Visual Navigation for Flying Robots

Exercise 1 - Odometry

Dr. Jürgen Sturm

Nikolas Engelhard

Organization

- 26 submissions
- 9 teams (1x4, 6x3, 2x2)
- Re-balance large team

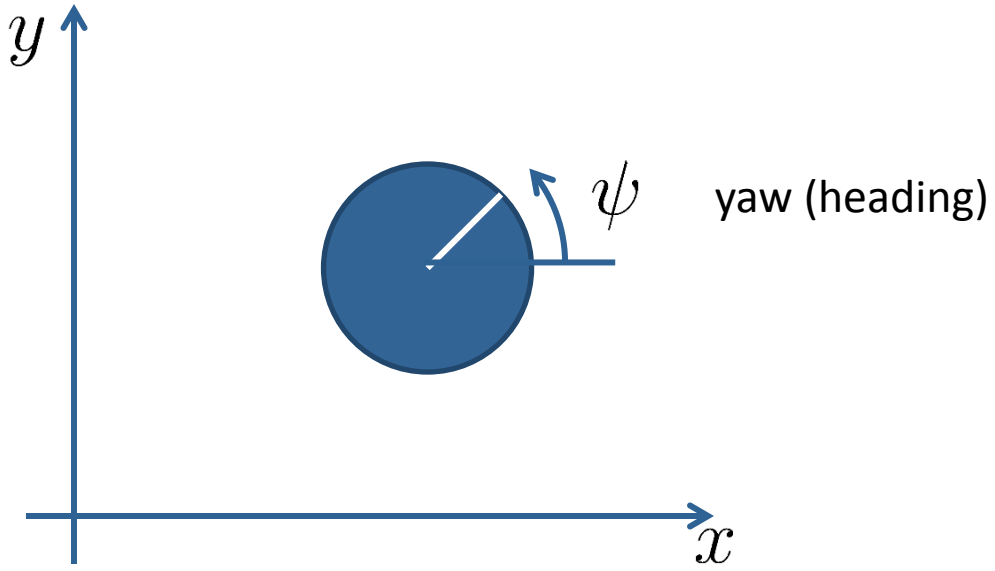
- Good news: All students who have handed in a submission will be admitted to the course

Coordinate Transforms

- Local coordinates
- Global coordinates

Coordinate Transforms

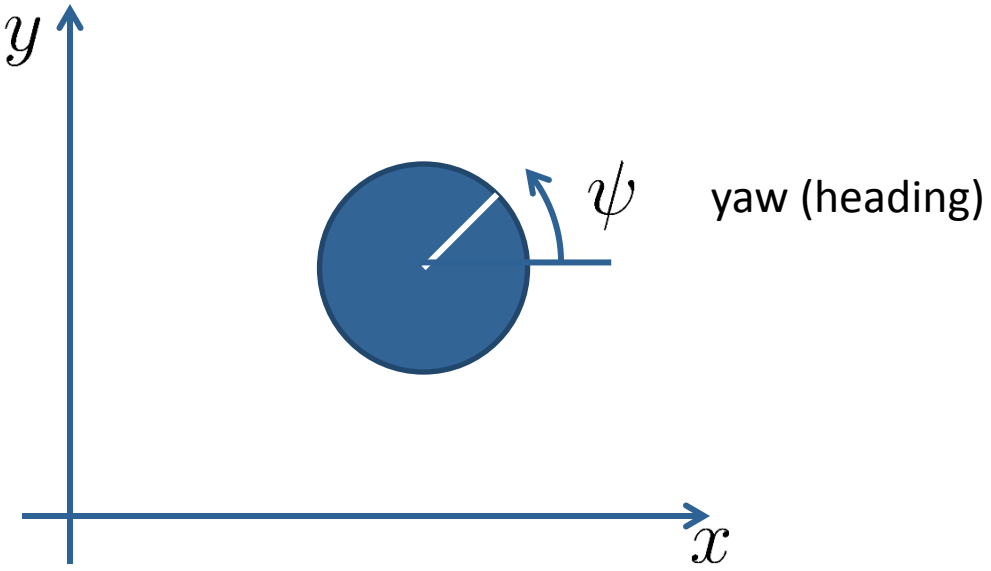
- Robot is located somewhere in space



Coordinate Transforms

- Robot is located somewhere in space

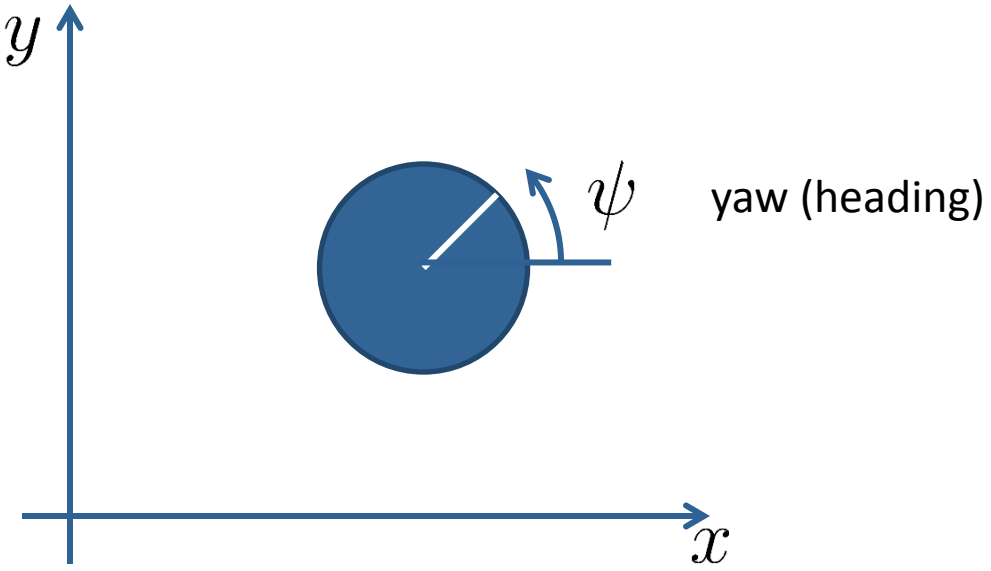
$$X = \begin{pmatrix} R & \mathbf{t} \\ \mathbf{0} & 1 \end{pmatrix} = \begin{pmatrix} cos\psi & -sin\psi & x \\ sin\psi & cos\psi & y \\ 0 & 0 & 1 \end{pmatrix} \in \mathrm{SE}(2) \subset \mathbb{R}^{3x3}$$



Coordinate Transforms

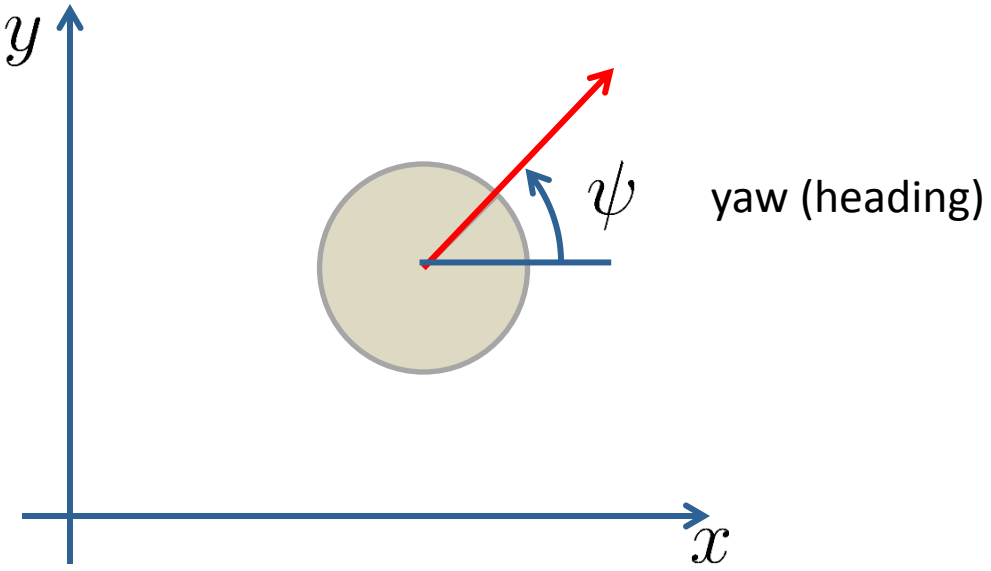
- Robot is located at x=0.7, y=0.5, yaw=45deg

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



Vector Transformation

- Robot is located at x=0.7, y=0.5, yaw=45deg
- Robot moves forward with 1m/s (in its local frame)
- What is its speed in the global frame?



Vector Transformation

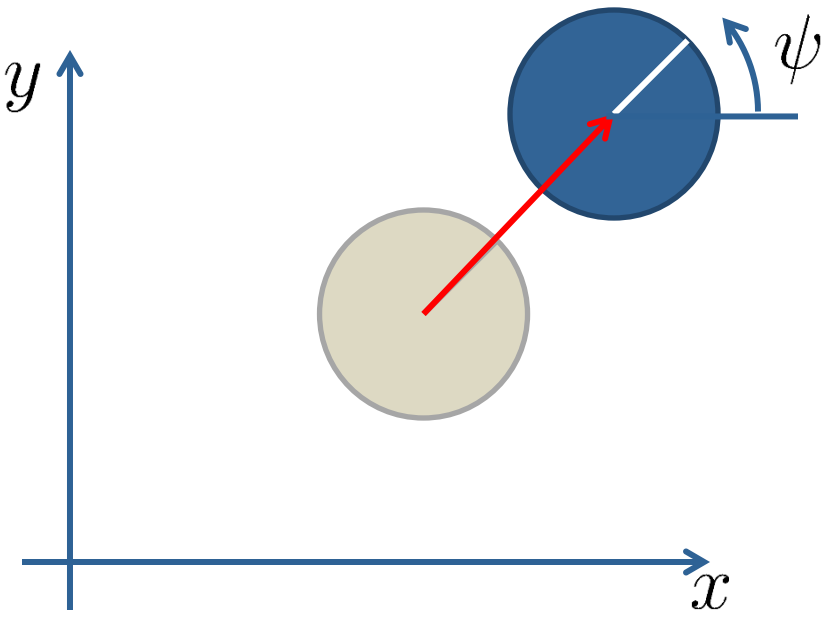
- Robot is located at x=0.7, y=0.5, yaw=45deg
- Robot moves forward with 1m/s

Inhomogeneous coordinates

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

Homogeneous coordinates

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



Vector Transformation

- Robot is located at x=0.7, y=0.5, yaw=45deg
- Robot moves forward with 1m/s

$$\tilde{\mathbf{v}}_{\text{global}} = X \tilde{\mathbf{v}}_{\text{local}}$$

Vector Transformation

- Robot is located at x=0.7, y=0.5, yaw=45deg
- Robot moves forward with 1m/s

$$\tilde{\mathbf{v}}_{\text{global}} = X \tilde{\mathbf{v}}_{\text{local}}$$

$$= \begin{pmatrix} 0.71 & -0.71 & 0.7 \\ 0.71 & 0.71 & 0.5 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}$$

Vector Transformation

- Robot is located at x=0.7, y=0.5, yaw=45deg
- Robot moves forward with 1m/s

$$\tilde{\mathbf{v}}_{\text{global}} = X \tilde{\mathbf{v}}_{\text{local}}$$

$$= \begin{pmatrix} 0.71 & -0.71 & 0.7 \\ 0.71 & 0.71 & 0.5 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}$$

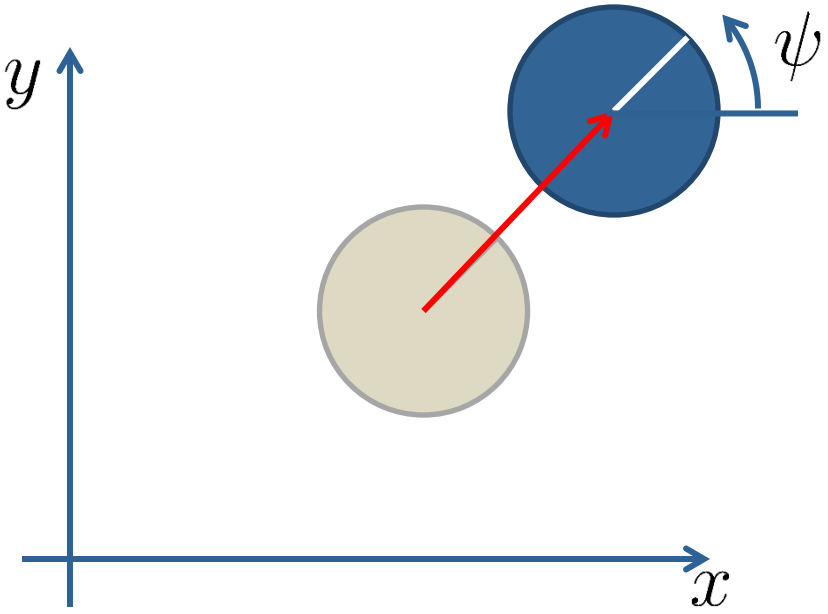
$$= \begin{pmatrix} 1.41 \\ 1.21 \\ 1 \end{pmatrix}$$

Vector Transformation

- Robot is located at x=0.7, y=0.5, yaw=45deg
- Robot moves forward with 1m/s

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

$$\tilde{\mathbf{v}}_{\text{global}} = \begin{pmatrix} 1.41 \\ 1.21 \\ 1 \end{pmatrix}$$



Vector Transformation

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

$$\tilde{\mathbf{v}}_{\text{global}} = X \tilde{\mathbf{v}}_{\text{local}}$$

Vector Transformation

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

$$\tilde{\mathbf{v}}_{\text{global}} = X \tilde{\mathbf{v}}_{\text{local}}$$

$$\tilde{\mathbf{v}}_{\text{local}} = X^{-1} \tilde{\mathbf{v}}_{\text{global}}$$

Inverse Transformations

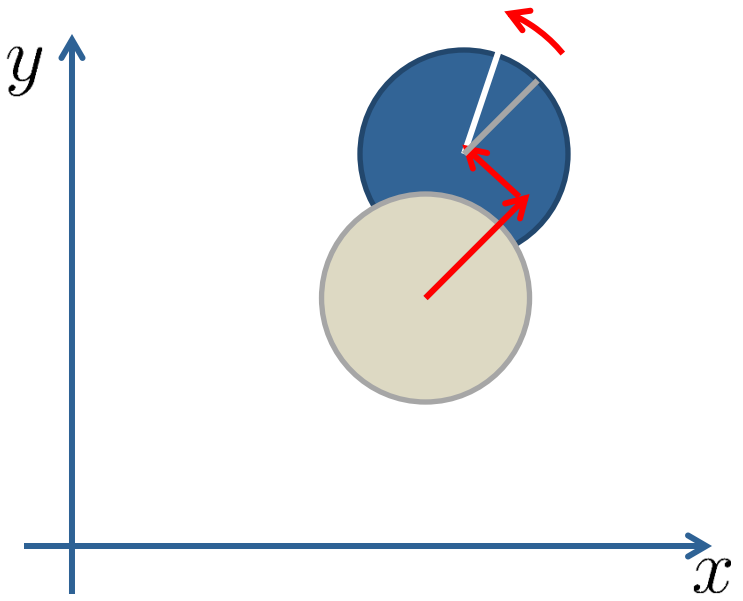
- We transformed local to global coordinates
- Sometimes we need to do the inverse
- How can we transform global coordinates into local coordinates?

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

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

Coordinate System Transformations

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



Coordinate System Transformations

- Robot moves 0.2m forward, 0.1m sideways, and turns by 10deg

$$U_1 = \begin{pmatrix} \cos 10 & -\sin 10 & 0.2 \\ \sin 10 & \cos 10 & 0.1 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 0.98 & -0.17 & 0.2 \\ 0.17 & 0.98 & 0.1 \\ 0 & 0 & 1 \end{pmatrix}$$

Coordinate System Transformations

- After this motion, the robot pose (in the global frame) becomes

$$X_2 = XU$$

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

Coordinate System Transformations

- Note: The order matters
- Move 1m forward, then turn 90deg left
- Turn 90deg left, then move 1m forward

$$AB \neq BA$$

3D Transformations

- Translation $\mathbf{t}$ has 3 degrees of freedom
- Rotation R has 3 degrees of freedom

$$X = \begin{pmatrix} R & \mathbf{t} \\ \mathbf{0} & 1 \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Orientations in 3D

- Rotation matrix (9 parameters)
  - Concatenation: matrix multiply
  - Inverse: matrix inverse

Orientations in 3D

- Euler angles (3 parameters)
  - Concatenation: convert to rotation matrix, multiply, convert back
  - Inverse: convert to rotation matrix, invert, convert back

$$R_Z(\psi_1)R_Y(\theta_1)R_X(\phi_1) \cdot R_Z(\psi_2)R_Y(\theta_2)R_X(\phi_2)$$
$$\neq R_Z(\psi_1+\psi_2)R_Y(\theta_1+\theta_2)R_X(\phi_1+\phi_2)$$

Orientations in 3D

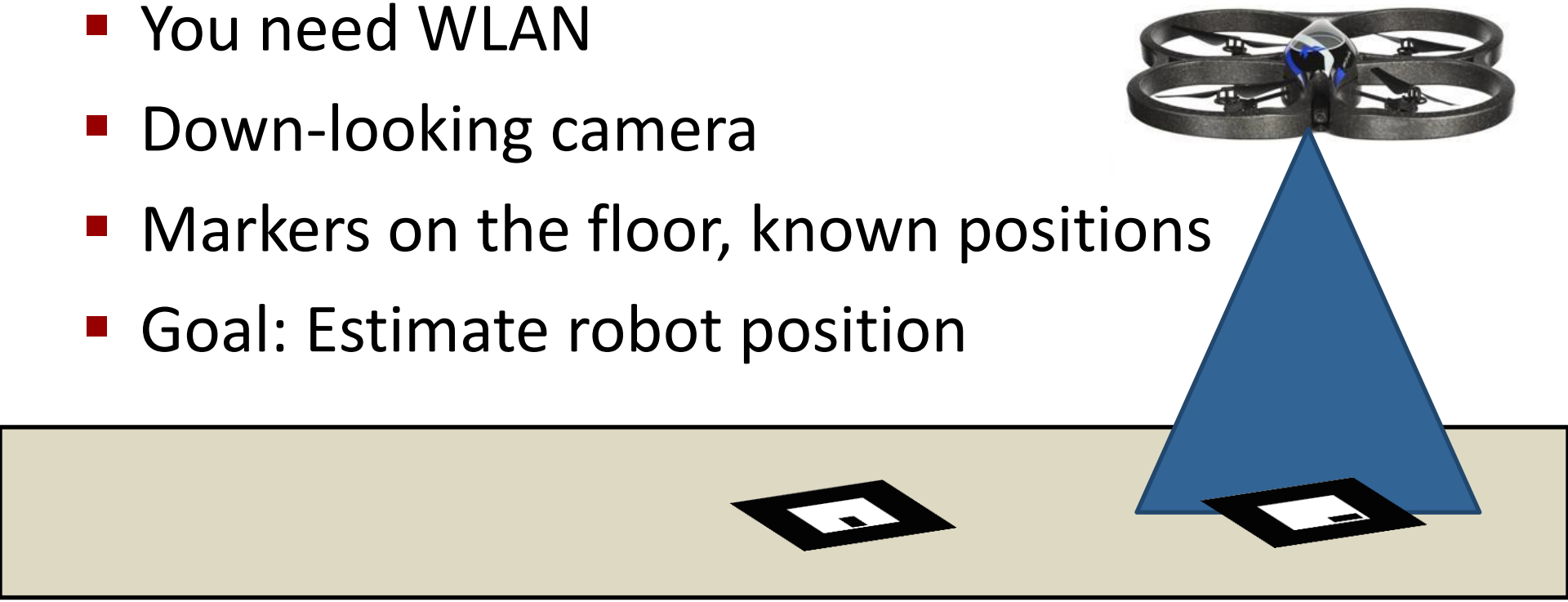
- Quaternions (4 parameters)
  - Concatenation: quaternion multiplication
  - Inverse: multiplication with conjugate

Orientations in 3D

- Axis/angle (3 parameters)
  - Concatenation: convert to rotation matrix or quaternions, multiply, convert back
  - Inverse: convert to rotation matrix or quaternions, invert, convert back

Sheet 2

- Fly the robot with a joystick
- Record own bag files
- You need WLAN
- Down-looking camera
- Markers on the floor, known positions
- Goal: Estimate robot position



Sheet 2

1. Fly the robot!
   - Using a joystick
   - Record your own bag file
2. Kalman Filter – Prediction step
   - Will provide C++ framework
   - Run on recorded bag files (or online)
   - Change parameters and see what happens
3. Kalman Filter – Correction step
   - Specify observation function and compute Jacobian
   - Implement it