Session 3: Transformations

Part A: Transformations
Part B: Motion - Linear and Angular (Preview)

Dr. Arshiya Sood

June 03, 2025

Table of Contents

Session 2 Recap & Reflect: What Did We Learn?

Session 2 Recap & Reflect

Why Study Vectors, Frames, & Transformations?

In Robotics:

- Robots operate in 3D environments — position and orientation matter
- Vectors represent direction and magnitude
- Coordinate frames provide reference contexts (world, base, tool)
- Convert vectors/points between coordinate frames

Transformations Represent:

- **Rotation** – changes orientation
- **Translation** – shifts origin
- **Scaling** – alters object size

Session 2: Recap & Reflect (contd.)

Physical Quantities:
Physical quantities are properties or characteristics of a system that can be measured or quantified.

Scalars:

- Have only magnitude
- Examples: Mass, Energy, Temperature

Vectors:

- Have both magnitude and direction
- Examples: Velocity, Force, Acceleration

1D Vector Algebra:

- Vectors behave like signed numbers
- Operations: Addition, Subtraction, Scalar Multiplication

2D Vector Algebra – Parallelogram Law:

- Two adjacent vectors form a parallelogram
- Resultant vector is the diagonal
- Magnitude:

$$|\vec{R}| = \sqrt{A^2 + B^2 + 2AB\cos\theta}$$

- Direction:

$$\tan\phi = \frac{B\sin\theta}{A + B\cos\theta}$$

Vector Representation in 2D and 3D

In 2D:

$$\vec{A} = A_x\hat{i} + A_y\hat{j} \qquad |\vec{A}| = \sqrt{A_x^2 + A_y^2} \qquad \theta = \tan^{-1}\left(\frac{A_y}{A_x}\right)$$

In 3D:

$$\vec{A} = A_x\hat{i} + A_y\hat{j} + A_z\hat{k} \qquad |\vec{A}| = \sqrt{A_x^2 + A_y^2 + A_z^2}$$

Direction Cosines:

$$\cos\alpha = \frac{A_x}{|\vec{A}|}, \quad \cos\beta = \frac{A_y}{|\vec{A}|}, \quad \cos\gamma = \frac{A_z}{|\vec{A}|}$$

Vector Operations: Dot and Cross Product

Dot Product (Scalar Product):

$$\vec{A} \cdot \vec{B} = AB\cos\phi = A_x B_x + A_y B_y + A_z B_z$$

- Result is a scalar
- $\phi = 0°$: parallel $\rightarrow \vec{A} \cdot \vec{B} = AB$
- $\phi = 90°$: perpendicular $\rightarrow \vec{A} \cdot \vec{B} = 0$
- $\phi = 180°$: antiparallel $\rightarrow \vec{A} \cdot \vec{B} = -AB$

Cross Product (Vector Product):

$$\vec{A} \times \vec{B} = |\vec{A}||\vec{B}| \sin\phi \; \hat{n}$$

$$= \begin{vmatrix} \hat{i} & \hat{j} & \hat{k} \\ A_x & A_y & A_z \\ B_x & B_y & B_z \end{vmatrix}$$

- Result is a vector perpendicular to both
- Follows the right-hand rule

Coordinate Frames in Robotics & System Conventions

What is a Frame?

- A coordinate system = **origin + orientation**
- Describes **pose** (position + orientation)
- All measurements and vectors are expressed **relative to a frame**

Six Parameters to define pose:

- **Translation:** x, y, z
- **Rotation:** roll (x), pitch (y), yaw (z)

Coordinate System Conventions

- **Right-handed system:** Common in robotics
- **Left-handed system:** Less common

Right-hand rule for axes:

- Index finger $\rightarrow +x$
- Middle finger $\rightarrow +y$
- Thumb $\rightarrow +z$

Right-hand rule for rotation:

- Thumb along axis
- Curl of fingers $\rightarrow$ direction of positive rotation

Session 2: Recap & Reflect (contd.)

Types of Coordinate Frames in Robotics

1. **World Frame**
   - A global reference frame.
   - Used to describe the robot's environment.
   - Shared among multiple robots or systems.
2. **Base Frame**
   - Fixed to the robot's base or body.
   - Serves as the origin for all robot joint calculations.
3. **Tool Frame**
   - Attached to the robot's end-effector or tool.
   - Critical for precise control during tasks like welding, painting, or picking.
4. **User-Defined Frame**
   - Custom frame created for specific tasks.
   - Useful for inclined surfaces, fixtures, or part-specific orientation.

Recap Quiz: Vectors and Frames (1/4)

Q1. A robot moves 3m east, then 4m north. What is the magnitude of the resultant displacement vector?

A. 5m

B. 7m

C. 1m

D. 25m

Q2: If the angle between two vectors is 90°, which of the following is always true?

A. Dot product is maximum

B. Dot product is negative

C. Dot product is zero

D. Cross product is zero

Recap Quiz: Vectors and Frames (2/4)

Q3. Which of the following best explains why robots need multiple coordinate frames?

A. To calculate kinetic energy of moving parts

B. To handle complex wiring in embedded systems

C. To express positions and orientations relative to different parts of the robot

D. To switch between analog and digital control modes

Q4. If vector $\vec{A}$ points along $+x$ and $\vec{B}$ along $+y$, the direction of $\vec{A} \times \vec{B}$ is:

A. $-z$

B. $+z$

C. $+y$

D. 0

Recap Quiz: Vectors and Frames (3/4)

Q5. A robotic arm's end-effector is defined in 3D space using which of the following?

A. Length, Mass, and Force

B. Velocity, Acceleration, and Jerk

C. Translation and Rotation

D. Input voltage and Output torque

Q6. A robot's gripper is moving to pick up a small box. The command specifies position relative to the end-effector. Which frame is being used?

A. World Frame

B. Base Frame

C. Tool Frame

D. User Frame

Recap Quiz: Vectors and Frames (4/4)

Q7. Which of the following is true for any vector in 3D space?

A. Must lie along a coordinate axis

B. Can be described using direction cosines

C. Cannot lie in a plane

D. Always has non-zero dot product with all vectors

Q8. If a vector in 3D has $\cos\alpha = 0.6$, $\cos\beta = 0.8$, $\cos\gamma = 0$, then:

A. It lies entirely on the z-axis

B. It has no component in the x-y plane

C. It lies in the x-y plane

D. Its magnitude must be zero

Recap Quiz: Answer Key

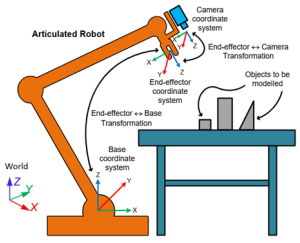
- **Q1:** A (Pythagoras: $\sqrt{3^2 + 4^2} = 5$)
- **Q2:** C (Dot product is zero when $\theta = 90°$)
- **Q3:** C (Different parts need relative positioning)
- **Q4:** B (Right-hand rule: $x \times y = z$)
- **Q5:** C (Pose = Translation + Rotation)
- **Q6:** C (Tool frame is attached to end-effector)
- **Q7:** B (All vectors can be described using direction cosines)
- **Q8:** C (Zero z-component $\rightarrow$ lies in x-y plane)

Part A: Transformations

Motivation: Why Transformations?

Robots operate in a world of motion and orientation. To describe and control this motion, we need transformations.

- **Multiple Coordinate Frames:**
  - Every joint, link, and tool has its own local frame.
  - The world has a global frame.
  - We must convert between frames to control the robot.
- **Transformations = Translation + Rotation**
  - Translation: Shifts position in space.
  - Rotation: Changes orientation.
  - Together, they define an object's **pose**.
- **Examples:**
  - Robot arm computing end-effector pose.
  - Drone flying through 3D space.
- **Why This Matters:**
  - Enables motion planning, sensor fusion, simulation.
  - Helps robots understand where they are and what to do.



What is Pose in Robotics?

Pose = Position + Orientation

Position:

- The location of the robot or its end-effector in 3D space
- Represented by a vector: $\mathbf{p} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$

Orientation:

- The rotation of the robot or its end-effector relative to a reference frame
- Can be represented using:
  - Rotation matrix (3×3)
  - Euler angles (Yaw, Pitch, Roll)
  - Quaternion or Axis-Angle

Pose is a combination of both — and will eventually be represented using a transformation matrix.

Translation in 2D and 3D

Translation refers to moving a point or frame from one position to another without changing its orientation. It is the simplest form of transformation and a foundation for motion planning and coordinate frame conversions.

1. Basic Translation:

- In 2D:

$$P = (x, y), \quad \vec{d} = (dx, dy), \quad P' = (x + dx,\ y + dy)$$

- In 3D:

$$\vec{p'} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} dx \\ dy \\ dz \end{bmatrix} = \begin{bmatrix} x + dx \\ y + dy \\ z + dz \end{bmatrix}$$

- Represents the position of a point or robot's end-effector in 3D space
- Describes "where" the robot or object is located
- Units are typically in meters (m)

Translation in 2D and 3D

2. Translation Between Frames: Let the **child frame** be located at position $(1.5,\ 1.0,\ 0.5)$ w.r.t the **parent frame**.

- A point is defined in the child frame:

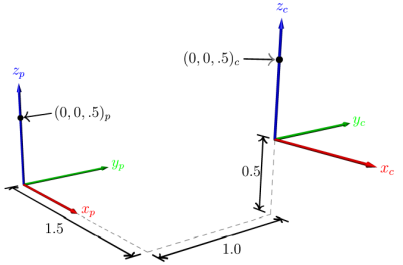
$$\vec{p_c} = (0,\ 0,\ 0.5)_c$$

- Add the translation offset:

$$\vec{p_p} = \vec{p_c} + \vec{d} = (0+1.5,\ 0+1.0,\ 0.5+0.5) = (1.5,\ 1.0,\ 1.0)_p$$

- To convert in the opposite direction (parent to child), subtract the offset:

$$\vec{p_p} = (0,\ 0,\ 0.5)_p \Rightarrow \vec{p_c} = \vec{p_p} - \vec{d} = (-1.5,\ -1.0,\ 0)_c$$



When frames are related by pure translation, transforming a point only requires simple vector addition or subtraction.

What is Rotation?

Rotation describes the change in orientation of a point or object around a fixed axis or origin.

Rotation matrix: is a transformation matrix that operates on a vector and produces a rotated vector such that the coordinate axes always remain fixed. These matrices rotate a vector in the counterclockwise direction by an angle

Rotation in 2D:

$$R(\theta) \in \mathbb{R}^{2\times 2}, \quad R(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

Rotation in 3D:

$$R \in \mathbb{R}^{3\times 3}$$

Properties of Rotation Matrices:

- Orthogonal: $R^\top R = RR^\top = I$
- Inverse is transpose: $R^{-1} = R^\top$
- Determinant is always 1: $\det(R) = 1$

2D Rotation: Derivation

Goal: Derive a matrix that rotates a point

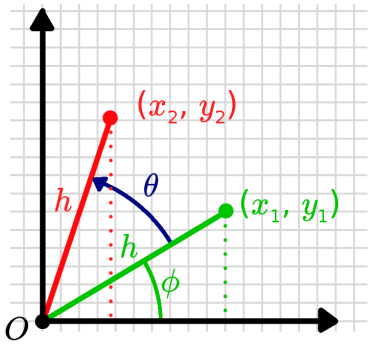
$$\vec{p}_1 = \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} \quad \text{by angle } \theta \text{ to get} \quad \vec{p}_2 = \begin{bmatrix} x_2 \\ y_2 \end{bmatrix}$$

Step 1: Represent in polar coordinates

$$\begin{bmatrix} x_1 \\ y_1 \end{bmatrix} = \begin{bmatrix} h\cos\varphi \\ h\sin\varphi \end{bmatrix}$$

Step 2: After rotation by θ:

$$\begin{bmatrix} x_2 \\ y_2 \end{bmatrix} = \begin{bmatrix} h\cos(\varphi+\theta) \\ h\sin(\varphi+\theta) \end{bmatrix}$$



Rotating a point by angle θ about the origin

Source: https://articulatedrobotics.xyz/tutorials/coordinate-transforms/rotation-matrices-2d

2D Rotation Matrix: Derivation Continued

Step 3: Use trig identities and simplify

$$\cos(\varphi + \theta) = \cos\varphi\cos\theta - \sin\varphi\sin\theta$$

$$\sin(\varphi + \theta) = \sin\varphi\cos\theta + \cos\varphi\sin\theta$$

$$\begin{bmatrix} x_2 \\ y_2 \end{bmatrix} = \begin{bmatrix} h\cos\varphi\cos\theta - h\sin\varphi\sin\theta \\ h\sin\varphi\cos\theta + h\cos\varphi\sin\theta \end{bmatrix} = \begin{bmatrix} x_1\cos\theta - y_1\sin\theta \\ x_1\sin\theta + y_1\cos\theta \end{bmatrix}$$

Step 4: Final matrix form

$$\begin{bmatrix} x_2 \\ y_2 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \end{bmatrix}$$

3D Rotation: Derivation – Matrix along Z-axis

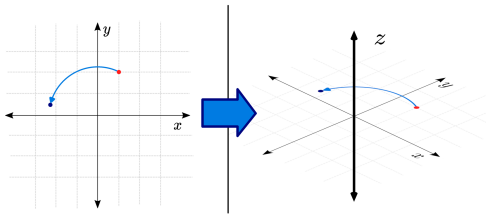
Step 1: Represent the point

$$\vec{p} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

We want to rotate it around the Z-axis by angle θ.

Step 2: Rotate in the XY-plane

$$\begin{bmatrix} x_{\text{new}} \\ y_{\text{new}} \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$



Rotation in the XY-plane (Z-axis)

Source: `https://articulatedrobotics.xyz/tutorials/coordinate-transforms/rotations-3d`

3D Rotation: Derivation – Matrix along Z-axis

Step 3: Z coordinate remains unchanged

$$z_{\text{new}} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = z$$

Step 4: Final matrix form (3D Rotation about Z-axis)

$$\begin{bmatrix} x_{\text{new}} \\ y_{\text{new}} \\ z_{\text{new}} \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

This matrix rotates a point around the Z-axis by angle θ.

Rotation Matrices for Each Axis

3D Rotation can be performed about X, Y, or Z axes.

Rotation about X-axis (Roll):

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

Rotation about Y-axis (Pitch):

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

Rotation about Z-axis (Yaw):

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

Roll, Pitch, and Yaw: Head Movement Analogy

Roll (X-axis) – "Tilt" Motion (Tilt head toward shoulder)

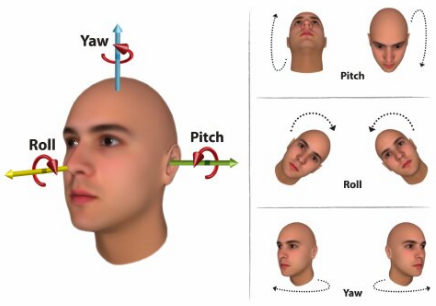
- Axis goes through the nose to the back (X-axis)
- Affects Y and Z, X remains fixed

Pitch (Y-axis) – "Yes" Motion (Nod head up and down)

- Axis goes through the ears (Y-axis)
- Affects X and Z, Y remains fixed

Yaw (Z-axis) – "No" Motion (Shaking head left and right)

- Axis goes up through the head (Z-axis)
- Affects X and Y, Z remains fixed



Source: https://www.researchgate.net/publication/279291928_Enhanced_real-time_head_pose_estimation_system_for_mobile_device/figures?lo=1

Why Homogeneous Transformation Matrices?

Motivation: Unifying Rotation and Translation

- In robotics, we frequently need to express both:
  - **Rotation:** using a 3×3 matrix
  - **Translation:** using a 3×1 vector
- Applying rotation and translation separately can be **inefficient and error-prone**.
- To streamline computations, we use a **single** 4×4 matrix called the **homogeneous transformation matrix**.
- This matrix allows us to perform both operations using **matrix multiplication** in homogeneous coordinates.

Key Benefit: Enables chaining multiple transformations (e.g., Base $\rightarrow$ Joint $\rightarrow$ Tool) seamlessly.

Homogeneous Coordinates and Transformation Matrix

1. Homogeneous Coordinates

A point in 3D: $\mathbf{p} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$, in homogeneous form: $\mathbf{p}_h = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$

2. Homogeneous Transformation Matrix

$$T = \begin{bmatrix} R & \mathbf{d} \\ \mathbf{0}^\top & 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & d_x \\ r_{21} & r_{22} & r_{23} & d_y \\ r_{31} & r_{32} & r_{33} & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

where:

- $R \in \mathbb{R}^{3\times3}$: Rotation matrix
- $\mathbf{d} \in \mathbb{R}^{3\times1}$: Translation vector

Transforming Points Using Homogeneous Matrix

3. Transformation of a 3D point:

$$\mathbf{p}_h = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Then, the transformed point is:

$$\mathbf{p}_h = T \cdot \mathbf{p}$$

Result:

$$\mathbf{p}_h = \begin{bmatrix} R & \mathbf{d} \\ \mathbf{0}^T & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} R \cdot \mathbf{p}_{xyz} + \mathbf{d} \\ 1 \end{bmatrix}$$

4. Interpretation:

- The original point is rotated and then translated.
- The last row ensures correct affine transformation using matrix multiplication.

Chaining Transformations Using Homogeneous Matrices

Suppose we have three reference frames: A, B, and C.

Let:

- T_{AB}: transformation from frame A to frame B
- T_{BC}: transformation from frame B to frame C

Given a vector $\vec{x}_A$ in frame A, we can express it in frame C by chaining the transformations:

$$\vec{x}_C = T_{BC} \cdot T_{AB} \cdot \vec{x}_A$$

This means:

- First, transform the vector from frame A to frame B using T_{AB}
- Then, transform it from frame B to frame C using T_{BC}

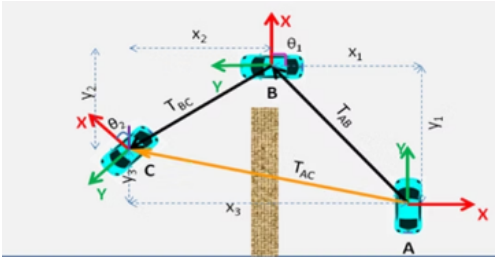
This scales to longer chains as well:

$$\vec{x}_E = T_{DE} \cdot T_{CD} \cdot T_{BC} \cdot T_{AB} \cdot \vec{x}_A$$

Chaining Transformations – Car Example

Setting: We have three cars in a parking area — A, B, and C.

Car A can see Car B, but not Car C (blocked by a wall).
Car B can see Car C.



We attach local frames to each car.
Let's say the observer is in Car A.
Car A sees Car B at position $[x_1, y_1]$ and orientation θ_1.
Transformation from frame A to B:

$$T_{AB}$$

$$T_{AB} = \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & x_1 \\ \sin\theta_1 & \cos\theta_1 & y_1 \\ 0 & 0 & 1 \end{bmatrix}$$

Chaining Transformations – Car Example

From Car B's perspective, Car C is at $[x_2, y_2]$ and rotated by angle θ_2.

Transformation from frame B to C:

$$T_{BC}$$

$$T_{BC} = \begin{bmatrix} \cos\theta_2 & -\sin\theta_2 & x_2 \\ \sin\theta_2 & \cos\theta_2 & y_2 \\ 0 & 0 & 1 \end{bmatrix}$$

Now, the task for Car A is to find the pose of Car C.

Since A knows T_{AB} and B knows T_{BC}, A can compute:

$$T_{AC} = T_{AB} \cdot T_{BC}$$

$$T_{AC} = T_{AB} * T_{BC}$$

$$T_{AC} = \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & x_1 \\ \sin\theta_1 & \cos\theta_1 & y_1 \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} \cos\theta_2 & -\sin\theta_2 & x_2 \\ \sin\theta_2 & \cos\theta_2 & y_2 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} \cos\theta_3 & -\sin\theta_3 & x_3 \\ \sin\theta_3 & \cos\theta_3 & y_3 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \cos\theta_1 * \cos\theta_2 - \sin\theta_1 * \sin\theta_2 & -\cos\theta_1 * \sin\theta_2 - \sin\theta_1 * \cos\theta_2 & \cos\theta_1 * x_2 - \sin\theta_1 * y_2 + x_1 \\ \sin\theta 1 * \cos\theta_2 + \cos\theta_1 * \sin\theta_2 & -\sin\theta_1 * \sin\theta_2 + \cos\theta_1 * \cos\theta_2 & \sin\theta_1 * x_2 + \cos\theta_1 * y_2 + y_1 \\ 0 & 0 & 1 \end{bmatrix}$$

Chaining Transformations – Car Example

With this chained transformation, Car A can now determine the position and orientation of Car C **without seeing it directly**.

$\cos\theta_1: c_1, \sin\theta_1 : s_1$; $\cos\theta_2: c_2, \sin\theta_2: s_2$; $\cos\theta_3: c_3, \sin\theta_3: s_3$; $\cos\theta: c, \sin\theta: s$

$\cos\theta_1 * \sin\theta_1 = c_1 s_1$; $\cos\theta_1 * \sin\theta_2 = c_1 s_2$; $\cos\theta_2 * \sin\theta_1 = c_2 s_1$; so on

$$\begin{bmatrix} c_3 & -s_3 & x_3 \\ s_3 & c_3 & y_3 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} c_1c_2 - s_1s_2 & -c_1s_2 - s_1c_2 & c_1x_2 - s_1y_2 + x_1 \\ s_1c_2 + c_1s_2 & -s_1s_2 + c_1c_2 & s_1x_2 + c_1y_2 + y_1 \\ 0 & 0 & 1 \end{bmatrix}$$

From above equation I can derive x3 and y3 by equating last column elements,

$x_3 = c_1x_2 - s_1y_2 + x_1$ & $y_3 = s_1x_2 + c_1y_2 + y_1$

Chaining Transformations – Car Example

From the final transformation T_{AC}, Car A can extract:

- The global coordinates $[x_3, y_3]$ of Car C.
- The orientation θ_3.

$$\frac{-s_3}{c_3} = \frac{-c_1 s_2 - s_1 c_2}{-s_1 s_2 + c_1 c_2}$$

$$\frac{\sin\theta_3}{\cos\theta_3} = \frac{c_1 s_2 + s_1 c_2}{-s_1 s_2 + c_1 c_2}$$

$$\tan\theta_3 = \frac{c_1 s_2 + s_1 c_2}{-s_1 s_2 + c_1 c_2}$$

$$\theta_3 = \operatorname{atan2}\left(\frac{c_1 s_2 + s_1 c_2}{-s_1 s_2 + c_1 c_2}\right)$$

Source: https://www.rosroboticslearning.com/rigid-body-transformations

Rotation Concepts – Demo Videos

To deepen our understanding of rotations and orientation, we'll now watch a few short video clips:

- **GeoGebra Rotation Animation**
  `youtu.be/PIDNpWG2s1Y`
  Simple visual explanation of 2D/3D rotations using GeoGebra animations.

- **Robots: Axis and Orientation of Movement – Pitch, Roll, Yaw**
  `youtu.be/EjvdpkX9fD8`
  Demonstrates robot arm orientation through roll, pitch, and yaw motion examples.

- **Understanding the Rotation Matrix in 3D**
  `youtu.be/8GrdqAizcfU`
  Explains how the 3D rotation matrix works with axis-angle visuals.

B: Motion : Linear and Angular (Preview)

Motion in Robotics: Linear and Angular

Motion in robotics can be broadly classified into:

- **Linear Motion:**
  - Movement along a straight or curved path.
  - Examples: Robot arm extending, mobile robot moving forward.
- **Angular Motion:**
  - Rotation about a fixed axis or point.
  - Examples: Joint rotation in a robotic arm, drone propeller spinning.

We will explore the physics and vector mathematics behind both forms of motion.

Linear Motion: Velocity and Acceleration

Linear Velocity ($\vec{v}$):

- Describes the rate of change of position with time.
- Vector quantity: has both magnitude and direction.
- Mathematically: $\vec{v} = \frac{d\vec{x}}{dt}$

Linear Acceleration ($\vec{a}$):

- Describes the rate of change of velocity with time.
- Indicates speeding up, slowing down, or changing direction.
- Mathematically: $\vec{a} = \frac{d\vec{v}}{dt}$

Both quantities are fundamental in understanding how robots move linearly in space.

Angular Motion: Velocity and Acceleration

Angular Velocity ($\vec{\omega}$):

- Describes the rate of change of angular position.
- It is a vector along the axis of rotation.
- Mathematically: $\vec{\omega} = \frac{d\vec{\theta}}{dt}$
- Units: radians/second

Angular Acceleration ($\vec{\alpha}$):

- Describes the rate of change of angular velocity.
- Indicates if a rotating body is speeding up or slowing down.
- Mathematically: $\vec{\alpha} = \frac{d\vec{\omega}}{dt}$
- Units: radians/second^2

Relationship Between Linear and Angular Motion

Tangential Velocity ($\vec{v}$):

- An object at distance r from the axis of rotation has linear velocity:

$$\vec{v} = \vec{\omega} \times \vec{r}$$

- $\vec{v}$ is perpendicular to both $\vec{\omega}$ and $\vec{r}$.

Tangential Acceleration ($\vec{a}_t$):

- Caused by angular acceleration:

$$\vec{a}_t = \vec{\alpha} \times \vec{r}$$

Centripetal Acceleration ($\vec{a}_c$):

- Always directed towards the center:

$$\vec{a}_c = -\omega^2 \vec{r}$$