbers of the form:

$$z = e^{i\theta} = \cos(\theta) + i\sin(\theta)$$

Action:

Multiplying any complex number/vector $v \in \mathbb{C}$ by $z \in U(1)$ rotates it by angle θ.

📌 **Example:**

Let's rotate the point $v = 1 + 0i$ by $\theta = 90^\circ = \frac{\pi}{2}$

$$z = e^{i\pi/2} = \cos(\pi/2) + i\sin(\pi/2) = i$$

$$zv = i \cdot (1 + 0i) = i$$

So (1,0) rotated by 90° becomes (0,1)

✅ Example 2: 180° Rotation

- Vector: $v = 2 + 3i$
- $\theta = 180^\circ = \pi$
- $z = e^{i\pi} = -1$

$$v' = -1 \cdot (2 + 3i) = -2 - 3i$$

✅ It's a **flip through the origin**, i.e., rotated 180°.

✅ Example 3: 45° Rotation

- Vector: $v = 1 + 0i$
- $\theta = 45^\circ = \frac{\pi}{4}$
- $z = e^{i\pi/4} = \cos(\pi/4) + i\sin(\pi/4) = \frac{\sqrt{2}}{2} + i\frac{\sqrt{2}}{2}$

$$v' = z \cdot v = \left(\frac{\sqrt{2}}{2} + i\frac{\sqrt{2}}{2}\right)(1 + 0i) = \frac{\sqrt{2}}{2} + i\frac{\sqrt{2}}{2}$$

✅ This lies at 45° on the unit circle.

2. 🔢 U(2) – Unitary 2×2 Matrices

Concept:

U(2) is the group of **2×2 complex matrices** U such that:

$$U^\dagger U = I$$

Where:

- $U^\dagger$: Conjugate transpose of U
- I: Identity matrix

These matrices preserve **complex inner products** (important in quantum mechanics).

🔎 **Example:**

$$U = \begin{bmatrix} e^{i\phi} & 0 \\ 0 & e^{i\psi} \end{bmatrix} \in U(2)$$

Check:

$$U^\dagger = \begin{bmatrix} e^{-i\phi} & 0 \\ 0 & e^{-i\psi} \end{bmatrix}, \quad U^\dagger U = I$$

3. 🔁 SO(2) – Rotations in 2D (Real Space)

Definition:

SO(2) = group of **real 2×2 orthogonal matrices with determinant = 1**

These are **2D rotation matrices**:

$$R(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

Connection to U(1):

- $R(\theta)$ acts on real vectors.
- $e^{i\theta}$ (U(1)) acts similarly on complex numbers.

$$\mathrm{SO}(2) \cong \mathrm{U}(1) \quad \text{(isomorphic)}$$

📌 **Example:**

Rotate vector $v = [1, 0]^T$ by 45°:

$$R\left(\frac{\pi}{4}\right) = \begin{bmatrix} \cos\frac{\pi}{4} & -\sin\frac{\pi}{4} \\ \sin\frac{\pi}{4} & \cos\frac{\pi}{4} \end{bmatrix} = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}$$

$$Rv = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

🔁 2. Unitary Matrix Intuition

Geometric Intuition:

Just like **real orthogonal matrices** preserve vector lengths and angles in ℝ², **unitary matrices** preserve them in ℂ².

- **U(1)**: rotates 1D complex vectors → **circle**
- **U(2)**: rotates 2D complex vectors → **4D space**

🔣 3. Properties of U(2)

- Complex entries
- Preserves complex inner product

$$\langle Ux, Uy \rangle = \langle x, y \rangle$$

- Determinant satisfies:

$$|\det(U)| = 1$$

🧮 4. Numeric Example of U(2)

Let's construct a simple 2×2 unitary matrix:

Let:

$$U = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & i \\ i & 1 \end{bmatrix}$$

Let's verify whether this is **unitary**.

Step 1: Compute $U^\dagger$

Conjugate transpose:

$$U^\dagger = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & -i \\ -i & 1 \end{bmatrix}$$

Step 2: Check $U^\dagger U = I$

$$U^\dagger U = \left(\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & -i \\ -i & 1 \end{bmatrix}\right)\left(\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & i \\ i & 1 \end{bmatrix}\right) = \frac{1}{2}\begin{bmatrix} 1\cdot 1 + (-i)(i) & 1\cdot i + (-i)(1) \\ (-i)(1) + 1\cdot i & (-i)(i) + 1\cdot 1 \end{bmatrix}$$

$$= \frac{1}{2}\begin{bmatrix} 1+1 & i-i \\ -i+i & 1+1 \end{bmatrix} = \frac{1}{2}\begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = I$$

✅ Confirmed: **U is unitary**, so $U \in U(2)$

🔁 5. Apply U(2) to a Vector

Let's apply U to a vector $v = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$:

$$Uv = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & i \\ i & 1 \end{bmatrix}\begin{bmatrix} 1 \\ 0 \end{bmatrix} = \frac{1}{\sqrt{2}}\begin{bmatrix} 1\cdot 1 + i\cdot 0 \\ i\cdot 1 + 1\cdot 0 \end{bmatrix} = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 \\ i \end{bmatrix}$$

✅ The output is a **rotated + scaled** complex vector (unit length preserved).

🧠 6. General Form of U(2)

Any matrix in U(2) can be written as:

$$U = e^{i\phi}\begin{bmatrix} a & b \\ -\bar{b} & \bar{a} \end{bmatrix} \quad \text{with} \quad |a|^2 + |b|^2 = 1$$

This has **4 real parameters**:

- 1 global phase ϕ
- 3 real parameters from $a, b \in \mathbb{C}$

✅ **Special Case: SU(2)**

If $\det(U) = 1$, the matrix is in **SU(2)** (Special Unitary group). SU(2) is closely related to:

- 3D rotations (covers SO(3) twice)
- Quantum spin systems
- Quaternions

🔷 What is SU(2)?

SU(2) = Set of **2×2 unitary matrices with determinant 1**

A matrix $U \in \mathrm{SU}(2)$ satisfies:

1. $U^\dagger U = I$ (unitary)
2. $\det(U) = 1$

🧮 General Form:

$$U = \begin{bmatrix} a & b \\ -\bar{b} & \bar{a} \end{bmatrix}, \quad \text{with } |a|^2 + |b|^2 = 1$$

This gives SU(2) **3 real degrees of freedom**.

🔁 Connection to Rotations (SO(3)):

- SU(2) describes **3D rotations**, like SO(3), but is a **double cover**: One rotation in SO(3) ↔ two matrices in SU(2)
- SU(2) is **isomorphic to unit quaternions**

$$q = \cos(\theta/2) + \sin(\theta/2)(n_x i + n_y j + n_z k)$$

✅ Example:

Let $a = \cos(\theta/2),\ b = i\sin(\theta/2)$

Then:

$$U = \begin{bmatrix} \cos(\theta/2) & i\sin(\theta/2) \\ i\sin(\theta/2) & \cos(\theta/2) \end{bmatrix} \in \mathrm{SU}(2)$$

⚛️ Application:

- SU(2) rotates **qubit states** in quantum mechanics (e.g. Bloch sphere rotations)
- Used in **quantum gates, spin systems, quaternion math**

Let me know if you want the SU(2) → SO(3) mapping or code examples.

Absolutely! Let's build a **concrete example** of an **SU(2) matrix**, and then show its correspondence to both a **3D rotation in SO(3)** and a **unit quaternion**.

🔷 Step 1: Pick a Rotation

Let's rotate by **θ = 120° (2π/3 radians)** around the axis

$$\hat{n} = \frac{1}{\sqrt{3}}(1, 1, 1)$$

This is a **standard axis** for testing SU(2)-SO(3)-quaternion correspondence.

🧮 Step 2: Build SU(2) Matrix

General form of an SU(2) matrix for rotation by angle θ around axis $\hat{n} = (n_x, n_y, n_z)$:

$$U = \cos\left(\frac{\theta}{2}\right) I + i\sin\left(\frac{\theta}{2}\right)(n_x\sigma_x + n_y\sigma_y + n_z\sigma_z)$$

Where:

- $\sigma_x, \sigma_y, \sigma_z$ are **Pauli matrices**:

$$\sigma_x = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad \sigma_y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}, \quad \sigma_z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

Now plug in:

- $\theta = \frac{2\pi}{3} \Rightarrow \frac{\theta}{2} = \frac{\pi}{3}$
- $\cos\left(\frac{\pi}{3}\right) = \frac{1}{2}$
- $\sin\left(\frac{\pi}{3}\right) = \frac{\sqrt{3}}{2}$
- $\hat{n} = \frac{1}{\sqrt{3}}(1, 1, 1)$

Then:

$$U = \frac{1}{2}I + i\cdot\frac{\sqrt{3}}{2}\cdot\frac{1}{\sqrt{3}}(\sigma_x + \sigma_y + \sigma_z) = \frac{1}{2}I + \frac{i}{2}(\sigma_x + \sigma_y + \sigma_z)$$

Now compute:

$$\sigma_x + \sigma_y + \sigma_z = \begin{bmatrix} 1 & 1-i \\ 1+i & -1 \end{bmatrix}$$

So:

$$U = \frac{1}{2}\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \frac{i}{2}\begin{bmatrix} 1 & 1-i \\ 1+i & -1 \end{bmatrix} = \begin{bmatrix} \frac{1}{2} + \frac{i}{2} & \frac{i}{2}(1-i) \\ \frac{i}{2}(1+i) & \frac{1}{2} - \frac{i}{2} \end{bmatrix}$$

Simplify:

- $i(1-i) = i - i^2 = i + 1$
- $i(1+i) = i + i^2 = i - 1$

Final matrix:

$$U = \begin{bmatrix} \frac{1}{2} + \frac{i}{2} & \frac{1+i}{2} \\ \frac{-1+i}{2} & \frac{1}{2} - \frac{i}{2} \end{bmatrix}$$

✅ This matrix is in **SU(2)**.

🔁 Step 3: Convert SU(2) to Quaternion

Any SU(2) matrix corresponds to a **unit quaternion**:

$$q = \cos\left(\frac{\theta}{2}\right) + \sin\left(\frac{\theta}{2}\right)(n_x i + n_y j + n_z k)$$

Using same parameters as above:

- $\theta/2 = \pi/3$
- $\cos(\pi/3) = 0.5, \sin(\pi/3) = \frac{\sqrt{3}}{2}$
- $\hat{n} = \frac{1}{\sqrt{3}}(1, 1, 1)$

$$q = 0.5 + \frac{\sqrt{3}}{2\sqrt{3}}(i + j + k) = 0.5 + 0.5i + 0.5j + 0.5k$$

✅ This is the **unit quaternion** corresponding to the same 3D rotation.

🧭 Step 4: Convert Quaternion to SO(3)

To get the 3D rotation matrix (SO(3)) from a unit quaternion:

$$q = w + xi + yj + zk = 0.5 + 0.5i + 0.5j + 0.5k$$

Use this formula:

$$R = \begin{bmatrix} 1 - 2(y^2 + z^2) & 2(xy - wz) & 2(xz + wy) \\ 2(xy + wz) & 1 - 2(x^2 + z^2) & 2(yz - wx) \\ 2(xz - wy) & 2(yz + wx) & 1 - 2(x^2 + y^2) \end{bmatrix}$$

Plug in:

- $w = 0.5, x = 0.5, y = 0.5, z = 0.5$

Compute:

$$R = \begin{bmatrix} 1 - 2(0.25 + 0.25) & 2(0.25 - 0.25) & 2(0.25 + 0.25) \\ 2(0.25 + 0.25) & 1 - 2(0.25 + 0.25) & 2(0.25 - 0.25) \\ 2(0.25 - 0.25) & 2(0.25 + 0.25) & 1 - 2(0.25 + 0.25) \end{bmatrix}$$

$$R = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

✅ This is a **120° rotation about the axis (1,1,1) — same as SU(2 and quaternion!**

🔗 Summary of the Correspondence

Description	Object
Rotation	120° about $\hat{n} = \frac{1}{\sqrt{3}}(1, 1, 1)$
SU(2) Matrix	Complex 2×2 matrix (above)
Quaternion	$q = 0.5 + 0.5i + 0.5j + 0.5k$
SO(3) Matrix	$\begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$

✅ All three represent **the same 3D rotation** in different mathematical languages:

- **SU(2)** for quantum spin/qubit
- **Quaternion** for efficient 3D math
- **SO(3)** for matrix-based rotation

3. 🔁 SO(2) – Rotations in 2D (Real Space)

Definition:

SO(2) = group of **real 2×2 orthogonal matrices with determinant = 1**

These are **2D rotation matrices**:

$$R(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

Connection to U(1):

- $R(\theta)$ acts on real vectors.
- $e^{i\theta}$ (U(1)) acts similarly on complex numbers.

$$\mathrm{SO}(2) \cong \mathrm{U}(1) \quad \text{(isomorphic)}$$

📌 **Example:**

Rotate vector $v = [1, 0]^T$ by 45°:

$$R\left(\frac{\pi}{4}\right) = \begin{bmatrix} \cos\frac{\pi}{4} & -\sin\frac{\pi}{4} \\ \sin\frac{\pi}{4} & \cos\frac{\pi}{4} \end{bmatrix} = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}$$

$$Rv = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

4. 🌐 SO(3) – Rotations in 3D

Definition:

SO(3) = real 3×3 orthogonal matrices with determinant = 1

These represent **rotations in 3D space**.

A generic rotation about the z-axis:

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

- SO(3) is **non-abelian** (order matters!)
- Cannot be written as simply as U(1) or SO(2)

5. ⚙️ Quaternions and Rotations

Quaternion Basics:

A quaternion:

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

Where:

- $a, b, c, d \in \mathbb{R}$
- $i^2 = j^2 = k^2 = ijk = -1$

Unit Quaternions:

$$|q| = \sqrt{a^2 + b^2 + c^2 + d^2} = 1 \Rightarrow \text{Unit Quaternion}$$

Used to represent **rotations in 3D** without gimbal lock.

Rotation using Quaternions:

To rotate a vector $\vec{v} \in \mathbb{R}^3$ using quaternion q:

1. Convert $\vec{v}$ to pure quaternion: $v = 0 + v_x i + v_y j + v_z k$
2. Use:

$$v' = qvq^{-1}$$

Where $q^{-1} = \bar{q}$ if $|q| = 1$

📌 **Numeric Example:**

Rotate vector $\vec{v} = [1, 0, 0]$ by 180° around the z-axis.

1. Axis: $\vec{n} = [0, 0, 1]$
2. Angle: $\theta = \pi$

Quaternion:

$$q = \cos(\pi/2) + \sin(\pi/2)(0i + 0j + 1k) = 0 + 0i + 0j + 1k = k$$

Vector $v = 0 + 1i + 0j + 0k$

Then:

$$v' = qvq^{-1} = k(i)(-k) = -i \Rightarrow \vec{v}' = [-1, 0, 0]$$

Rotated by 180°!

6. 🔗 Summary of Connections

Group	Dim	Acts on	Structure	Notes
U(1)	1D	ℂ (1D complex)	Abelian	Rotations in complex plane
SO(2)	2D	ℝ²	Abelian	Same as U(1), real-valued
U(2)	4D	ℂ²	Non-Abelian	Quantum systems
SO(3)	3D	ℝ³	Non-Abelian	3D rotations
Quaternions	4D	ℝ³ via action	Non-Abelian	Efficient rotation representation

🔷 What Is SO(3)?

- **SO(3)** = Special Orthogonal Group in 3D
- It consists of all **3×3 rotation matrices** R that:
  - Preserve length and angles: $R^T R = I$ (orthogonal)
  - Have $\det(R) = +1$ (no reflection)

Each element in SO(3) represents a **rotation in 3D space**.

🧭 Axis-Angle Representation of SO(3)

💡 **Key Idea:**

Every rotation in 3D can be described by:

- A **unit vector** $\hat{n} \in \mathbb{R}^3$: the **axis** of rotation
- An **angle** $\theta \in [0, \pi]$: how much to rotate **around** that axis

This is called the **axis-angle representation**.

🔴 Representing SO(3) as a Solid Ball

We can represent SO(3) as points **inside a solid 3D ball of radius** π.

🎯 **How?**

We map each rotation to a vector:

$$\vec{v} = \theta\hat{n} \quad \text{where } \theta \in [0, \pi]$$

So:

- The **direction** of $\vec{v}$ gives the rotation axis $\hat{n}$
- The **length** $\|\vec{v}\|$ gives the angle of rotation θ

🧠 What the Ball Represents:

- The **interior** of the ball (not including surface): all rotations with angle $\theta < \pi$
- The **surface** of the ball: rotations with angle $\theta = \pi$
- The **center (0,0,0)**: the **identity rotation** (no rotation at all)

🧭 Why Radius π?

Because:

- Rotating by θ is the same as rotating by $-\theta$ around $-\hat{n}$
- The **maximum unique angle** of rotation is π (180°)
- After $\theta = \pi$, rotations start repeating (you come back the "long way")

🔁 Important Topology Note

- Points **on opposite sides of the surface** of the ball represent the **same rotation**.
- So the ball has a **non-trivial topology**: It's a **3D ball with antipodal points on the surface identified** — this is called **projective 3-space**, or $\mathbb{RP}^3$

🔄 What's Rotated About What?

Let's clarify that:

- You're rotating a **vector** $\vec{v}$ in 3D space
- You rotate it **around** a fixed axis $\hat{n}$
- By an angle θ
- This defines a rotation matrix $R \in \mathrm{SO}(3)$
- You apply the rotation: $\vec{v}_{\text{rotated}} = R \cdot \vec{v}$

✅ Example

Let's rotate around axis $\hat{n} = (0, 0, 1)$ by $\theta = \frac{\pi}{2}$ (90°)

Then the axis-angle vector is:

$$\vec{v} = \theta\hat{n} = \frac{\pi}{2}(0, 0, 1) = \left(0, 0, \frac{\pi}{2}\right)$$

This lies **halfway** from the origin to the surface of the SO(3) ball, along the z-axis.

The corresponding rotation matrix is:

$$R = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

This rotates vectors 90° around the z-axis.

🧠 Summary

Term	Meaning
Axis $\hat{n}$	Direction around which rotation happens
Angle θ	How much to rotate
Vector $\theta\hat{n}$	Represents the rotation in a solid ball of radius π
SO(3)	All such rotations: mapped to points in the ball
Radius Range	0 (identity) to π (180° rotation)
Opposite points on surface	Represent same rotation (e.g. $\theta = \pi$ around $\hat{n}$ and $-\hat{n}$)

🧠 Conceptual Setup: SO(3) = Set of 3D Rotations

Every element of SO(3) is a rotation in 3D space:

- Defined by a **rotation axis** (a unit vector $\vec{v} \in \mathbb{R}^3$)
- And a **rotation angle** $\theta \in [0, \pi]$

So a rotation can be uniquely described by a vector:

$$\vec{r} = \theta \cdot \vec{v}$$

That means SO(3) can be represented as:

> All vectors inside a solid 3D ball of **radius** π (i.e., $\|\vec{r}\| \le \pi$)

Orthogonal Group O(3)

- O(3) consists of set of 3 ×3 orthogonal matrices (determinant +1 or -1)
- **Direct product with inversion group**

$$O(3) = SO(3) \otimes C_s$$

- Each element of SO(3) will be specified by three parameters

$$R_{\hat{n}}(\psi) = R(\psi, \theta, \phi)$$

Determinant of SO(3) = +1 (proper rotations)

Cs would consist of identity and diag(-1, -1, -1) elemnet
The group manifold of the parameter space of group SO(3) is a solid sphere of radisu pi which is compact

Special Orthogonal Group SO(3)

Rotation by angle ψ about the direction $\hat{\mathbf{n}} = (\theta, \phi)$:

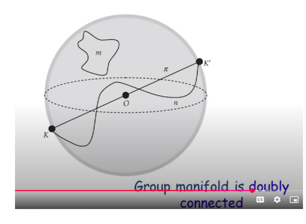
$R_{-\hat{n}}(\psi) = R_{\hat{n}}(-\psi)$ which implies $0 \le \psi \le \pi$

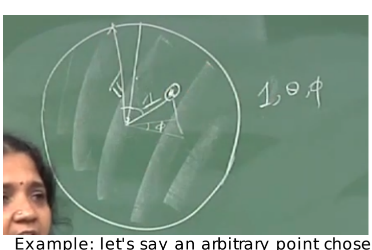
$R_{-\hat{n}}(\pi) = R_{\hat{n}}(\pi)$

Group manifold is a sphere of radius π.
∴ SO(3) is a compact group.

For rotation with Axis angle, chose and axis on a sphere
Every point in the sphere correspond to a rotation element
psi is the radial vector contained in a sphere from origin

SO(3) group manifold





Example: let's say an arbitrary point chosen in the sphere which has rotation=1 in radians which make angles (theta and phi) {in spherical coordinates)

Lie Algebra

- Lie algebra g is a vector space on which is defined a binary operation having the following properties

(1) For all x and y in $\mathfrak{g}$, $[x, y]$ is in $\mathfrak{g}$.
(2) For all x, y and z in $\mathfrak{g}$, and scalars λ and μ, $[\lambda x + \mu y, z] = \lambda[x, z] + \mu[y, z]$.
(3) $[x, y] = -[y, x]$.
(4) $[x, [y, z]] + [y, [z, x]] + [z, [x, y]] = 0$. [Jacobi identity]

[,] is called Lie Bracket

Lie Bracket [x,y]=0 for all x,y implies Lie algebra is abelian

Lecture 39, Prof Ramadevi IITB

1. 🔢 What Are Quaternions?

Quaternions extend complex numbers from 2D to 4D and are used heavily in 3D rotations.

🧮 **Definition:**

A quaternion is written as:

$$q = w + xi + yj + zk$$

Where:

- $w, x, y, z \in \mathbb{R}$
- i, j, k are imaginary units satisfying:

$$i^2 = j^2 = k^2 = ijk = -1$$

And non-commutative multiplication:

$$ij = k, \quad ji = -k, \quad jk = i, \quad kj = -i, \quad ki = j, \quad ik = -j$$

So quaternions live in ℝ⁴.

✅ **Unit Quaternions**

A **unit quaternion** satisfies:

$$|q| = \sqrt{w^2 + x^2 + y^2 + z^2} = 1$$

Unit quaternions represent **3D rotations**, just like SO(3).

🔄 Quaternion as Rotation

A unit quaternion:

$$q = \cos\left(\frac{\theta}{2}\right) + \sin\left(\frac{\theta}{2}\right)(xi + yj + zk)$$

represents rotation of angle θ around axis $\vec{v} = (x, y, z)$, where $\vec{v}$ is a **unit vector**.

To rotate a 3D vector $\vec{v} \in \mathbb{R}^3$:

1. Turn it into a **pure quaternion**: $v_q = 0 + xi + yj + zk$
2. Compute:

$$v' = qv_q q^{-1}$$

2. 🧮 What is U(2)?

U(2) is the **unitary group** of 2×2 complex matrices:

$$U(2) = \{U \in \mathbb{C}^{2\times 2} \mid U^\dagger U = I\}$$

Where:

- $U^\dagger$ = conjugate transpose
- $U^\dagger U = I$ means **length is preserved** under multiplication
- Elements of U(2) are like "complex rotations" in 2D complex vector spaces

🧠 Structure of U(2)

U(2) contains:

- **SU(2)**: special unitary group with **determinant = 1**
- A **global phase**: multiplication by $e^{i\phi}$

Formally:

$$U(2) \cong SU(2) \times U(1)$$

So U(2) is like "SU(2) with an overall complex phase".

3. 🔗 Connection Between Quaternions and SU(2) ⊂ U(2)

Every unit quaternion corresponds to a 2×2 complex matrix in SU(2):

Given quaternion:

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

Its corresponding matrix in SU(2) is:

$$Q = \begin{bmatrix} a + bi & c + di \\ -c + di & a - bi \end{bmatrix} \in SU(2)$$

✅ This matrix is unitary and has determinant = 1.

🎯 So How Are Quaternions Related to U(2)?

- **Unit quaternions** ↔ **SU(2)** (1-to-1)
- **SU(2) ⊂ U(2)**, so quaternions can be embedded in U(2)
- U(2) allows **a global phase** $e^{i\phi}$ times a unit quaternion matrix

So U(2) can be thought of as:

$$U(2) \cong (\text{unit quaternion matrix}) \times (\text{complex phase})$$

🔢 Example

Let's take a unit quaternion:

$$q = \cos\left(\frac{\pi}{4}\right) + \sin\left(\frac{\pi}{4}\right)(0i + 0j + 1k) = \frac{\sqrt{2}}{2} + \frac{\sqrt{2}}{2}k$$

This is a rotation by 90° about the z-axis.

Corresponding SU(2) matrix:

$$Q = \begin{bmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2}i \\ -\frac{\sqrt{2}}{2}i & \frac{\sqrt{2}}{2} \end{bmatrix} \in SU(2) \subset U(2)$$

This matrix rotates 3D vectors just like the quaternion.

🧭 Summary Table

Structure	Description	Dimension	Rotation Use
Quaternion	ℝ⁴ numbers with i, j, k	4	Efficient rotation
Unit Quaternion	Norm = 1	3 (S³)	Represents 3D rotation
U(2)	2×2 unitary matrices	4	Rotations + phase
SU(2)	2×2 unitary with det=1	3	Exactly matches unit quaternions
SO(3)	3×3 real rotation matrices	3	Physical rotations in 3D