ECE276A: Sensing & Estimation in Robotics
Lecture 8: Motion and Observation Models

Instructor:

Nikolay Atanasov: natanasov@ucsd.edu

Teaching Assistants:

Qiaojun Feng: qif007@eng.ucsd.edu
Tianyu Wang: tiw161@eng.ucsd.edu
Ibrahim Akbar: iakbar@eng.ucsd.edu
You-Yi Jau: yjau@eng.ucsd.edu
Harshini Rajachander: hrajacha@eng.ucsd.edu

UC San Diego
JACOBS SCHOOL OF ENGINEERING
Electrical and Computer Engineering

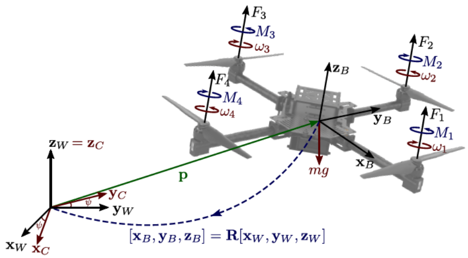
Motion Model

- A motion model describes the density function $p_f(\cdot \mid x, u)$ of a new robot state after motion for a given state x with control input u
- A motion model can be obtained using:
  - Supervised learning from a dataset $D = \{(x_i, u_i, x_i')\}$ of transitions
  - Model-based reinforcement learning, where it is inferred indirectly as the robot is learning to perform a task
  - Kinematics or dynamics modeling
    - Differential drive model
    - Ackermann drive (bicycle) model
    - Quadrotor model
    - Legged locomotion model
  - Odometry, i.e., using sensor data (e.g., wheel encoders, IMU, camera, laser) to estimate ego motion in retrospect, after the robot has moved

Quadrotor Motion Model

- **State** $x = (p, \dot{p}, R, \omega_B)$ with position $p \in \mathbb{R}^3$, velocity $\dot{p} \in \mathbb{R}^3$, orientation $R \in SO(3)$, and rotational velocity $\omega_B \in \mathbb{R}^3$
- **Control** $u = (h, M_B)$ with thrust $h \in \mathbb{R}$ and moment $M_B \in \mathbb{R}^3$
- **Continuous-time model** with mass $m \in \mathbb{R}_{>0}$, gravitational acceleration g, moment of inertial $J \in \mathbb{R}^{3\times 3}$ and z-axis $e_3 \in \mathbb{R}^3$:

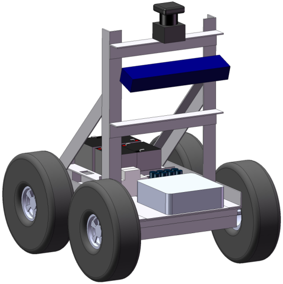
$$\dot{x} = f(x, u) = \begin{cases} m\ddot{p} = -mge_3 + hRe_3 \\ \dot{R} = R\hat{\omega}_B \\ J\dot{\omega}_B = -\omega_B \times J\omega_B + M_B \end{cases}$$



Differential-drive Motion Model

- **State** $s = (p, \theta) \in SE(2)$, where $p = (x, y) \in \mathbb{R}^2$ is the position and $\theta \in (-\pi, \pi]$ is the orientation (yaw angle)
- **Control** $u = (v, \omega)$, where $v \in \mathbb{R}$ is the linear velocity and $\omega \in \mathbb{R}$ is the rotational velocity (yaw rate)
- **Continuous-time model**:

$$\dot{s} = f(s, u) = \begin{cases} \dot{p} = v \begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix} \\ \dot{\theta} = \omega \end{cases}$$

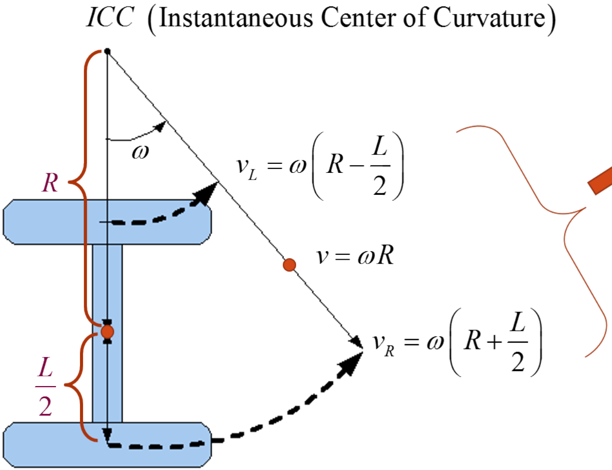


- **Discrete-time model** with time discretization τ:

$$s_{t+1} = f(s_t, u_t) := s_t + \tau \begin{pmatrix} v_t \mathbf{sinc}\left(\frac{\omega_t \tau}{2}\right) \cos\left(\theta_t + \frac{\omega_t \tau}{2}\right) \\ v_t \mathbf{sinc}\left(\frac{\omega_t \tau}{2}\right) \sin\left(\theta_t + \frac{\omega_t \tau}{2}\right) \\ \omega_t \end{pmatrix}$$

Continuous-time Differential-drive Model

- Let $s_t := (x_t, y_t, \theta_t)^T \in SE(2)$ be the pose of a differential drive robot
- What is the state $s_{t+\tau}$ after τ seconds if we apply linear velocity v_t and angular velocity ω_t?



$$\omega = \frac{v_R - v_L}{L}$$

$$R = \frac{L}{2}\left(\frac{v_L + v_R}{v_R - v_L}\right) = \frac{v}{\omega}$$

$$v = \frac{v_R + v_L}{2}$$

$$\dot{x}(t) = v\cos\theta(t)$$
$$\dot{y}(t) = v\sin\theta(t)$$
$$\dot{\theta}(t) = \omega$$

Discrete-time Differential-drive Model

- To convert the continuous-time differential-drive model to discrete time, we can solve the ordinary differential equations:

$$
\begin{aligned}
\dot{x}(t) &= v\cos\theta(t) \\
\dot{y}(t) &= v\sin\theta(t) \\
\dot{\theta}(t) &= \omega
\end{aligned}
\quad \Rightarrow \quad
\begin{aligned}
\theta(t) &= \theta(t_0) + \int_{t_0}^{t} \omega ds = \theta(t_0) + \omega(t - t_0) \\
x(t) &= x(t_0) + v\int_{t_0}^{t} \cos\theta(s) ds \\
&= x(t_0) + \frac{v}{\omega}\left(\sin\left(\omega(t-t_0) + \theta(t_0)\right) - \sin\theta(t_0)\right) \\
&= x(t_0) + v(t-t_0)\frac{\sin(\omega(t-t_0)/2)}{\omega(t-t_0)/2}\cos(\theta(t_0) + \omega(t-t_0)/2) \\
y(t) &= y(t_0) + v\int_{t_0}^{t} \sin\theta(s) ds \\
&= y(t_0) - \frac{v}{\omega}\left(\cos\theta(t_0) - \cos\left(\omega(t-t_0) + \theta(t_0)\right)\right) \\
&= y(t_0) + v(t-t_0)\frac{\sin(\omega(t-t_0)/2)}{\omega(t-t_0)/2}\sin(\theta(t_0) + \omega(t-t_0)/2)
\end{aligned}
$$

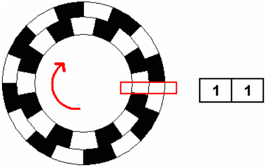
- Let $\tau := t - t_0$ be the time discretization

Encoders

- A magnetic encoder consists of a rotating gear, a permanent magnet and a sensing element
- The sensor has two output channels with offset phase to determine the direction of rotation
- A microcontroller counts the number of transitions adding or subtracting 1 (depending on the direction of rotation) to the counter
- The distance traveled by the wheel, corresponding to one tick on the encoder is:

$$\text{meters per tick} = \frac{\pi \times (\text{wheel diameter})}{\text{ticks per revolution}}$$

- The distance traveled during time τ for a given encoder count c, wheel diameter d, and 360 ticks per revolution is $\tau v \approx \frac{\pi d c}{360}$ and can be used to predict the position change in a differential-drive model



Odometry-based Motion Model

- A "drifting" estimate of the robot pose ${}_W\hat{T}_t := \begin{bmatrix} \hat{R}_t & p_t \\ \mathbf{0} & 1 \end{bmatrix}$ in the world frame $\{W\}$ is provided by the motion sensors over time (e.g., by integrating the encoder measurements through the differential drive motion model)

- The pose trajectory is noisy due to integration errors but any individual transformation from time $t+1$ to time t is accurate:

$$u_t := {}_t\hat{T}_{t+1} = \left({}_W\hat{T}_t\right)^{-1} {}_W\hat{T}_{t+1} \in SE(3)$$

- The relative transformation u_t can be used to define an odometry-based motion model to predict a new robot state $x_{t+1} \in SE(3)$ (specifying the transformation from the body frame at time $t+1$ to the world frame) from the current robot state $x_t \in SE(3)$:

$$x_{t+1} = x_t \oplus u_t$$

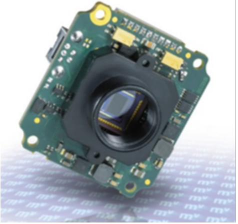
  where $\oplus$ emphasizes that the above is a composition of $SE(3)$ elements

Observation Model

- An observation model describes the measurement likelihood $p_h(z \mid x, m)$ for a given sensor pose x and environment representation m
- **Position model**: direct position measurements, e.g., GPS, RGBD camera, laser scanner
- **Bearing model**: angular measurements to points in 3-D, e.g., compass, RGB camera
- **Range model**: distance measurements to points in 3-D, e.g., radio received signal strength (RSS) or time-of-flight
- **Inertial measurement unit**: magnetometer, gyroscope, accelerometer



Cameras



Global shutter

Stereo (+ IMU)

RGBD

Event-based

Lasers

Single-beam Garmin Lidar

2-D Hokuyo Lidar

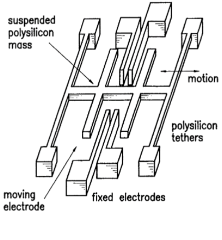
3-D Velodyne Lidar

Observation Models

- **Position sensor**: state $x = (p, R)$, position $p \in \mathbb{R}^3$, orientation $R \in SO(3)$, observed point $m \in \mathbb{R}^3$, measurement $z \in \mathbb{R}^3$:
$$z = h(x, m) = R^T(m - p)$$
- **Range sensor**: state $x = (p, R)$, position $p \in \mathbb{R}^3$, orientation $R \in SO(3)$, observed point $m \in \mathbb{R}^n$, measurement $z \in \mathbb{R}$:
$$z = h(x, m) = \|R^T(m - p)\|_2 = \|m - p\|_2$$
- **Bearing sensor**: state $x = (p, \theta)$, position $p \in \mathbb{R}^2$, orientation $\theta \in (-\pi, \pi]$, observed point $m \in \mathbb{R}^2$, bearing $z \in (-\pi, \pi]$:
$$z = h(x, m) = \arctan\left(\frac{m_2 - p_2}{m_1 - p_1}\right) - \theta$$
- **Camera sensor**: state $x = (p, R)$, position $p \in \mathbb{R}^3$, orientation $R \in SO(3)$, intrinsic camera matrix $K \in \mathbb{R}^{2\times 3}$, observed point $m \in \mathbb{R}^3$, pixel $z \in \mathbb{N}^2$:
$$z = h(x, m) = K\pi(R^T(m - p)) \qquad \text{projection:} \quad \pi(m) := \frac{1}{m_z} m$$

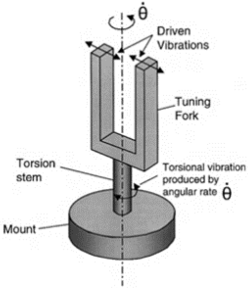
MEMS Strapdown IMU

- **MEMS**: micro-electro-mechanical system
- **IMU**: inertial measurement unit:
  - triaxial accelerometer
  - triaxial gyroscope (measures angular velocity)
  - **Strapdown**: the IMU and the object/vehicle inertial frames are joined together/identical



Surface Micromachined Accelerometer

- **Accelerometer**:
  - A mass m on a spring with constant k. The spring displacement is prop. to the system acceleration: $F = ma = kd \quad \Rightarrow \quad d = \frac{ma}{k}$
  - VLSI Fabrication: the displacement of a metal plate with mass m is measured with respect to another plate using capacitance
  - Used for car airbags (if the acceleration goes above $2g$, the car is hitting something!)



- **Gyroscope**: uses Coriolis force to detect rotational velocity from the changing mechanical resonsance of a tuning fork

IMU Observation Model

- **Robot State** $(p, \dot{p}, \ddot{p}, R, \omega_B, \dot{\omega}_B, b_g, b_a)$ with position $p \in \mathbb{R}^3$, velocity $\dot{p} \in \mathbb{R}^3$, acceleration $\ddot{p} \in \mathbb{R}^3$, orientation $R \in SO(3)$, rotational velocity $\omega_B \in \mathbb{R}^3$ (body frame), and rotational acceleration $\dot{\omega}_B \in \mathbb{R}^3$ (body frame), gyroscope bias $b_g \in \mathbb{R}^3$, accelerometer bias $b_a \in \mathbb{R}^3$
- **Extrinsics**: IMU position ${}_Bp_I \in \mathbb{R}^3$ and orientation ${}_BR_I \in SO(3)$ in the body frame (assumed known or obtained via calibration)
- **Measurement** (z_ω, z_a) with rotational velocity measurement $z_\omega \in \mathbb{R}^3$ and linear acceleration measurement $z_a \in \mathbb{R}^3$

IMU Observation Model

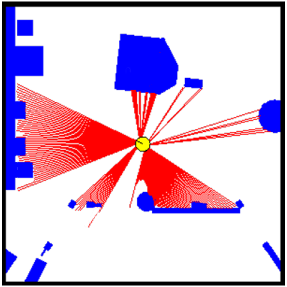
- **Continuous-time model**: with gravitational acceleration g, gyro measurement noise $n_g \in \mathbb{R}^3$, accelerometer measurement noise $n_a \in \mathbb{R}^3$ (assumed zero-mean white Gaussian):

$$
\begin{aligned}
z_\omega &= {}_B R_I^T \omega_B + b_g + n_g \\
z_a &= {}_W R_I^T \left({}_W \ddot{p}_I - g \right) + b_a + n_a \\
&= \left(R \, {}_B R_I \right)^T \left(\frac{d}{dt^2} \left(p + R \, {}_B p_I \right) - g \right) + b_a + n_a \\
&= {}_B R_I^T \left(R^T (\ddot{p} - g) + \left[\hat{\dot{\omega}}_B \right] {}_B p_I + \left[\hat{\omega}_B^2 \right] {}_B p_I \right) + b_a + n_a
\end{aligned}
$$

- **Discrete-time model**: A. Mourikis and S. Roumeliotis, "A Multi-State Constraint Kalman Filter for Vision-aided Inertial Navigation"

LIDAR Model

- **Lidar**: LIght Detection And Ranging
- Illuminates the scene with pulsed laser light and measures the return times and wavelengths of the reflected pulses
- Mirrors are used to steer the laser beam in the *xy* plane (and *zy* plane for 3D lidars)
- Example: Hokuyo URG-04LX; detectable range: 0.02 to 4m; 240° field of view with 0.36° angular resolution (666 beams); 100 ms/scan



Laser Range-Azimuth-Elevation Model

- Consider a Lidar with position $p \in \mathbb{R}$ and orientation $R \in SO(3)$ observing a points $m \in \mathbb{R}^3$ in the world frame
- The point m has coordinates $\bar{m} := R^T(m - p)$ in the lidar frame
- The lidar provides a spherical coordinate measurement of m:

$$R^T(m - p) = \begin{bmatrix} r\cos\alpha\cos\epsilon \\ r\sin\alpha\cos\epsilon \\ r\sin\epsilon \end{bmatrix}$$

  where r is the range, α is the azimuth, and ϵ is the elevation

- **Inverse observation model**: expresses the lidar state p, R and environment state m, in terms of the measurement $z = \begin{bmatrix} r & \alpha & \epsilon \end{bmatrix}^T$
- Inverting gives the **laser range-azimuth-elevation model**:

$$z = \begin{bmatrix} r \\ \alpha \\ \epsilon \end{bmatrix} = \begin{bmatrix} \|\bar{m}\|_2 \\ \arctan\left(\bar{m}_y / \bar{m}_x\right) \\ \arcsin\left(\bar{m}_z / \|\bar{m}\|_2\right) \end{bmatrix} \qquad \bar{m} = R^T(m - p)$$

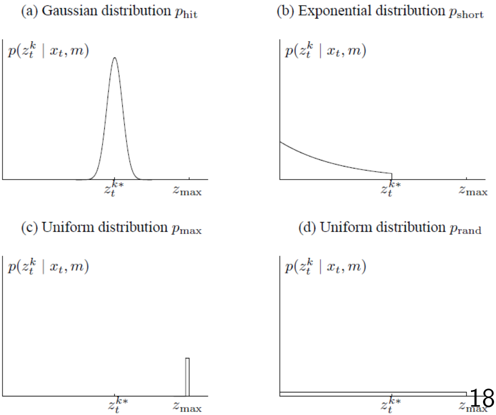
Laser Beam Model

- Let z_t^k be the k-th laser beam obtained from sensor pose x_t in map m
- Let z_t^{k*} be the expected range measurement from x_t in m and let z_{max} be the max laser range
- The laser sensor model assumes that the **beams are independent**:

$$p_h(z_t \mid x_t, m) = \prod_k p(z_t^k \mid x_t, m)$$

Four types of measurement noise:

1. Small measurement noise: p_{hit}, Gaussian
2. Unexpected object: p_{short}, Exponential
3. Unexplained noise: p_{rand}, Uniform
4. No objects hit: p_{max}, Uniform



Laser Beam Model

- Independent beam assumption: $p_h(z_t \mid x_t, m) = \prod_k p(z_t^k \mid x_t, m)$
- Four types of noise:

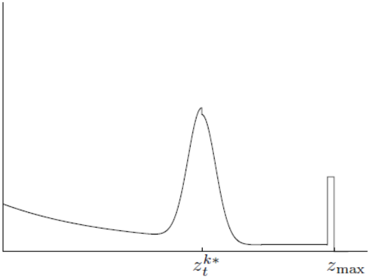
$$p(z_t^k \mid x_t, m) = \alpha_1 p_{hit}(z_t^k \mid x_t, m) + \alpha_2 p_{short}(z_t^k \mid x_t, m) + \alpha_3 p_{rand}(z_t^k \mid x_t, m) + \alpha_4 p_{max}(z_t^k \mid x_t, m)$$

$$p_{hit}(z_t^k \mid x, m) = \begin{cases} \frac{\phi(z_t^k; z_t^{k*}, \sigma^2)}{\int_0^{z_{max}} \phi(s; z_t^{k*}, \sigma^2) ds} & \text{if } 0 \leq z_t^k \leq z_{max} \\ 0 & \text{else} \end{cases}$$

$$p_{short}(z_t^k \mid x, m) = \begin{cases} \frac{\lambda_s e^{-\lambda_s z_t^{k*}}}{1 - e^{-\lambda_s z_t^{k*}}} & \text{if } 0 \leq z_t^k \leq z_t^{k*} \\ 0 & \text{else} \end{cases}$$

$$p_{rand}(z_t^k \mid x, m) = \begin{cases} \frac{1}{z_{max}} & \text{if } 0 \leq z_t^k < z_{max} \\ 0 & \text{else} \end{cases}$$

$$p_{max}(z_t^k \mid x, m) = \delta(z_t^k; z_{max}) := \begin{cases} 1 & \text{if } z_t^k = z_{max} \\ 0 & \text{else} \end{cases}$$



Laser Correlation Model

- A model for a laser scan z obtained from sensor pose x in an occupancy map m obtained by modeling the correlation between z and m
- **Occupancy grid map**: a grid with free ($m_i = 0$) and occupied ($m_i = 1$) cells
- **Laser Correlation Model**:
  1. Transform the scan z to the world frame using x and find all points y in the grid that correspond to the scan
  2. Let the observation model be proportional to the similarty $\mathbf{corr}(y, m)$ between the transformed scan y and the grid m
- The correlation is large if y and m agree:

$$\mathbf{corr}(y, m) := \sum_i \mathbb{1}\{m_i = y_i\}$$

- The weights can be converted to probabilities via the **softmax** function:

$$p_h(z \mid x, m) = \frac{e^{\mathbf{corr}(y,m)}}{\sum_v e^{\mathbf{corr}(v,m)}} \propto e^{\mathbf{corr}(y,m)}$$