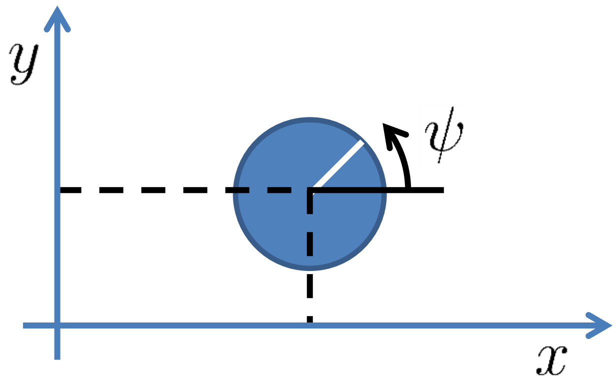
Autonomous Navigation for Flying Robots

Lecture 2.3:
2D Robot Example

Jürgen Sturm
Technische Universität München

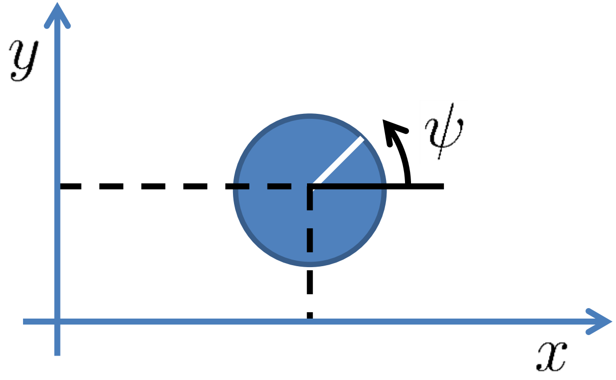
2D Robot

- Robot is located somewhere in space



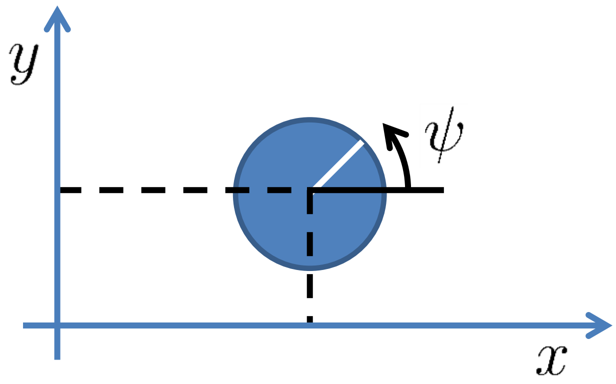
2D Robot

- Robot is located somewhere in space
- Robot pose:
  - Position x, y
  - Orientation ψ (yaw angle/heading)



Robot Pose

- Robot is located somewhere in space
- Robot pose:
  - Position x, y
  - Orientation ψ (yaw angle/heading)
- Robot pose represented as transformation matrix:

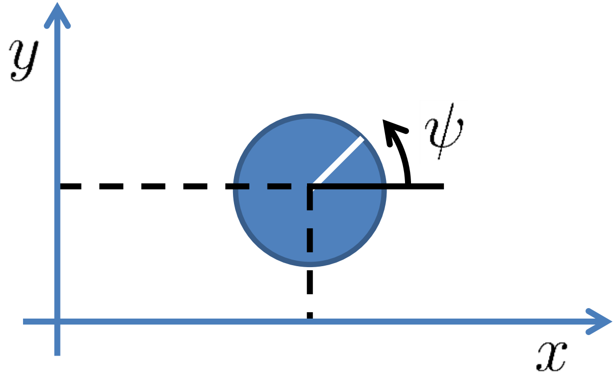


$$\mathbf{X} = \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{pmatrix} = \begin{pmatrix} \cos\psi & -\sin\psi & x \\ \sin\psi & \cos\psi & y \\ 0 & 0 & 1 \end{pmatrix} \in \mathrm{SE}(2) \subset \mathbb{R}^{3x3}$$

Robot Pose

- Robot is located at

$$x = 0.7$$
$$y = 0.5$$
$$\psi = 45°$$

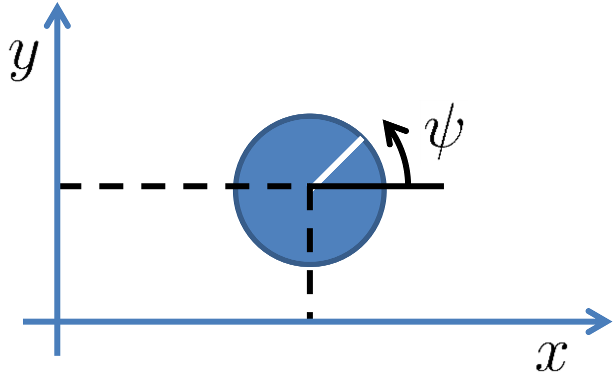


- Robot pose

$$\mathbf{X} = \begin{pmatrix} \cos 45 & -\sin 45 & 0.7 \\ \sin 45 & \cos 45 & 0.5 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 0.71 & -0.71 & 0.7 \\ 0.71 & 0.71 & 0.5 \\ 0 & 0 & 1 \end{pmatrix}$$

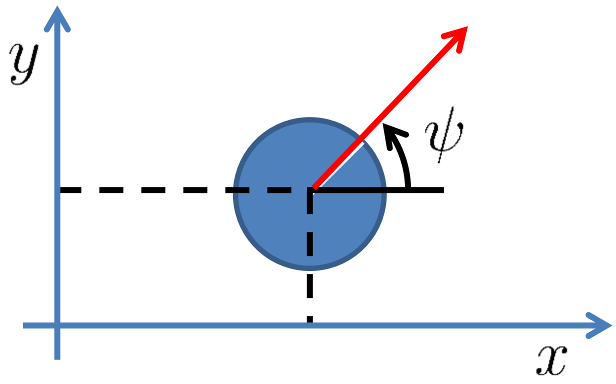
Coordinate Transformations

- Robot is located somewhere in space
- Robot pose:
  - Position x, y
  - Orientation ψ (yaw angle/heading)
- What is the pose after moving 1m forward?
- How do we need to move to reach a certain position?



Coordinate Transformations

- Robot moves forward by 1m
- What is its position afterwards?

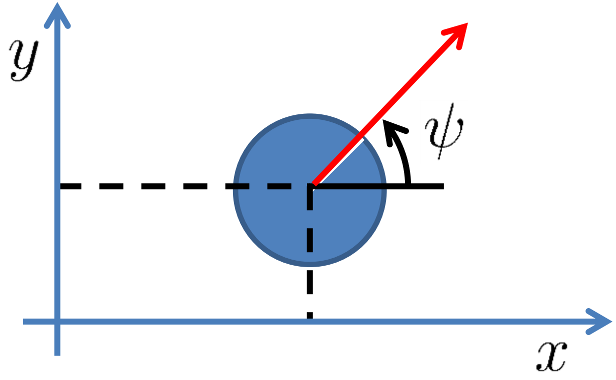


- Point located 1m in front of the robot in local coordinates:

$$\tilde{\mathbf{p}}_{\text{local}} = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}$$

Coordinate Transformations

- Robot moves forward by 1m
- What is its position afterwards?



- Point located 1m in front of the robot in global coordinates:

$$\tilde{\mathbf{p}}_{\text{global}} = \mathbf{X}\tilde{\mathbf{p}}_{\text{local}} = \begin{pmatrix} 0.71 & -0.71 & 0.7 \\ 0.71 & 0.71 & 0.5 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1.41 \\ 1.21 \\ 1 \end{pmatrix}$$

Coordinate Transformations

- We transformed local to global coordinates
- Sometimes we need to do the inverse
- How can we transform global coordinates into local coordinates?

Coordinate Transformations

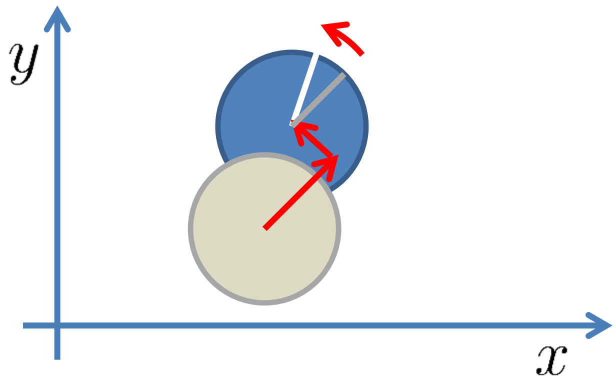
- We transformed local to global coordinates
- Sometimes we need to do the inverse
- How can we transform global coordinates into local coordinates?

$$\tilde{\mathbf{p}}_{\text{global}} = \mathbf{X}\tilde{\mathbf{p}}_{\text{local}} = \begin{pmatrix} R & \mathbf{t} \\ \mathbf{0} & 1 \end{pmatrix} \tilde{\mathbf{p}}_{\text{local}}$$

$$\tilde{\mathbf{p}}_{\text{local}} = \mathbf{X}^{-1}\tilde{\mathbf{p}}_{\text{global}} = \begin{pmatrix} R^\top & -R^\top\mathbf{t} \\ \mathbf{0} & 1 \end{pmatrix} \tilde{\mathbf{p}}_{\text{global}}$$

Coordinate Transformations

- Now consider a different motion
- Robot moves 0.2m forward,
  0.1m sideward and turns by 10deg

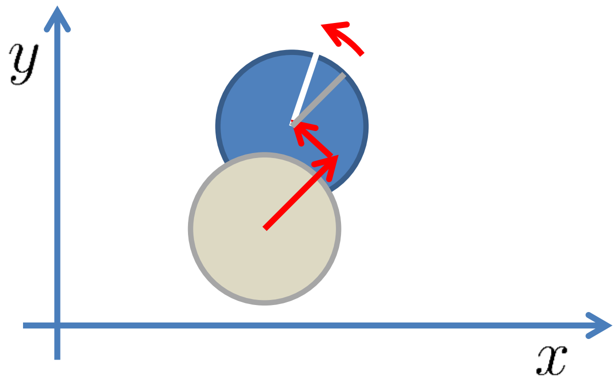


- Euclidean transformation:

$$\mathbf{U} = \begin{pmatrix} \cos 10 & -\sin 10 & 0.2 \\ \sin 10 & \cos 10 & 0.1 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 0.98 & -0.17 & 0.2 \\ 0.17 & 0.98 & 0.1 \\ 0 & 0 & 1 \end{pmatrix}$$

Coordinate System Transformations

- Now consider a different motion
- Robot moves 0.2m forward,
  0.1m sideward and turns by 10deg



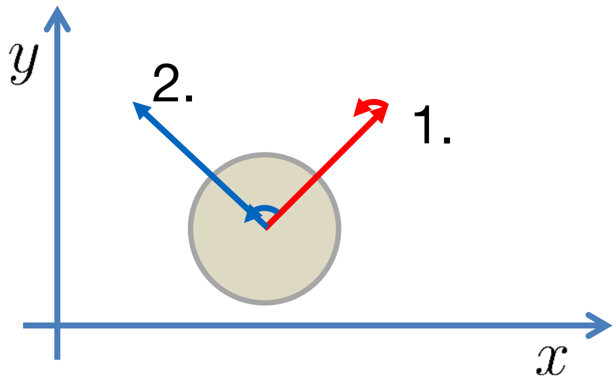
- After this motion, the robot pose becomes

$$\mathbf{X}' = \mathbf{X}\mathbf{U} = \begin{pmatrix} 0.71 & -0.71 & 0.7 \\ 0.71 & 0.71 & 0.5 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0.98 & -0.17 & 0.2 \\ 0.17 & 0.98 & 0.1 \\ 0 & 0 & 1 \end{pmatrix} = \cdots$$

Coordinate System Transformations

Note: The order matters!

$$\mathbf{AB} \neq \mathbf{BA}$$



Compare:

1. Move 1m forward, then turn 90deg left
2. Turn 90deg left, then move 1m forward

Robot Odometry

How can we estimate the robot motion?

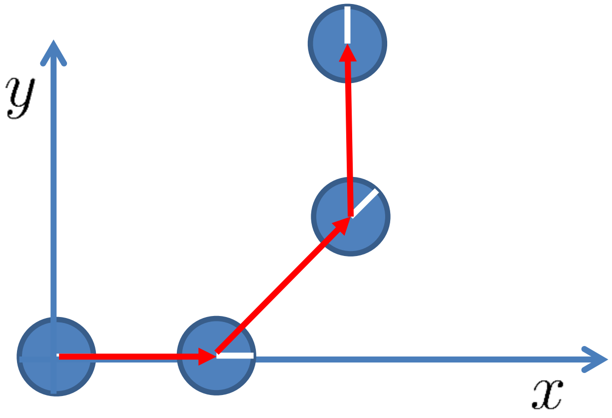
- **Control-based** models predict the estimated motion from the issued control commands
- **Odometry-based** models are used when systems are equipped with distance sensors (e.g., wheel encoders)
- **Velocity-based** models have to be applied when no wheel encoders are given

Dead Reckoning

- Integration of odometry is also called dead reckoning
- Mathematical procedure to determine the present location of a vehicle
- Achieved by calculating the current pose of the vehicle based on the estimated/measured velocities and the elapsed time

Motion Models

- Estimating the robot pose $\mathbf{X}_t$ based on the issued controls (or IMU readings) $\mathbf{u}_t$ and the previous location $\mathbf{X}_{t-1}$
- Motion model $\mathbf{X}_t = f(\mathbf{X}_{t-1}, \mathbf{u}_t)$



Exercise

- **Given:**
  - IMU readings from real flight of Ardrone quadrotor
  - Horizontal speed in the local frame
  - Yaw angular speed
- **Wanted:**
  - Position and orientation in global frame
  - Integrate these values to get robot pose

Lessons Learned

- 2D pose
- Conversion between local and global coordinates
- Concatenation of (robot) motions
- Robot odometry