Lecture 3

- Lecture 3: https://www.youtube.com/watch?v=XwNsJFcSQx4
- Written by Milad Abdi with some AI assistance.

State Estimation

- **Goal:** Determine **where we are in space** and **how we got there**.
- Needed to **plan sequences of movements** to reach a desired position.
- Requires understanding:
    - **Pose representations** (position + orientation)
    - **Coordinate transformations** for rigid bodies (e.g., F1TENTH race car).

Coordinate Frames and Transformations

Frames and Transformations on Autonomous Vehicles

- **Sensors provide measurements in their own frame of reference.**
    - Navigation requires **fusing multiple sensor perspectives**.
    - Each sensor is first **processed individually**, then combined.
- **Multiple sensor outputs** can be combined to produce a **Bird's Eye View**.
- **Mapping** is usually done by tying together submaps.

What is a Coordinate Frame?

1. A **set of orthogonal axes** attached to a body that describes positions of points relative to that body.
2. Axes intersect at a point called the **origin**.
   Why coordinate frames are needed:

- Compact representation of points and orientations.
- Coordinates are meaningless without specifying the frame.
    - Allows understanding and relating quantities observed by **different sensors or robots** at different time steps.
- **Pose = position + rotation**.
- **Rigid body transformations** convert points from one frame to another.

Right-handed Coordinate Frames

- **Direction of axes is important.**

- Mnemonic:
  - Positive **x-axis** → index finger
  - Positive **y-axis** → middle finger
  - Positive **z-axis** → thumb
- **Right-handed systems** are standard in robotics.
- Left-handed systems are sometimes used in graphics.

Transformations and Frames

- **Map frame:** Defines where the robot is relative to a fixed world origin.
- **Sensor frame:** Defines where obstacles are relative to the sensor.
- **Measurements in the sensor frame** do **not directly tell you where obstacles are in the map**
  - Must account for offsets $\Delta x, \Delta y, \Delta z$ from the center of the car.
- **Transformations** allow converting measurements **from one frame to another**.
  Two types of transformations:

1. **Static transformations:** Fixed offsets (e.g., sensor mounted on the car).
2. **Dynamic transformations:** Change over time (e.g., moving robot, rotating sensors)

Rigid Body Transformations and Reference Frames on F1TENTH

Rigid Body Definition:

- A rigid body maintains **constant distances between any two points** regardless of external forces.
- Deformations are assumed negligible relative to motion.
  Rigid Body Motion:

1. **Translation (rectilinear or curvilinear):**
   - Motion of the body can be described by the motion of any single point.
   - All points share the **same velocity and acceleration**.
2. **Rotation about a fixed axis:**
   - All particles move in **circular paths around the axis**.
   - Motion is fully determined by the **angular velocity** of rotation
     Why we need transform between frames:

- Sensor data is usually provided in the **sensor's local frame**.
- To **localize on a map**, we need transformations:
  - From **laser frame** → **map frame** (obstacle positions in global coordinates)
  - Between **components on the car** for accurate actuation

- Between **submaps** for building a consistent global map

Reference Frames on F1TENTH

- **map:** Global frame representing the environment; origin set arbitrarily when creating the map and kept consistent.
- **base_link:** Actuation frame, centered at the car's rear axle; moves with the car relative to map.
- **laser:** Measurement frame for LIDAR scans; moves with the car; transformation to base_link is static.
- **odom:** Odometry frame; tracks relative motion and provides pose estimates of the car relative to odom.
  Key Points:
- Measurements in the **laser frame** give information about obstacles and relative robot position.
- Measurements in the **map frame** align observations with the global environment.
- Planning, TTC, and safety computations are more accurate in the **global frame**.
- Example: TTC calculations can differ depending on travel direction if only laser-frame measurements are used; adding a **chassis-centered frame** standardizes calculations.

Rigid Body Transformations

- Lecture does a poor job at explaining so don't worry if you didn't understand the math section. Instead, watch these 3 videos and make sure you understand them:
- Part 1: **Introduction**: https://www.youtube.com/watch?v=FvgGSgvB2I0
- Part 2: **Forward Kinematics**: https://www.youtube.com/watch?v=_8T7RjXL07M
- Part 3: **Inverse Kinematics**: https://www.youtube.com/watch?v=8D0sO8mymQ8
- We go from 2D to 3D because translation is not a linear transformation. So we embed our 2D coordinates into a **fixed** 3D system using homogeneous coordinates, where $z = 1$ for all points. We never change z.

Note on Rigid Body Transform Notation

- $\mathbf{p}, \mathbf{p}'$: original and transformed points in standard coordinates
- $\mathbf{P}, \mathbf{P}'$: original and transformed points in homogeneous coordinates; extra variable is **set to 1 and never changed**
- R : rotation matrix (2×2 for 2D, 3×3 for 3D)
- $\mathbf{t}$: translation vector
- r_{ij} : entry of the rotation matrix at row i, column j of the rotation matrix. It shows how the rotation moves the original axes (x, y, z) to the new axes (x', y', z'). In 3D, all the r_{ij} numbers together control rotations around x, y, and z (yaw, pitch, roll).

- **Summary** : standard vector form $\mathbf{p}' = R\mathbf{p} + \mathbf{t}$; homogeneous form $\mathbf{P}' = T\mathbf{P}$, with the extra variable **fixed at 1**

1. 2D Rigid Body Transform

Standard vector form:

$$\mathbf{p}' = R\mathbf{p} + \mathbf{t}$$

$$\mathbf{p} = \begin{bmatrix} x \\ y \end{bmatrix}, \quad \mathbf{p}' = \begin{bmatrix} x' \\ y' \end{bmatrix}, \quad R = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}, \quad \mathbf{t} = \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

Homogeneous coordinates form:

$$\mathbf{P}' = T\mathbf{P}, \quad \mathbf{P} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}, \quad \mathbf{P}' = \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}, \quad T = \begin{bmatrix} R & \mathbf{t} \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & t_x \\ \sin\theta & \cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

2. 3D Rigid Body Transform

Standard vector form:

$$\mathbf{p}' = R\mathbf{p} + \mathbf{t}$$

$$\mathbf{p} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}, \quad \mathbf{p}' = \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix}, \quad R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}, \quad \mathbf{t} = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}$$

Homogeneous coordinates form:

$$\mathbf{P}' = T\mathbf{P}, \quad \mathbf{P} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}, \quad \mathbf{P}' = \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix}, \quad T = \begin{bmatrix} R & \mathbf{t} \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

3. Additional Notes on Rotation Matrices and Inverses

- **Rotation matrices** R are **orthogonal** and have **determinant 1**, which means they preserve lengths and angles when rotating points.
- **Why R^T works:** R^T reverses the rotation because the transpose of an orthogonal matrix is also its inverse, so applying R^T “undoes” the original rotation.
- **Why the inverse translation is $-R^T\mathbf{t}$:** After reversing the rotation with R^T, we must also move back by the opposite of the original translation, which becomes $-R^T\mathbf{t}$ in the transformed frame.
- **Inverse of a rigid body transform** T:

$$T^{-1} = \begin{bmatrix} R^T & -R^T\mathbf{t} \\ 0 & 1 \end{bmatrix}, \quad \text{where } R^T \text{ reverses the rotation, and } -R^T\mathbf{t} \text{ reverses the translation.}$$

Poses & Transformations

- Points, directions, and displacements are represented by **n-dimensional vectors**.
- Rotations and scalings are represented by **n×n matrices**.
- **Rigid transformations** consist of a rotation followed by a translation; they represent both rigid body motion and changes of coordinate frame.
- **Homogeneous coordinates:** represent rigid transforms in **(n+1)-dimensional space**, last coordinate 0 for vectors, 1 for points.
  Practical Tip:
- Clear assumptions, consistent notation, and/or using coordinate management tools reduces the risk of errors when working with multiple frames and transformations.