Autonomous Mobile Robot Design

Topic: Recap Vehicle Modeling, State Estimation

Dr. Kostas Alexis (CSE)

Autonomous Robot Challenges

How do I move?

Autonomous Mobile Robot Design

Topic: Vehicle Propulsion

Dr. Kostas Alexis (CSE)

The Micro Aerial Vehicle propeller

- Simplified model forces and moments:
  - **Thrust Force:** the resultant of the vertical forces acting on all the blade elements.

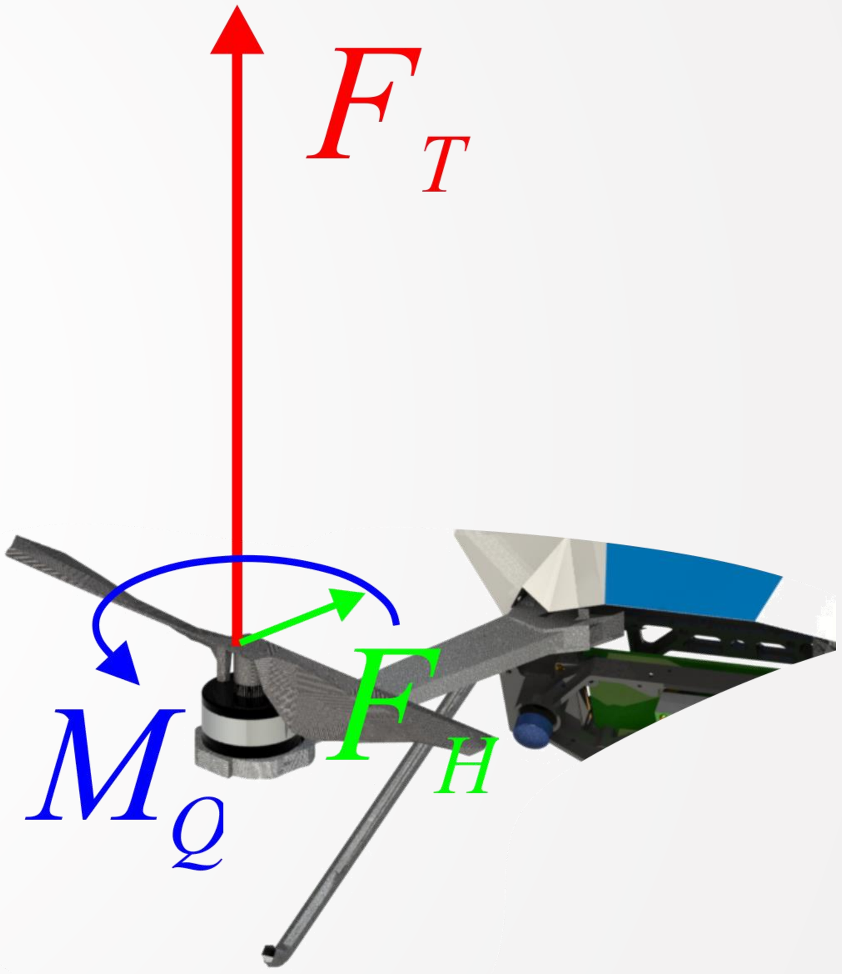
$$F_T = T = C_T \rho A (\Omega R)^2$$

  - **Hub Force:** the resultant of all the horizontal forces acting on all the blade elements.

$$F_H = H = C_H \rho A (\Omega R)^2$$

  - **Drag Moment:** This moment about the rotor shaft is caused by the aerodynamic forces acting on the blade elements. The horizontal forces acting on the rotor are multiplied by the moment arm and integrated over the rotor. Drag moment determines the power required to spin the rotor.

$$M_Q = Q = C_Q \rho A (\Omega R)^2 R$$



The wheel of a small ground robot

- **Circular Motion – Rotational Formulas**
  - Angular Velocity

    $$\omega = \theta / t \qquad v = \omega r$$

  - Angular Velocity and Acceleration

    $$\omega = \omega_0 + at$$

  - Angular Displacement

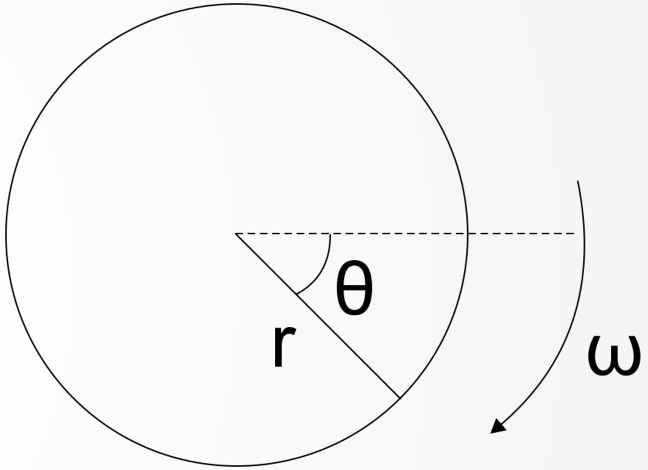
    $$\theta = \omega_0 t + \frac{1}{2} a t^2$$

  - Angular Acceleration

    $$a = \frac{d^2\theta}{dt} = \frac{d\omega}{dt}$$

  - Angular Momentum or Torque

    $$T = aJ_w$$



- ω = angular velocity
- θ = angular position
- r = radius of the wheel
- a = angular acceleration
- J_w = moment inertia
- T = angular momentum

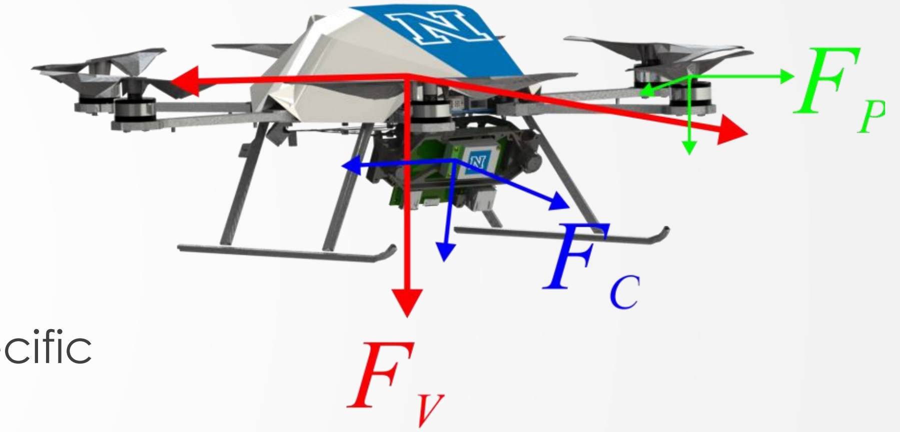
Autonomous Mobile Robot Design

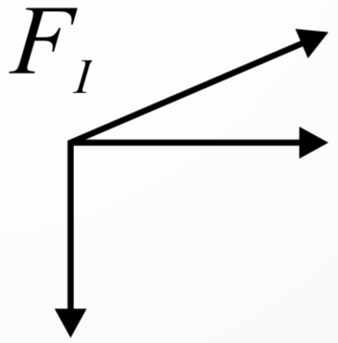
Topic: Coordinate Frame Transformations

Dr. Kostas Alexis (CSE)

Coordinate Frames

- In Guidance, Navigation and Control of aerial robots, reference coordinate frames are fundamental.
- Describe the relative position and orientation of:
  - Aerial Robot ***relative*** to the Inertial Frame
  - On-board Camera ***relative*** to the Aerial Robot body
  - Aerial Robot ***relative*** to Wind Direction
- Some expressions are easier to formulate in specific frames:
  - Newton's law
  - Aerial Robot Attitude
  - Aerodynamic forces/moments
  - Inertial Sensor data
  - GPS coordinates
  - Camera frames



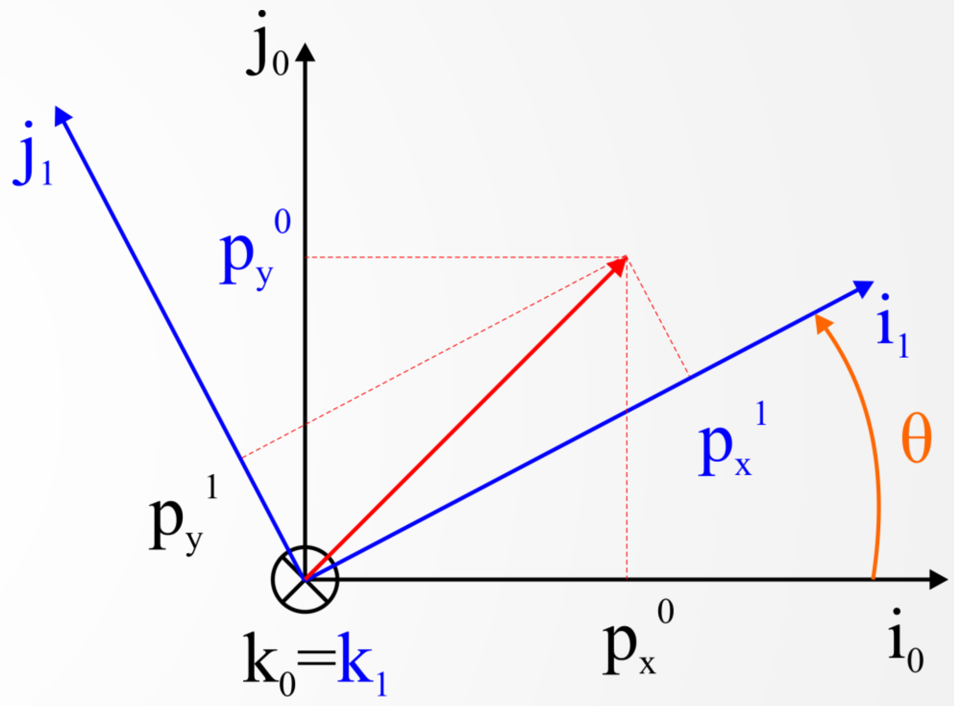


Rotation of Reference Frame

- Rotation around the k-axis

$$\mathbf{p} = p_x^0\mathbf{i}^0 + p_y^0\mathbf{j}^0 + p_z^0\mathbf{k}^0$$
$$\mathbf{p} = p_x^1\mathbf{i}^1 + p_y^1\mathbf{j}^1 + p_z^1\mathbf{k}^1$$
$$\mathbf{p}^1 = \begin{pmatrix} p_x^1 \\ p_y^1 \\ p_z^1 \end{pmatrix} = \begin{pmatrix} \mathbf{i}^1\mathbf{i}^0 & \mathbf{i}^1\mathbf{j}^0 & \mathbf{i}^1\mathbf{k}^0 \\ \mathbf{j}^1\mathbf{i}^0 & \mathbf{j}^1\mathbf{j}^0 & \mathbf{j}^1\mathbf{k}^0 \\ \mathbf{k}^1\mathbf{i}^0 & \mathbf{k}^1\mathbf{j}^0 & \mathbf{k}^1\mathbf{k}^0 \end{pmatrix} \begin{pmatrix} p_x^0 \\ p_y^0 \\ p_z^0 \end{pmatrix}$$

$$\mathbf{p}^1 = \mathcal{R}_0^1\mathbf{p}^0,\ \mathcal{R}_0^1 = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



Rotation of Reference Frame

- Rotation around the i-axis

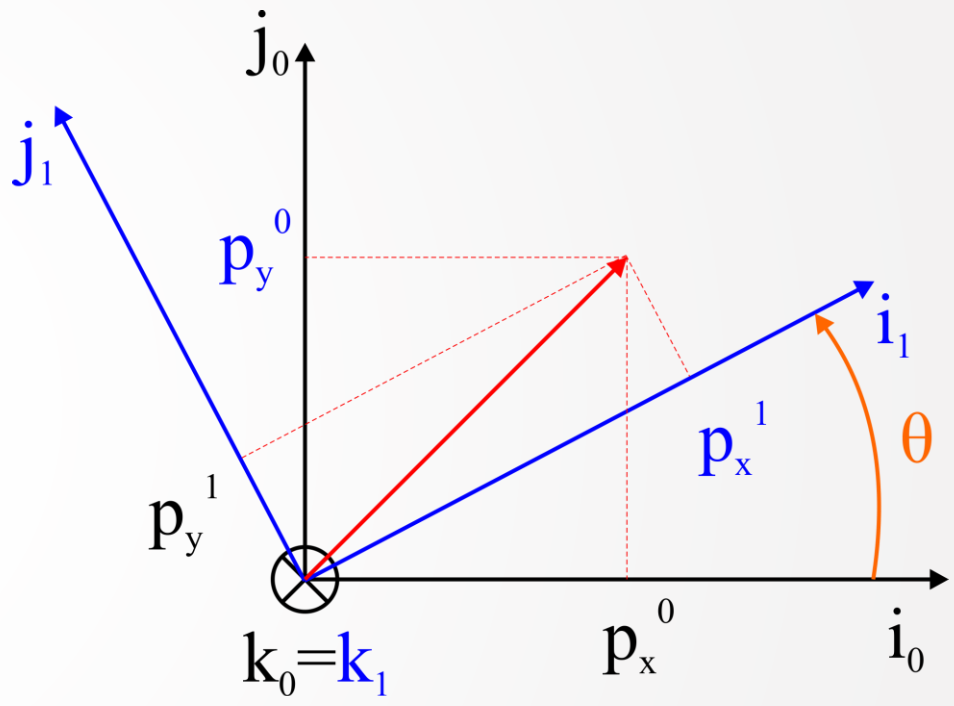
$$\mathcal{R}_0^1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & \sin\theta \\ 0 & -\sin\theta & \cos\theta \end{bmatrix}$$

- Rotation around the j-axis

$$\mathcal{R}_0^1 = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix}$$

- Rotation around the k-axis

$$\mathcal{R}_0^1 = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



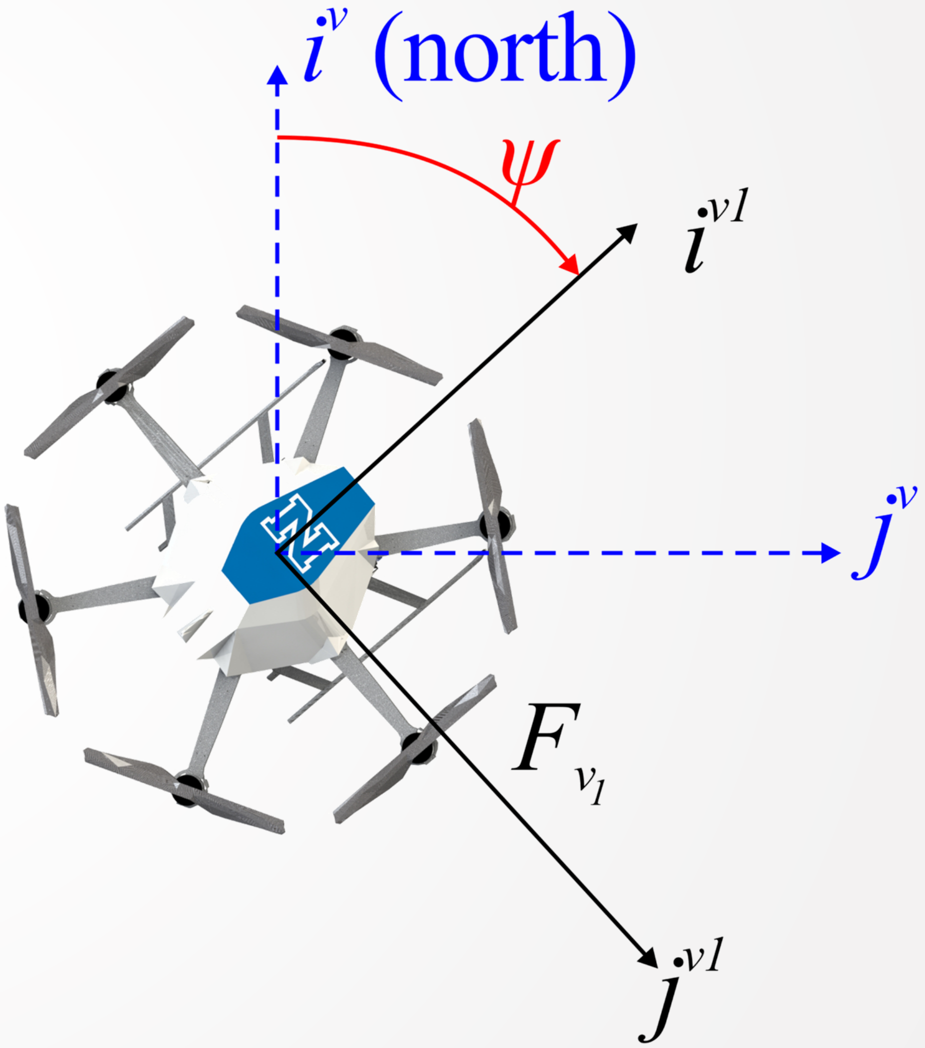
- Orthonormal matrix properties
  - $(\mathcal{R}_a^b)^-1 = (\mathcal{R}_a^b)^T = \mathcal{R}_b^a$
  - $\mathcal{R}_b^c \mathcal{R} a^b = \mathcal{R}_a^c$
  - $\det(\mathcal{R}_a^b) = 1$

Vehicle-1 Frame

$$\mathbf{p}^{v_1} = \mathcal{R}_v^{v_1}\mathbf{p}^v,$$

$$\mathcal{R}_v^{v_1} = \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- ψ represents the yaw angle

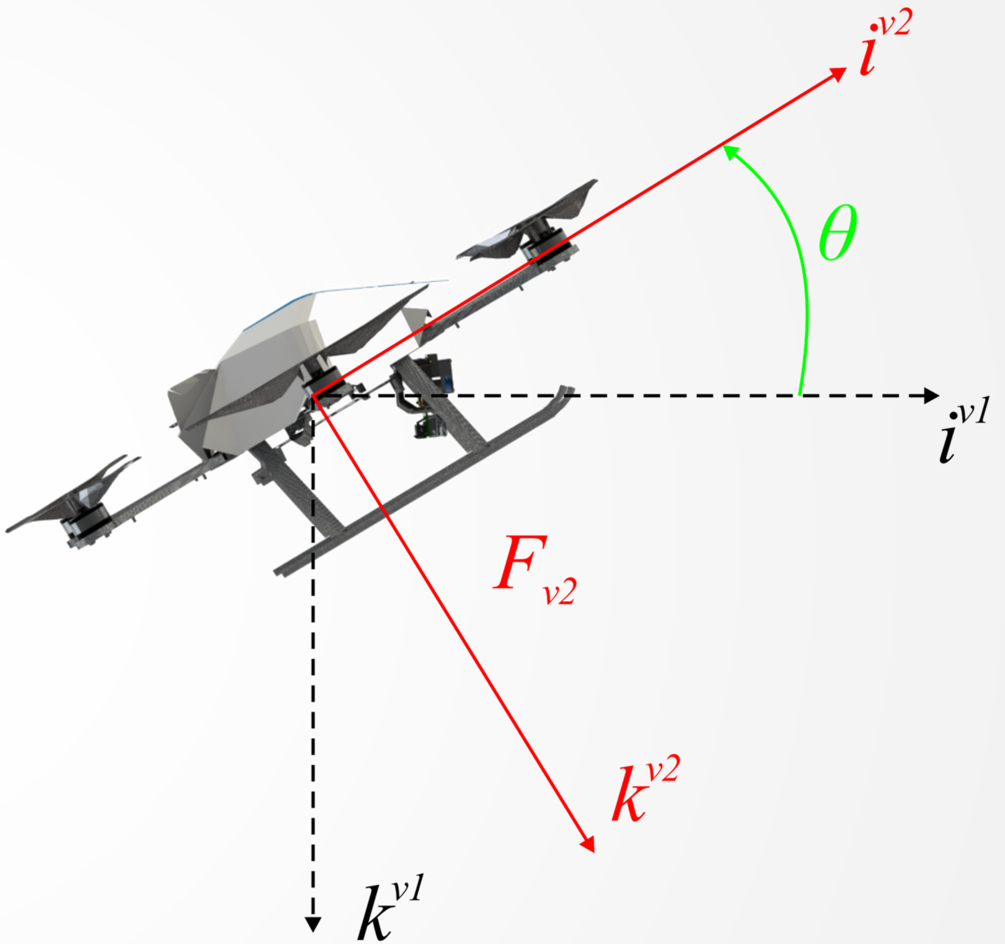


Vehicle-2 Frame

$$\mathbf{p}^{v_2} = \mathcal{R}_{v_1}^{v_2}\mathbf{p}^{v_1},$$

$$\mathcal{R}_{v_1}^{v_2} = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix}$$

- θ represents the pitch angle

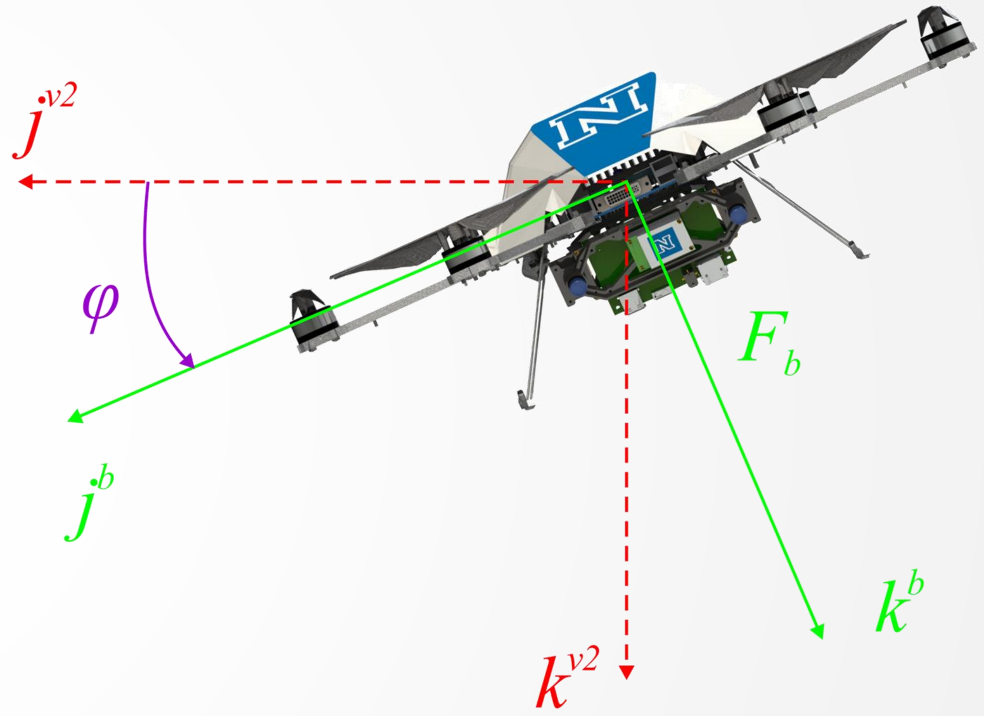


Body Frame

$$\mathbf{p}^b = \mathcal{R}_{v_2}^b \mathbf{p}^{v_2},$$

$$\mathcal{R}_{v_2}^b = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix}$$

- φ represents the roll angle



Inertial Frame to Body Frame

- Let:

$$\mathcal{R}_v^b(\phi,\theta,\psi) = \mathcal{R}_{v_2}(\phi)\mathcal{R}_{v_1}^{v_2}(\theta)\mathcal{R}_v^{v_1}(\psi)$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} c_\theta c_\psi & c_\theta s_\psi & -s_\theta \\ s_\phi s_\theta c_\psi - c_\phi s_\psi & s_\phi s_\theta s_\psi + c_\phi c_\psi & s_\phi c_\theta \\ c_\phi s_\theta c_\psi + s_\phi s_\psi & c_\phi s_\theta s_\psi - s_\phi c_\psi & c_\phi c_\theta \end{bmatrix}$$

- Then:

$$\mathbf{p}^b = \mathcal{R}_v^b \mathbf{p}^v$$

Relate Translational Velocity-Position

- Let *[u,v,w]* represent the body linear velocities

$$\frac{d}{dt}\begin{bmatrix} p_n \\ p_e \\ p_d \end{bmatrix} = \mathcal{R}_b^v \begin{bmatrix} u \\ v \\ w \end{bmatrix} = (\mathcal{R}_v^b)^T \begin{bmatrix} u \\ v \\ w \end{bmatrix}$$

- Which gives:

$$\begin{bmatrix} \dot{p}_n \\ \dot{p}_e \\ \dot{p}_d \end{bmatrix} = \begin{bmatrix} c_\theta c_\psi & s_\phi s_\theta c_\psi - c_\phi s_\phi & c_\phi s_\theta c_\psi + s_\phi s_\psi \\ c_\theta s_\psi & s_\phi s_\theta s_\psi + c_\phi c_\psi & c_\phi s_\theta s_\psi - s_{c\psi} \\ -s_\theta & s_\phi c_\theta & c_{c\theta} \end{bmatrix} \begin{bmatrix} u \\ v \\ w \end{bmatrix}$$

Body Rates – Euler Rates

- Let *[p,q,r]* denote the body angular rates

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} + \mathcal{R}_{v_2}^{b}(\phi) \begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + \mathcal{R}_{v_2}^{b}(\phi)\mathcal{R}_{v_1}^{v_2}(\theta) \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} =$$

$$\begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} =$$

$$\begin{bmatrix} 1 & 0 & -\sin\theta \\ 0 & \cos\phi & \sin\phi\cos\theta \\ 0 & -\sin\phi & \cos\phi\cos\theta \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix}$$

- Inverting this expression:

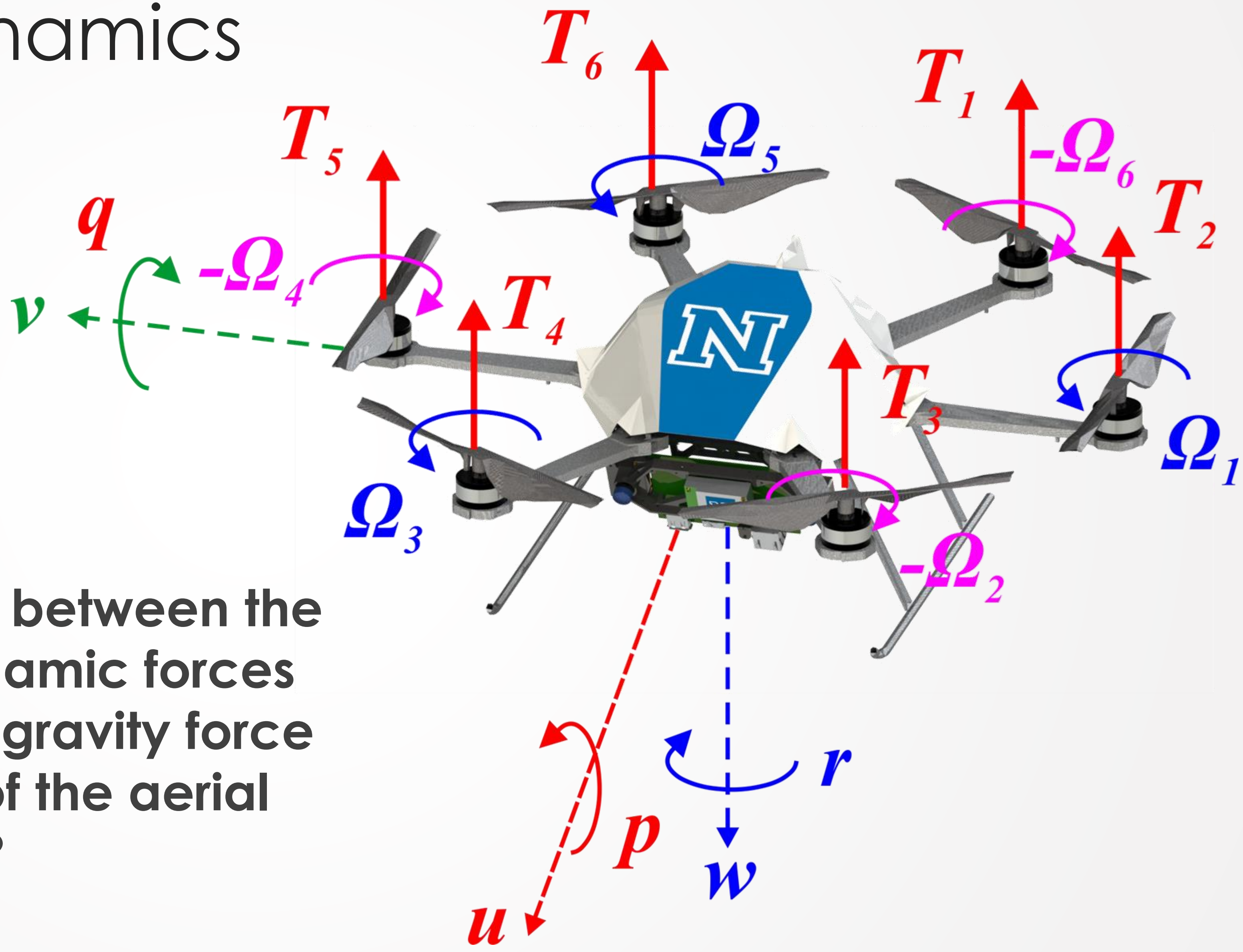
$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix}$$

Autonomous Mobile Robot Design

Topic: MAV Dynamics

Dr. Kostas Alexis (CSE)

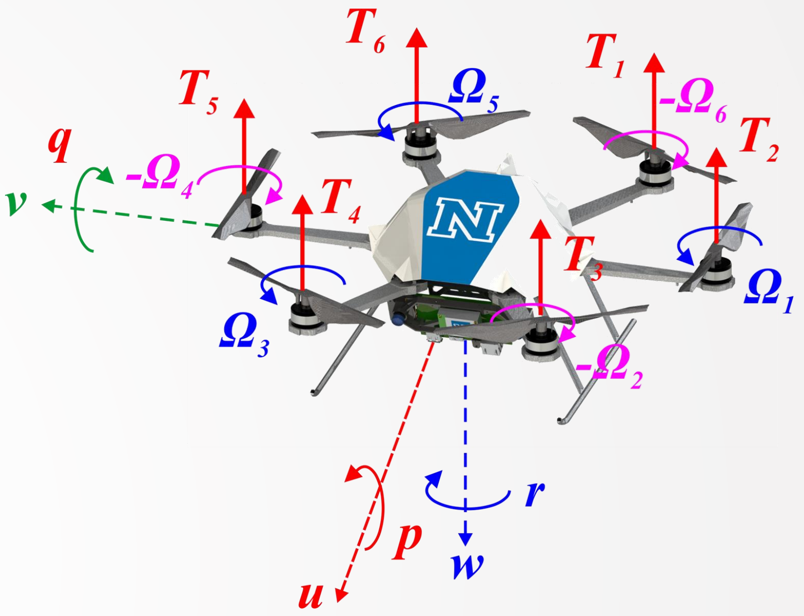
MAV Dynamics



What is the relation between the propeller aerodynamic forces and moments, the gravity force and the motion of the aerial robot?

MAV Dynamics

- Assumption 1: the Micro Aerial Vehicle is flying as a rigid body with negligible aerodynamic effects on it – for the employed airspeeds.
- The propeller is considered as a simple propeller disc that generates thrust and a moment around its shaft.



- Recall:

$$F_T = T = C_T \rho A (\Omega R)^2$$

$$M_Q = Q = C_Q \rho A (\Omega R)^2 R$$

- And let us write:

$$T_i = k_n \Omega_i^2$$

$$M_i = (-1)^{i-1} k_m T_i$$

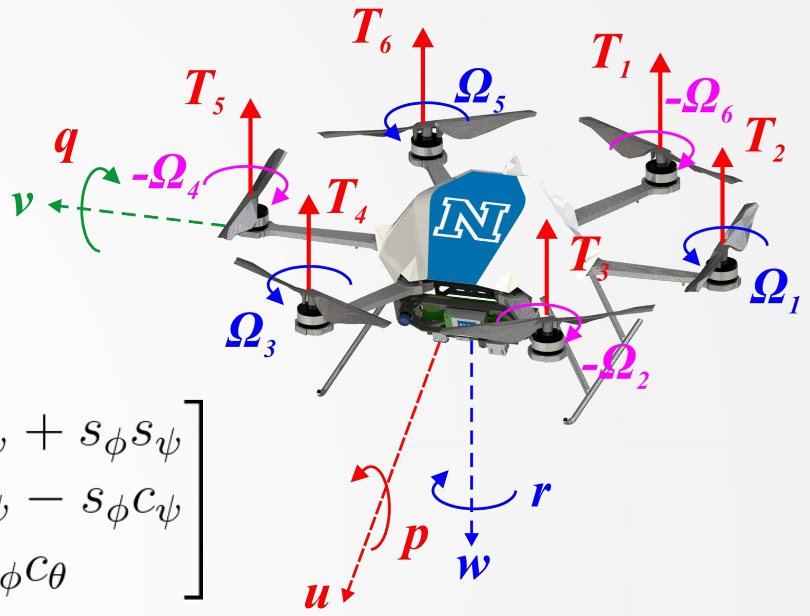
MAV Dynamics

- Recall the kinematic equations:
  - Translational Kinematic Expression:

$$\begin{bmatrix}\dot{p}_n\\ \dot{p}_e\\ \dot{p}_d\end{bmatrix} = \mathcal{R}_b^v \begin{bmatrix}u\\ v\\ w\end{bmatrix}, \quad \mathcal{R}_b^v = \begin{bmatrix} c_\theta c_\psi & s_\phi s_\theta c_\psi - c_\phi s_\psi & c_\phi s_\theta c_\psi + s_\phi s_\psi \\ c_\theta s_\psi & s_\phi s_\theta s_\psi + c_\phi c_\psi & c_\phi s_\theta s_\psi - s_\phi c_\psi \\ -s_\theta & s_\phi c_\theta & c_\phi c_\theta \end{bmatrix}$$

  - Rotational Kinematic Expression

$$\begin{bmatrix}\dot{\phi}\\ \dot{\theta}\\ \dot{\psi}\end{bmatrix} = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta \end{bmatrix} \begin{bmatrix}p\\ q\\ r\end{bmatrix}$$



MAV Dynamics

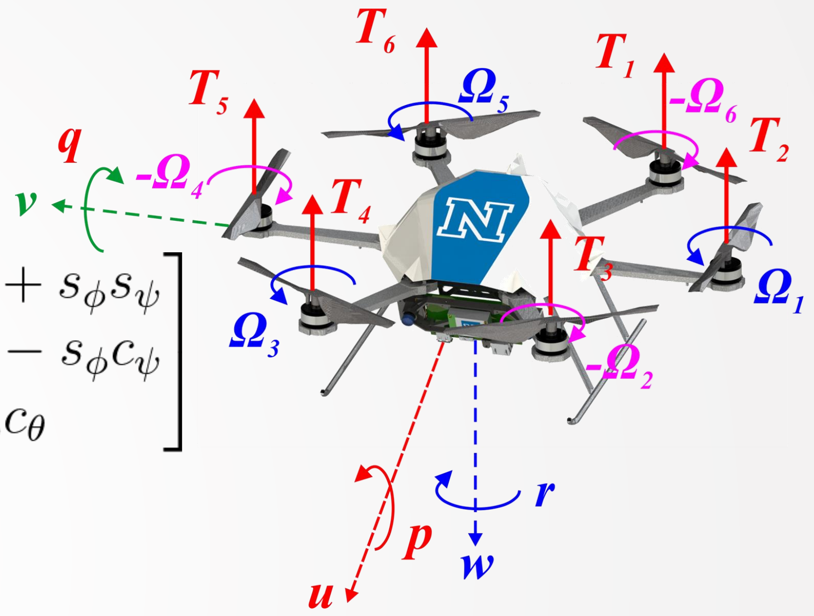
- To append the forces and moments we need to combine their formulation with

$$\begin{bmatrix}\dot{p}_n\\ \dot{p}_e\\ \dot{p}_d\end{bmatrix} = \mathcal{R}_b^v \begin{bmatrix}u\\ v\\ w\end{bmatrix},\ \mathcal{R}_b^v = \begin{bmatrix} c_\theta c_\psi & s_\phi s_\theta c_\psi - c_\phi s_\psi & c_\phi s_\theta c_\psi + s_\phi s_\psi \\ c_\theta s_\psi & s_\phi s_\theta s_\psi + c_\phi c_\psi & c_\phi s_\theta s_\psi - s_\phi c_\psi \\ -s_\theta & s_\phi c_\theta & c_\phi c_\theta \end{bmatrix}$$

$$\begin{bmatrix}\dot{u}\\ \dot{v}\\ \dot{w}\end{bmatrix} = \begin{bmatrix} rv - qw \\ pw - ru \\ qu - pv \end{bmatrix} + \frac{1}{m}\begin{bmatrix} f_x \\ f_y \\ f_z \end{bmatrix}$$

$$\begin{bmatrix}\dot{\phi}\\ \dot{\theta}\\ \dot{\psi}\end{bmatrix} = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix}$$

$$\begin{bmatrix}\dot{p}\\ \dot{q}\\ \dot{r}\end{bmatrix} = \begin{bmatrix} \frac{J_y - J_z}{J_x} qr \\ \frac{J_z - J_x}{J_y} pr \\ \frac{J_x - J_y}{J_z} qr \end{bmatrix} + \begin{bmatrix} \frac{1}{m} M_x \\ \frac{1}{m} M_y \\ \frac{1}{z} M_z \end{bmatrix}$$

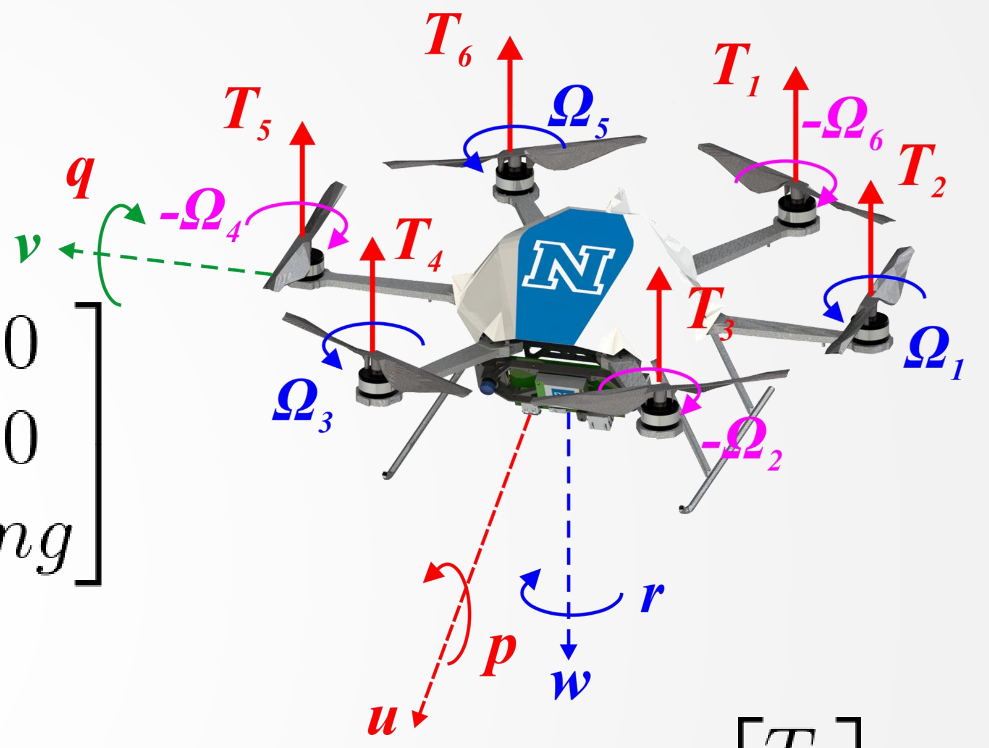


- ***Next step: append the MAV forces and moments***

MAV Dynamics

- MAV forces in the body frame:

$$\mathbf{f}_b = \begin{bmatrix} f_x \\ f_y \\ f_z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \sum_{i=1}^{6} T_i \end{bmatrix} - \mathcal{R}_v^b \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix}$$



- Moments in the body frame:

$$\mathbf{m}_b = \begin{bmatrix} M_x \\ M_y \\ M_z \end{bmatrix} = \begin{bmatrix} ls_{30} & l & ls_{30} & -ls_{30} & -l & ls_{30} \\ -lc_{60} & 0 & lc_{60} & lc_{60} & 0 & -lc_{60} \\ -k_m & k_m & -k_m & k_m & -k_m & k_m \end{bmatrix} \begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \\ T_5 \\ T_6 \end{bmatrix}$$

Autonomous Robot Challenges

Where am I?
What is my environment?

Autonomous Mobile Robot Design

Topic: Preliminaries on state estimation

Dr. Kostas Alexis (CSE)

Navigation Sensors

- Providing the capacity to estimate the **state** of the aerial robot
  - **Self-Localize and estimate its pose in the environment**
  - **Often this requires to also derive the map of the environment**
  - **In some cases also rely in external systems (e.g. GPS), while a lot of work is undergoing into making aerial robots completely autonomous.**

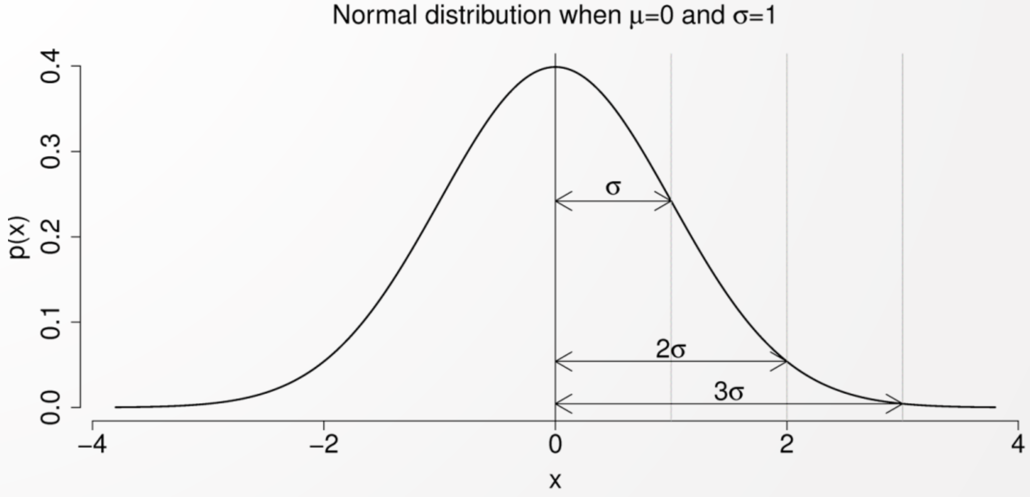
Classification of Sensors

- *What:*
  - **Proprioceptive sensors**
    - Measure values internally to the robot.
      - Angular rate, heading.
  - **Exteroceptive sensors**
    - Information from the robot environment
      - Distances to objects, extraction of features from the environment.

- *How:*
  - **Passive Sensors**
    - Measure energy coming from a signal of the environment – very much influenced from the environment.
  - **Active Sensors**
    - Emit their proper energy and measure reaction.
    - Better performance, but some influence on the environment.
    - Not always easily applicable concept.

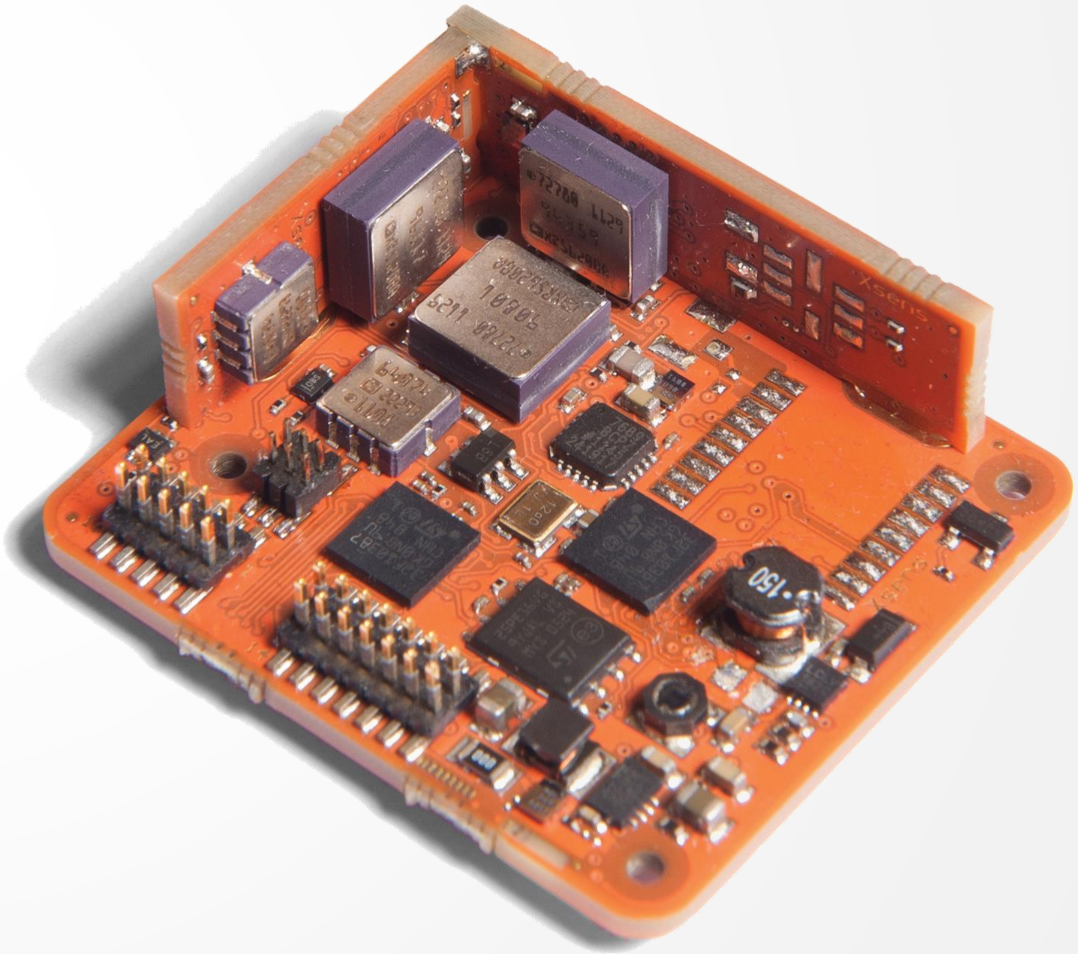
Uncertainty Representation

- **Sensing is always related to uncertainties**
  - How can uncertainty be represented or quantified?
  - How do they propagate – uncertainty of a function of uncertain values?
- **Systematic errors**
  - They are caused by factors or processes that can in theory be modeled and, thus, calibrated, (for example the misalignment of a 3-axes accelerometer)
- **Random errors**
  - They cannot be predicted using a sophisticated model but can only be described in probabilistic terms



Typical Navigation Sensors

- The following sensors are commonly used for the navigation of aerial robots:
  - **Inertial Sensors:**
    - Accelerometers
    - Gyroscopes
  - **Magnetometers (digital compass)**
  - **Pressure Sensors**
    - Barometric pressure for altitude sensing
    - Airspeed measurements
  - **GPS**
  - **Camera based systems**
  - **Time-of-Flight sensors**



World state (or system state)

- *Belief state:*
  - **Our belief/estimate of the world state**
- *World state:*
  - **Real state of the robot in the real world**

Autonomous Mobile Robot Design

Topic: State Estimation – Recap on Probabilities

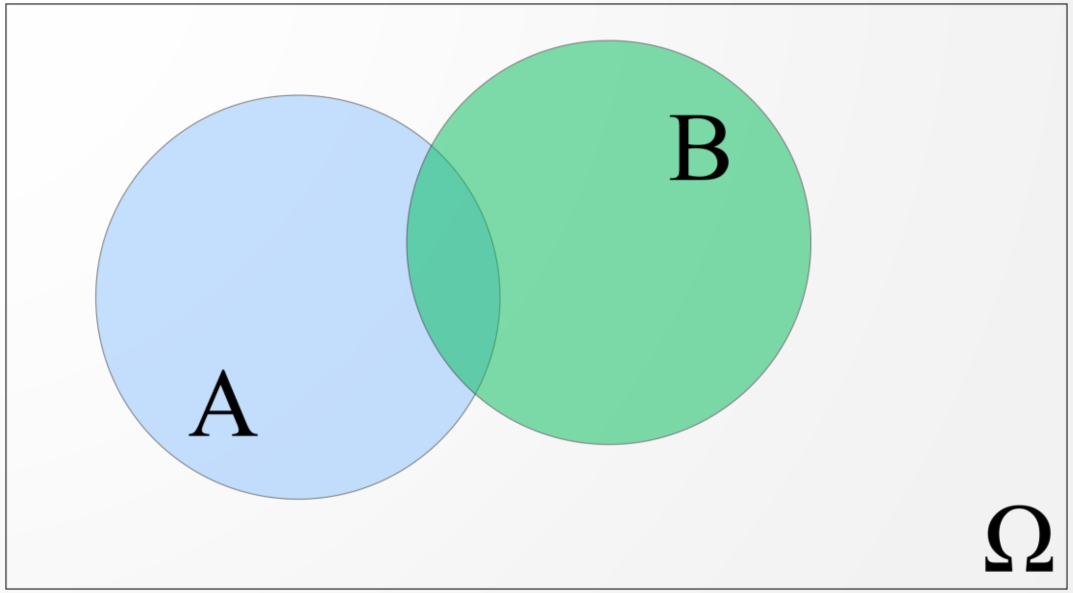
Dr. Kostas Alexis (CSE)

Probability theory

- **Random experiment** that can produce a number of outcomes, e.g. a rolling dice.
- Sample space, e.g.: $\{1,2,3,4,5,6\}$
- Event A is subset of outcomes, e.g. $\{1,3,5\}$
- Probability $P(A)$, e.g. $P(A)=0.5$

Axioms of Probability theory

- $0 \leq P(A) \leq 1$
- $P(\Omega) = 1, \; P(\emptyset) = 0$
- $P(A \cup B) = P(A) + P(B) - P(A \cap B)$



Discrete Random Variables

- X denotes a random variable
- X can take on a countable number of values in $\{x_1,x_2,\ldots,x_n\}$
- $P(X=x_i)$ is the probability that the random variable X takes on value x_i
- $P(.)$ is called the probability mass function

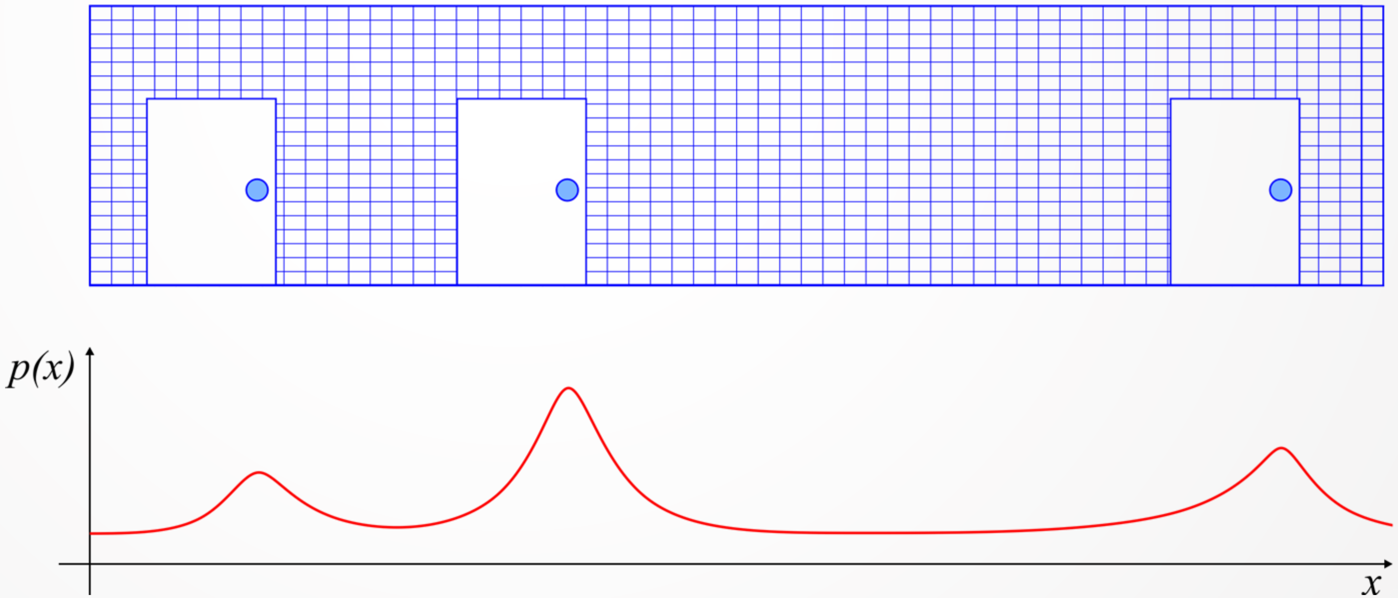
- Example: P(Room)=<0.6,0.3,0.06,0.03>, Room one of the office, corridor, lab, kitchen

Continuous Random Variables

- X takes on continuous values.
- P(X=x) or P(x) is called the **probability density function (PDF)**.

Thrun, Burgard, Fox, "Probabilistic Robotics", MIT Press, 2005

- Example:



Proper Distributions Sum To One

- Discrete Case

$$\sum_{x} P(x) = 1$$

- Continuous Case

$$\int p(x)dx = 1$$

Joint and Conditional Probabilities

- $p(X = x, \text{ and } Y = y) = P(x, y)$
- If X and Y are **independent** then:

$$P(x, y) = P(x)P(y)$$

- Is the probability of **x given y**

$$P(x|y)P(y) = P(x, y)$$

- If X and Y are independent then:

$$P(x|y) = P(x)$$

Conditional Independence

- Definition of conditional independence:

$$P(x, y|z) = P(x|z)P(y|z)$$

- Equivalent to:

$$P(x|z) = P(x|y, z)$$
$$P(y|z) = P(y|x, z)$$

- Note: this does not necessarily mean that:

$$P(x, y) = P(x)P(y)$$

Marginalization

- **Discrete case:** $$P(x) = \sum_y P(x, y)$$
- **Continuous case:** $$p(x) = \int p(x, y) dy$$

Marginalization example

P(X,Y)	x1	x1	x1	x1	P(Y) ↓
y1	1/8	1/16	1/32	1/32	1/4
y1	1/16	1/8	1/32	1/32	1/4
y1	1/16	1/16	1/16	1/16	1/4
y1	1/4	0	0	0	1/4
P(X) →	1/2	1/4	1/8	1/8	1

Expected value of a Random Variable

- **Discrete case:** $$E[X] = \sum_i x_i P(x_i)$$

- **Continuous case:** $$E[X] = \int x P(X = x) dx$$

- The expected value is the weighted average of all values a random variable can take on.
- Expectation is a linear operator:

$$E[aX + b] = aE[X] + b$$

Covariance of a Random Variable

- Measures the **square expected deviation from the mean**:

$$Cov[X] = E[X - E[X]]^2 = E[X^2] - E[X]^2$$

Estimation from Data

- Observations: $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n \in \mathcal{R}^d$
- Sample Mean: $\mu = \frac{1}{n} \sum_i \mathbf{x}_i$
- Sample Covariance:

$$\Sigma = \frac{1}{n-1} \sum_i (\mathbf{x}_i - \mu)(\mathbf{x}_i - \mu)$$

Autonomous Mobile Robot Design

Topic: State Estimation – Reasoning with Bayes Law

Dr. Kostas Alexis (CSE)

The State Estimation problem

- We want to estimate the world state x from:
  - Sensor measurements z and
  - Controls u

- We need to model the relationship between these random variables, i.e:

$$p(\mathbf{x}|\mathbf{z}) \qquad\qquad p(\mathbf{x}'|\mathbf{x}, \mathbf{u})$$

Causal vs. Diagnostic Reasoning

$P(\mathbf{x}|\mathbf{z})$ Is diagnostic

$P(\mathbf{z}|\mathbf{x})$ Is causal

- Diagnostic reasoning is typically what we need.
- Often causal knowledge is easier to obtain.
- Bayes rule allows us to use causal knowledge in diagnostic reasoning.

Bayes rule

- Definition of **conditional probability**:

$$P(x, z) = P(x|z)P(z) = P(z|x)P(x)$$

- **Bayes rule:**

Observation likelihood

Prior on world state

$$P(x|z) = \frac{P(z|x)P(x)}{P(z)}$$

Prior on sensor observations

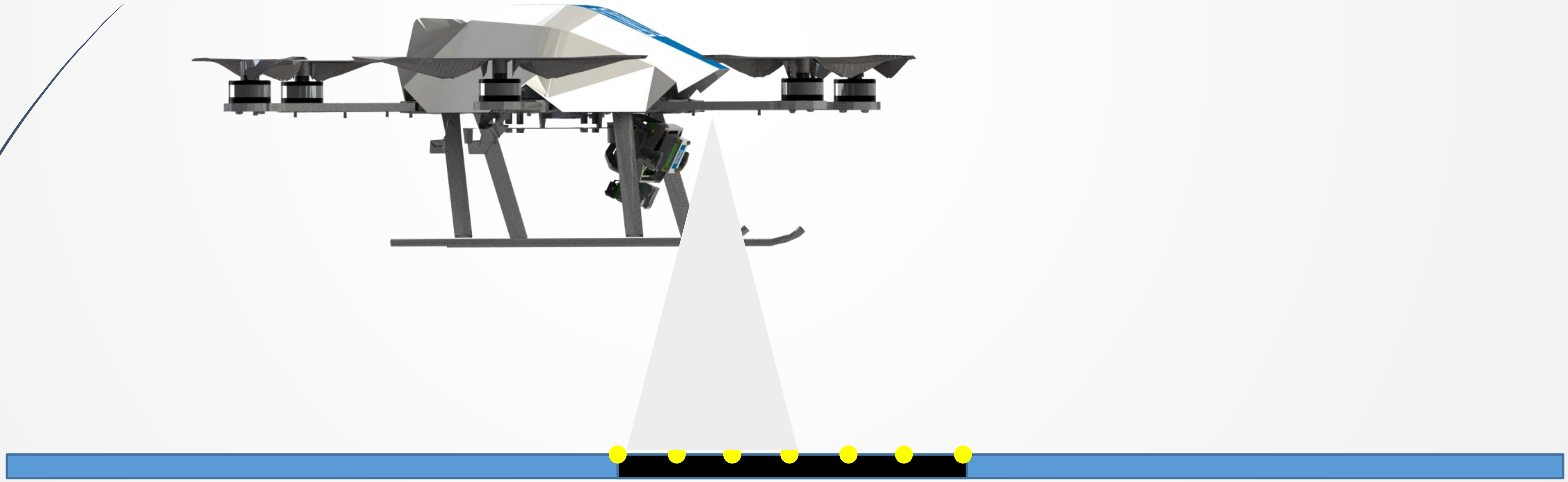
Normalization

- Direct computation of $P(\mathbf{z})$ can be difficult.
- Idea: compute improper distribution, normalize afterwards.

- **STEP 1:** $$L(x|z) = P(z|x)P(x)$$
- **STEP 2:** $$P(z) = \sum_x P(z,x) = \sum_x P(z|x)P(x) = \sum_x L(x|z)$$
- **STEP 3:** $$P(x|z) = L(x|z)/P(z)$$

Example: Sensor Measurement

- Quadrotor seeks the Landing Zone
- The landing zone is marked with many bright lamps
- The quadrotor has a light sensor.



Example: Sensor Measurement

- Binary sensor $Z \in \{bright, \bar{bright}\}$
- Binary world state $X \in \{home, \bar{home}\}$
- Sensor model $$P(Z = bright | X = home) = 0.6$$ $$P(Z = bright | X = \bar{home}) = 0.3$$
- Prior on world state $P(X = home) = 0.5$
- Assume: robot observes light, i.e. $Z = bright$
- What is the probability $P(X = home | Z = bright)$ that the robot is above the landing zone.

Example: Sensor Measurement

- Sensor model:

$$P(Z = bright|X = home) = 0.6$$
$$P(Z = bright|X = \bar{home}) = 0.3$$

- Prior on world state: $P(X = home) = 0.5$

- Probability after observation (using Bayes):

$$P(X = home|Z = bright) = \frac{P(bright|home)P(home)}{P(bright|home)P(home) + P(bright|\bar{home})P(\bar{home})} = \frac{0.6 \cdot 0.5}{0.6 \cdot 0.5 + 0.3 \cdot 0.5} = 0.67$$

Autonomous Mobile Robot Design

Topic: State Estimation – Bayes Filter

Dr. Kostas Alexis (CSE)

Markov Assumption

- Observations depend only on current state

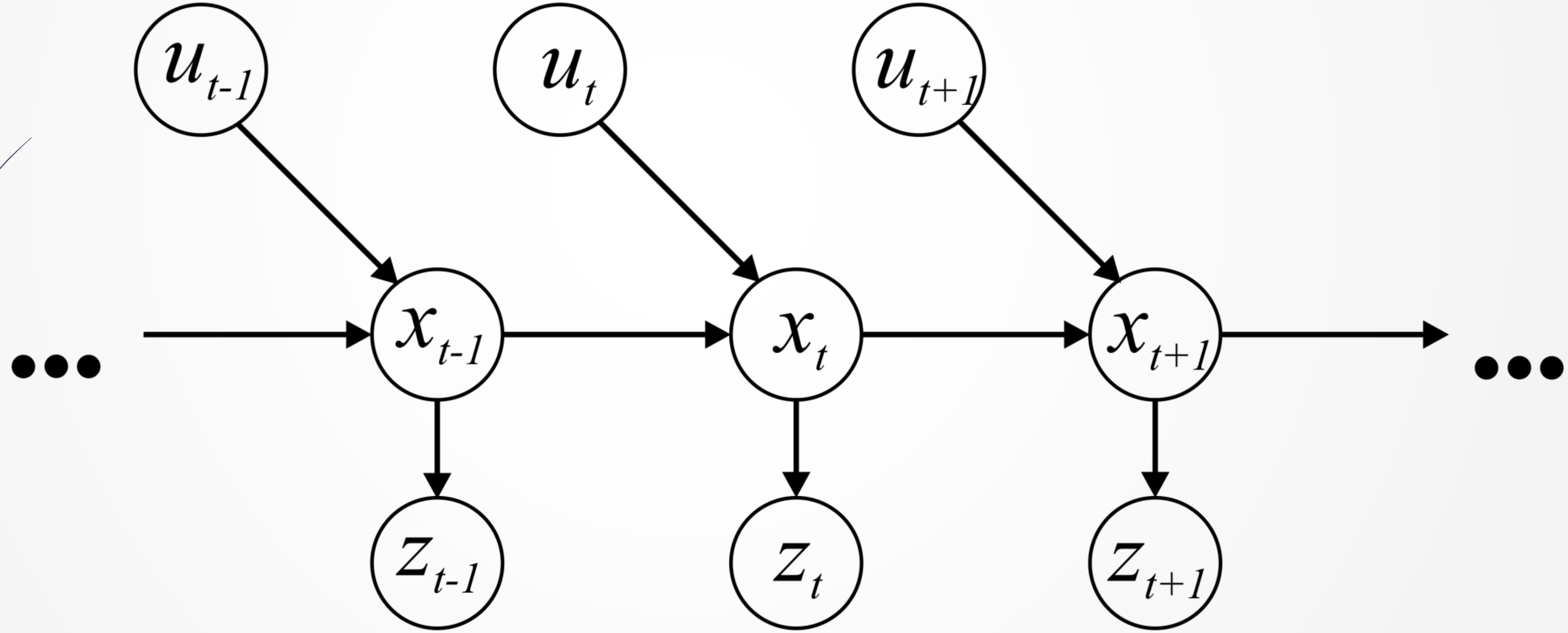
$$P(z_t|x_{0:t}, z_{1:t-1}, u_{1:t}) = P(z_t|x_t)$$

- Current state depends only on previous state and current action

$$P(x_t|x_{0:t}, z_{1:t}, u_{1:t}) = P(x_t|x_{t-1}, u_t)$$

Markov Chain

- A Markov Chain is a stochastic process where, given the present state, the past and the future states are independent.



Underlying Assumptions

- Static world
- Independent noise
- Perfect model, no approximation errors

Bayes Filter

- **Given**
  - Sequence of observations and actions: z_t, u_t
  - Sensor model: $P(z|x)$
  - Action model: $P(x'|x, u)$
  - Prior probability of the system state: $P(x)$
- **Desired**
  - Estimate of the state of the dynamic system: x
  - Posterior of the state is also called belief:

$$Bel(x_t) = P(x_t|u_1, z_1, \ldots, u_t, z_t)$$

Bayes Filter Algorithm

- **For each time step, do:**
  - Apply motion model:

$$\overline{Bel}(x_t) = \sum_{x_t - 1} P(x_t | x_{t-1}, u_t) Bel(x_{t-1})$$

  - Apply sensor model:

$$Bel(x_t) = \eta P(z_t | x_t) \overline{Bel}(x_t)$$

  - η is a normalization factor to ensure that the probability is maximum 1.

Autonomous Mobile Robot Design

Topic: State Estimation – Kalman Filter

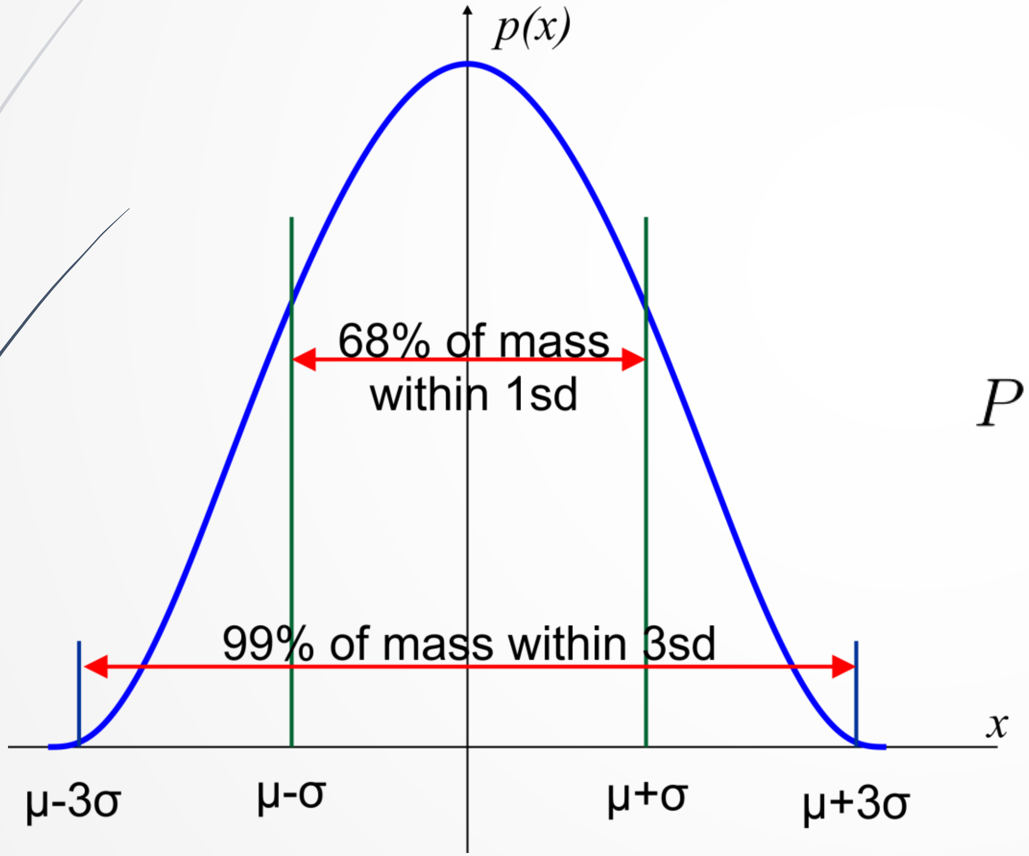
Dr. Kostas Alexis (CSE)

Kalman Filter

- Bayes filter is a useful tool for state estimation.
- Histogram filter with grid representation is not very efficient.
- How can we represent the state more efficiently?

Kalman Filter

- Univariate distribution



$$X \sim \mathcal{N}(\mu, \sigma^2)$$

mean

Variance (squared standard deviation)

$$P(X = x) = \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2})$$

Kalman Filter

- Multivariate normal distribution: $\mathbf{X} \sim \mathcal{N}(\mu, \mathbf{\Sigma})$
- Mean: $\mu \in \mathcal{R}^n$
- Covariance: $\mathbf{\Sigma} \in \mathbf{R}^{n \times m}$
- Probability density function:

$$p(\mathbf{X} = \mathbf{x}) = \mathcal{N}(\mathbf{x}; \mu, \mathbf{\Sigma}) =$$

$$\frac{1}{(2\pi)^{n/2}|\mathbf{\Sigma}|^{1/2}} \exp(-\frac{1}{2}(\mathbf{x} - \mu)^T \mathbf{\Sigma}^{-1}(\mathbf{x} - \mu))$$

Properties of Normal Distributions

- Linear transformation – remains Gaussian

$$\mathbf{X} \sim \mathcal{N}(\mu, \mathbf{\Sigma}), \mathbf{Y} \sim \mathbf{A}\mathbf{X} + \mathbf{B}$$
$$\Rightarrow \mathbf{Y} \sim \mathcal{N}(\mathbf{A}\mu + \mathbf{B}, \mathbf{A}\mathbf{\Sigma}\mathbf{A}^T)$$

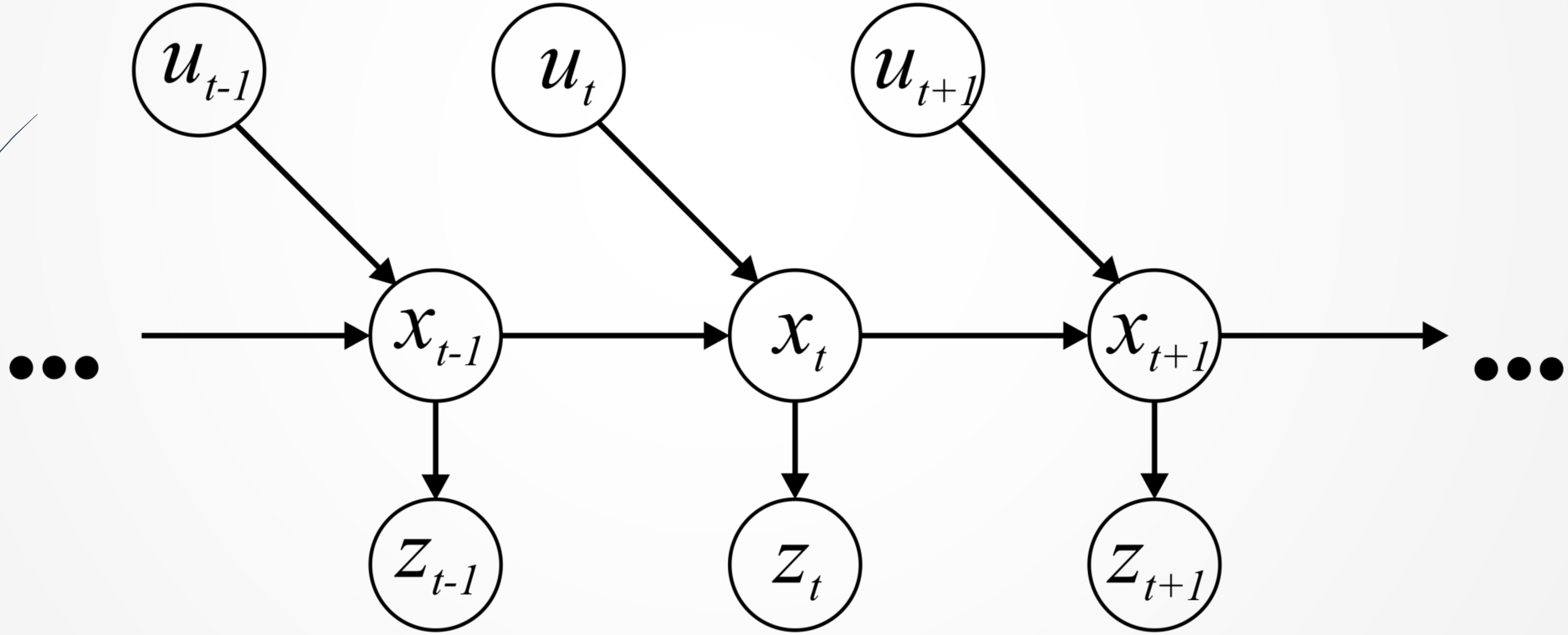
- Intersection of two Gaussians – remains Gaussian

$$\mathbf{X}_1 \sim \mathcal{N}(\boldsymbol{\mu}_1, \mathbf{\Sigma}_1), \mathbf{X}_2 \sim \mathcal{N}(\boldsymbol{\mu}_2, \mathbf{\Sigma}_2)$$

$$\boxed{p(\mathbf{X}_1)p(\mathbf{X}_2) = \mathcal{N}\left(\frac{\mathbf{\Sigma}_2}{\mathbf{\Sigma}_1 + \mathbf{\Sigma}_2}\boldsymbol{\mu}_1 + \frac{\mathbf{\Sigma}_1}{\mathbf{\Sigma}_1 + \mathbf{\Sigma}_2}\boldsymbol{\mu}_2, \frac{1}{\mathbf{\Sigma}_1^{-1} + \mathbf{\Sigma}_2^{-1}}\right)}$$

Linear Process Model

- Consider a time-discrete stochastic process (Markov chain)



Linear Process Model

- Consider a time-discrete stochastic process
- Represent the estimated state (belief) with a Gaussian

$$\mathbf{x}_t \sim \mathcal{N}(\mu_t, \mathbf{\Sigma}_t)$$

- Assume that the system evolves linearly over time, then depends linearly on the controls, and has zero-mean, normally distributed process noise

$$\mathbf{x}_t = \mathbf{A}\mathbf{x}_{t-1} + \mathbf{B}\mathbf{u}_t + \epsilon_t$$

- With $\epsilon_t \sim \mathcal{N}(\mathbf{0}, \mathbf{Q})$

Linear Observations

- Further, assume we make observations that depend linearly on the state and that are perturbed zero-mean, normally distributed observation noise

$$\mathbf{z}_t = \mathbf{C}\mathbf{x}_t + \delta_t$$

- With $\delta_t \sim \mathcal{N}(\mathbf{0}, \mathbf{R})$

Kalman Filter

- Estimates the state x_t of a discrete-time controlled process that is governed by the linear stochastic difference equation

$$\mathbf{x}_t = \mathbf{A}\mathbf{x}_{t-1} + \mathbf{B}\mathbf{u}_t + \epsilon_t$$

- And (linear) measurements of the state

$$\mathbf{z}_t = \mathbf{C}\mathbf{x}_t + \delta_t$$

- With $\epsilon_t \sim \mathcal{N}(\mathbf{0}, \mathbf{Q})$ and $\delta_t \sim \mathcal{N}(\mathbf{0}, \mathbf{R})$

Kalman Filter

- State $\mathbf{x} \in \mathbb{R}^n$
- Controls $\mathbf{u} \in \mathbb{R}^l$
- Observations $\mathbf{z} \in \mathbb{R}^k$
- Process equation $\mathbf{x}_t = \underset{nxn}{\mathbf{A}}\mathbf{x}_{t-1} + \underset{nxl}{\mathbf{B}}\mathbf{u}_t + \epsilon_t$
- Measurement equation $\mathbf{z}_t = \underset{nxk}{\mathbf{C}}\mathbf{x}_t + \delta_t$

Kalman Filter

- Initial belief is Gaussian

$$Bel(x_0) = \mathcal{N}(\mathbf{x}_0; \mu_0, \mathbf{\Sigma}_0)$$

- Next state is also Gaussian (linear transformation)

$$\mathbf{x}_t \sim \mathcal{N}(\mathbf{A}\mathbf{x}_t + \mathbf{B}\mathbf{u}_t, \mathbf{Q})$$

- Observations are also Gaussian

$$\mathbf{z}_t \sim \mathcal{N}(\mathbf{C}\mathbf{x}_t, \mathbf{R})$$

Recall: Bayes Filter Algorithm

- For each step, do:
  - Apply motion model

$$\overline{Bel}(\mathbf{x}_t) = \int p(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{u}_t) Bel(\mathbf{x}_{t-1}) d\mathbf{x}_{t-1}$$

  - Apply sensor model

$$Bel(\mathbf{x}_t) = \eta p(\mathbf{z}_t | \mathbf{x}_t) \overline{Bel}(\mathbf{x}_t)$$

From Bayes Filter to Kalman Filter

- For each step, do:
  - Apply motion model

$$\overline{Bel}(\mathbf{x}_t) = \int \underbrace{p(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{u}_t)}_{\mathcal{N}(\mathbf{x}_t; \mathbf{A}\mathbf{x}_{t-1} + \mathbf{B}\mathbf{u}_k t, \mathbf{Q})} \underbrace{Bel(\mathbf{x}_{t-1})}_{\mathcal{N}(\mathbf{x}_{t-1}; \mu_{t-1}, \mathbf{\Sigma}_{t-1})} d\mathbf{x}_{t-1}$$

$$= \mathcal{N}(\mathbf{x}_t; \mathbf{A}\mu_{t-1} + \mathbf{B}\mathbf{u}_t, \mathbf{A}\Sigma\mathbf{A}^T + \mathbf{Q})$$

$$= \mathcal{N}(\mathbf{x}_t; \bar{\mu}_t, \bar{\Sigma}_t)$$

From Bayes Filter to Kalman Filter

- For each step, do:
  - Apply sensor model

$$
\begin{aligned}
\overline{Bel}(\mathbf{x}_t) &= \eta \underbrace{p(\mathbf{z}_t|\mathbf{x}_t)}_{\mathcal{N}(\mathbf{z}_t;\mathbf{C}\mathbf{x}_t,\mathbf{R})} \underbrace{\overline{Bel}(\mathbf{x}_t)}_{\mathcal{N}(\mathbf{x}_t;\bar{\mu}_t,\bar{\mathbf{\Sigma}}_t)} \\
&= \mathcal{N}(\mathbf{x}_t; \bar{\mu}_t + \mathbf{K}_t(\mathbf{z}_t - \mathbf{C}\bar{\mu}), (\mathbf{I} - \mathbf{K}_t)\mathbf{C})\bar{\mathbf{\Sigma}}) \\
&= \mathcal{N}(x_t; \mu_t, \mathbf{\Sigma}_t)
\end{aligned}
$$

- With $\mathbf{K}_t = \bar{\mathbf{\Sigma}}_t \mathbf{C}^T (\mathbf{C}\bar{\mathbf{\Sigma}}_t \mathbf{C}^T + \mathbf{R})^{-1}$ (**Kalman Gain**)

From Bayes Filter to Kalman Filter

Blends between our previous estimate $\bar{\mu}_t$ and the discrepancy between our sensor observations and our predictions.

The degree to which we believe in our sensor observations is the Kalman Gain. And this depends on a formula based on the errors of sensing etc. In fact it depends on the ratio between our uncertainty Σ and the uncertainty of our sensor observations R.

$$\underbrace{\bar{\mu}_t}_{\text{old mean}} + \underbrace{\mathbf{K}_t}_{\text{Kalman Gain}}(\mathbf{z}_t - \mathbf{C}\bar{\mu})$$

Kalman Filter Algorithm

Prediction & Correction steps can happen in any order.

- For each step, do:
  - Apply motion model (**prediction step**)

$$\bar{\boldsymbol{\mu}}_t = \mathbf{A}\boldsymbol{\mu}_{t-1} + \mathbf{B}\mathbf{u}_t$$

$$\bar{\boldsymbol{\Sigma}}_t = \mathbf{A}\boldsymbol{\Sigma}\mathbf{A}^\top + \mathbf{Q}$$

  - Apply sensor model (**correction step**)

$$\boldsymbol{\mu}_t = \bar{\boldsymbol{\mu}}_t + \mathbf{K}_t(\mathbf{z}_t - \mathbf{C}\bar{\boldsymbol{\mu}}_t)$$

$$\boldsymbol{\Sigma}_t = (\mathbf{I} - \mathbf{K}_t\mathbf{C})\bar{\boldsymbol{\Sigma}}_t$$

    - With $\mathbf{K}_t = \bar{\boldsymbol{\Sigma}}_t\mathbf{C}^\top(\mathbf{C}\bar{\boldsymbol{\Sigma}}_t\mathbf{C}^\top + \mathbf{R})^{-1}$

Kalman Filter Algorithm

Prediction & Correction steps can happen in any order.

Prediction

$$\bar{\boldsymbol{\mu}}_t = \mathbf{A}\boldsymbol{\mu}_{t-1} + \mathbf{B}\mathbf{u}_t$$

$$\bar{\Sigma}_t = \mathbf{A}\Sigma\mathbf{A}^\top + \mathbf{Q}$$

Correction

$$\boldsymbol{\mu}_t = \bar{\boldsymbol{\mu}}_t + \mathbf{K}_t(\mathbf{z}_t - \mathbf{C}\bar{\boldsymbol{\mu}}_t)$$

$$\Sigma_t = (\mathbf{I} - \mathbf{K}_t\mathbf{C})\bar{\Sigma}_t$$

$$\mathbf{K}_t = \bar{\Sigma}_t\mathbf{C}^\top(\mathbf{C}\bar{\Sigma}_t\mathbf{C}^\top + \mathbf{R})^{-1}$$

Autonomous Mobile Robot Design

Topic: Extended Kalman Filter

Dr. Kostas Alexis (CSE)

These slides relied on the lectures from C. Stachniss, J. Sturm and the book "Probabilistic Robotics" from Thurn et al.

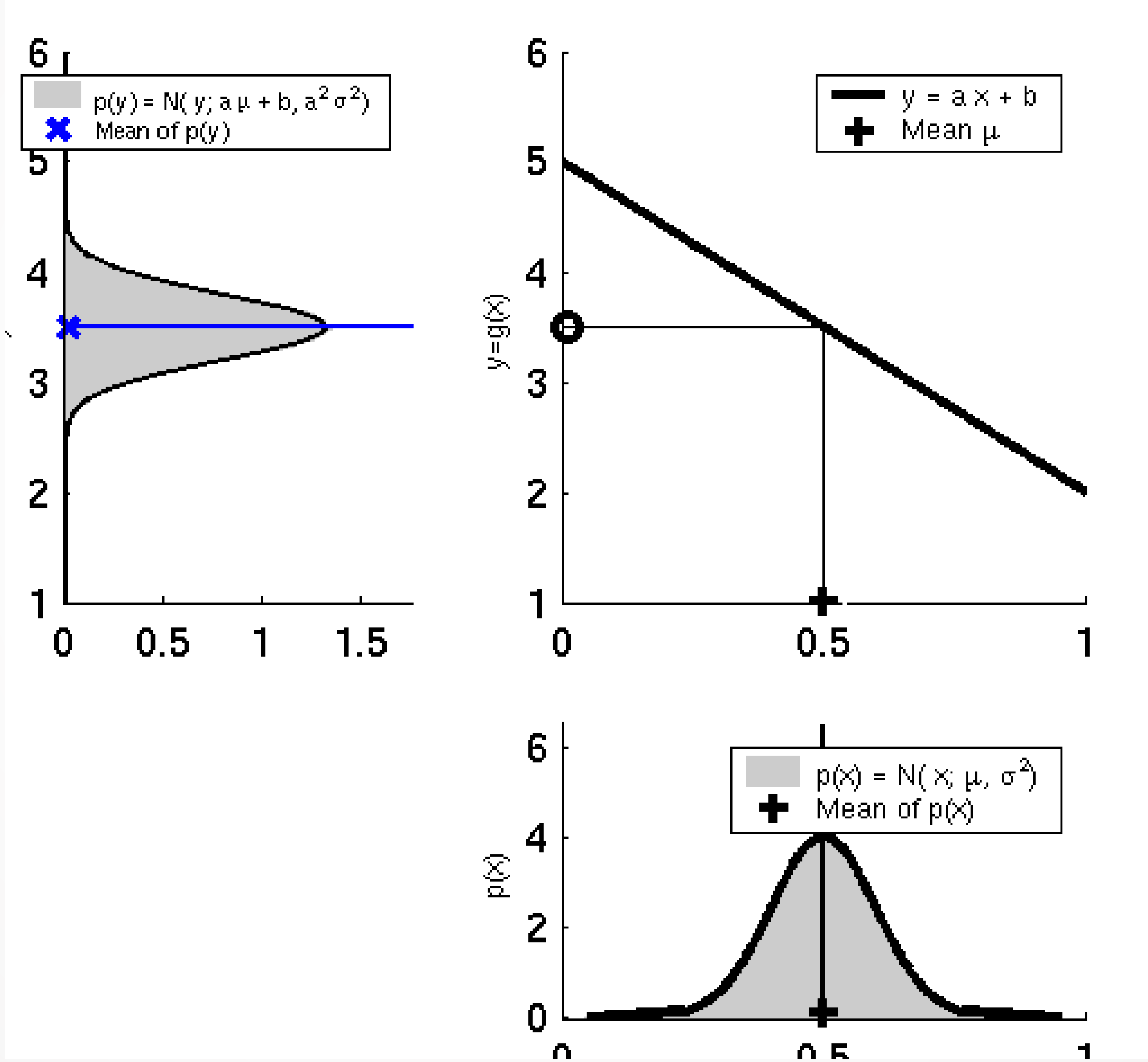
Kalman Filter Assumptions

- Gaussian distributions and noise
- Linear motion and observation model
  - **What if this is not the case?**

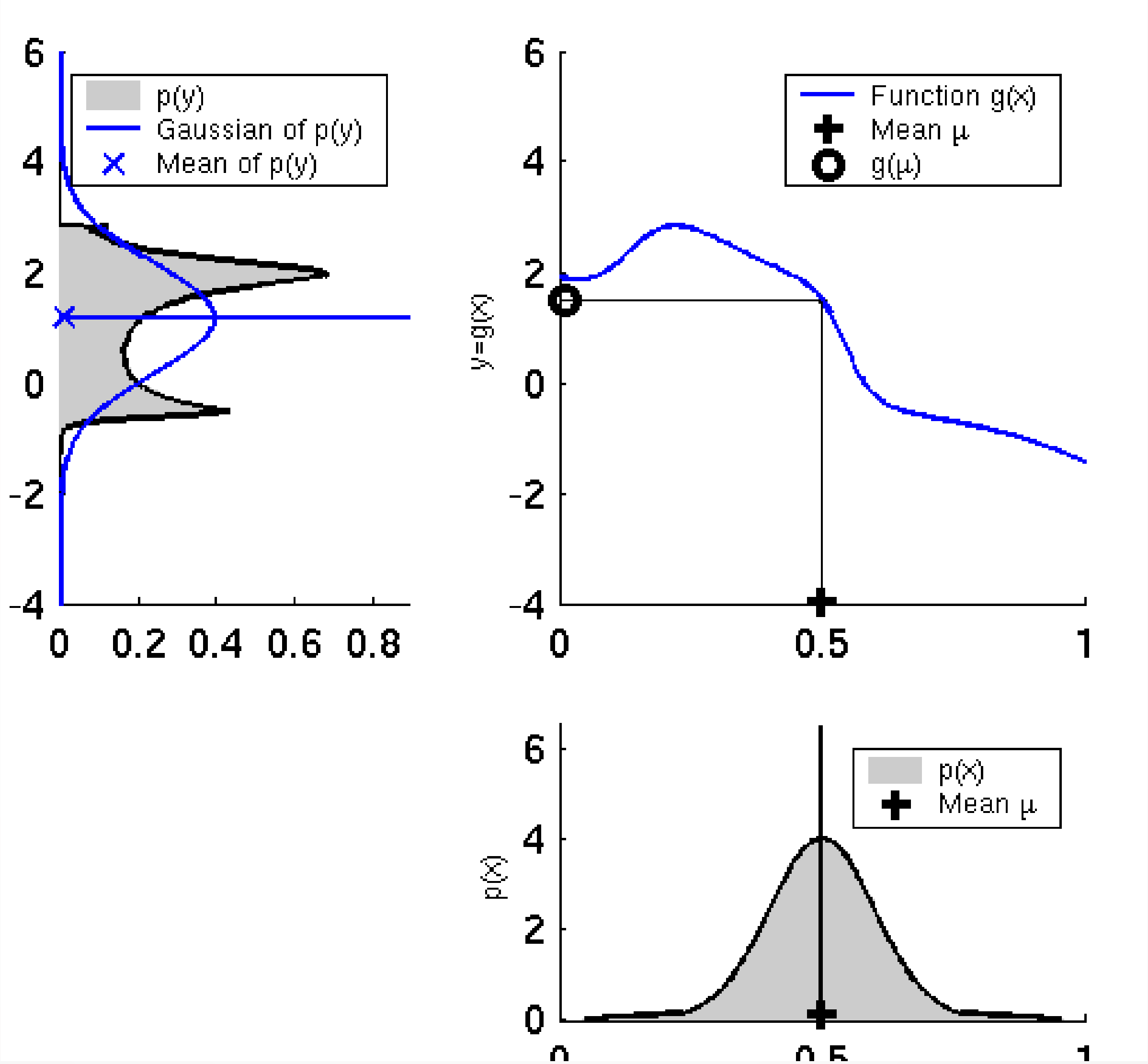
$$x_t = A_t x_{t-1} + B_t u_t + \epsilon_t$$

$$z_t = C_t x_t + \delta_t$$

Linearity Assumption Revisited



Nonlinear Function



Nonlinear Dynamical Systems

- Real-life robots are mostly nonlinear systems.
- The **motion equations** are expressed as **nonlinear differential (or difference) equations**:

$$x_t = g(u_t, x_{t-1})$$

- Also leading to a **nonlinear observation function:**

$$z_t = h(x_t)$$

Taylor Expansion

- Solution: approximate via linearization of both functions
- **Motion Function:**

$$\begin{aligned} g(x_{t-1}, u_t) &\approx g(\mu_{t-1}, u_t) + \frac{\partial g(\mu_{t-1}, u_t)}{\partial x_{t-1}}(x_{t-1} - \mu_{t-1}) \\ &= g(\mu_{t-1}, u_t) + G_t(x_{t-1} - \mu_{t-1}) \end{aligned}$$

- **Observation Function:**

$$\begin{aligned} h(x_t) &\approx h(\bar{\mu}_t) + \frac{\partial h(\bar{\mu}_t)}{\partial x_t}(x_t - \mu_t) \\ &= h(\bar{\mu}_t) + H_t(x_t - \mu_t) \end{aligned}$$

Reminder: Jacobian Matrix

- It is a non-square matrix mxn in general
- Given a vector-valued function:

$$g(x) = \begin{pmatrix} g_1(x) \\ g_2(x) \\ \vdots \\ g_m(x) \end{pmatrix}$$

- The **Jacobian matrix** is defined as:

$$G_x = \begin{pmatrix} \frac{\partial g_1}{\partial x_1} & \frac{\partial g_1}{\partial x_2} & \cdots & \frac{\partial g_1}{\partial x_n} \\ \frac{\partial g_2}{\partial x_1} & \frac{\partial g_2}{\partial x_2} & \cdots & \frac{\partial g_2}{\partial x_n} \\ \vdots & \vdots & \cdots & \vdots \\ \frac{\partial g_m}{\partial x_1} & \frac{\partial g_m}{\partial x_2} & \cdots & \frac{\partial g_m}{\partial x_n} \end{pmatrix}$$

Extended Kalman Filter

- For each time step, do:
- **Apply Motion Model:**

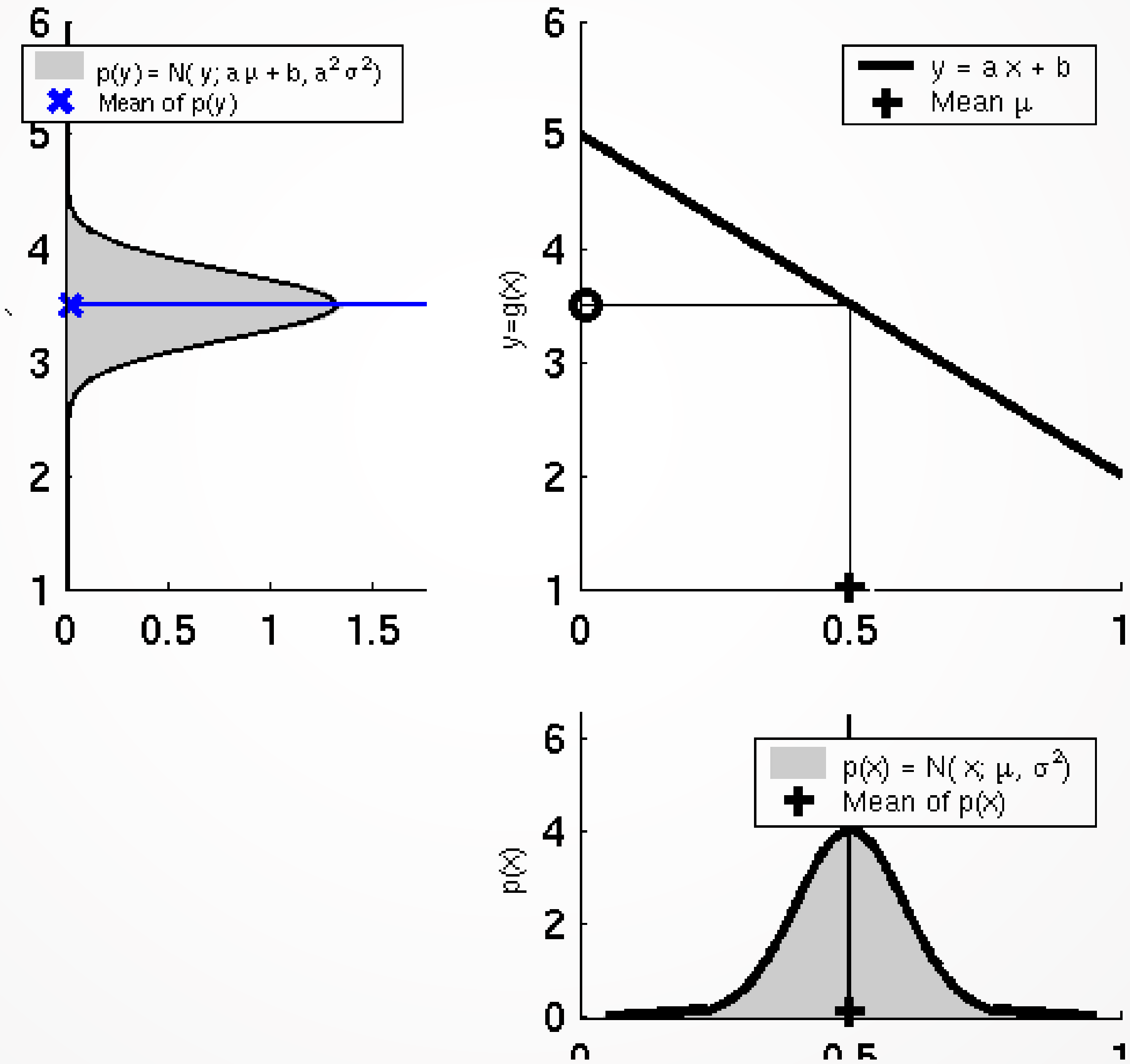
$$\bar{\mu}_t = g(\mu_{t-1}, u_t)$$
$$\bar{\Sigma}_t = G_t \Sigma G_t^\top + Q \quad \text{with} \quad G_t = \frac{\partial g(\mu_{t-1}, u_t)}{\partial x_{t-1}}$$

- **Apply Sensor Model:**

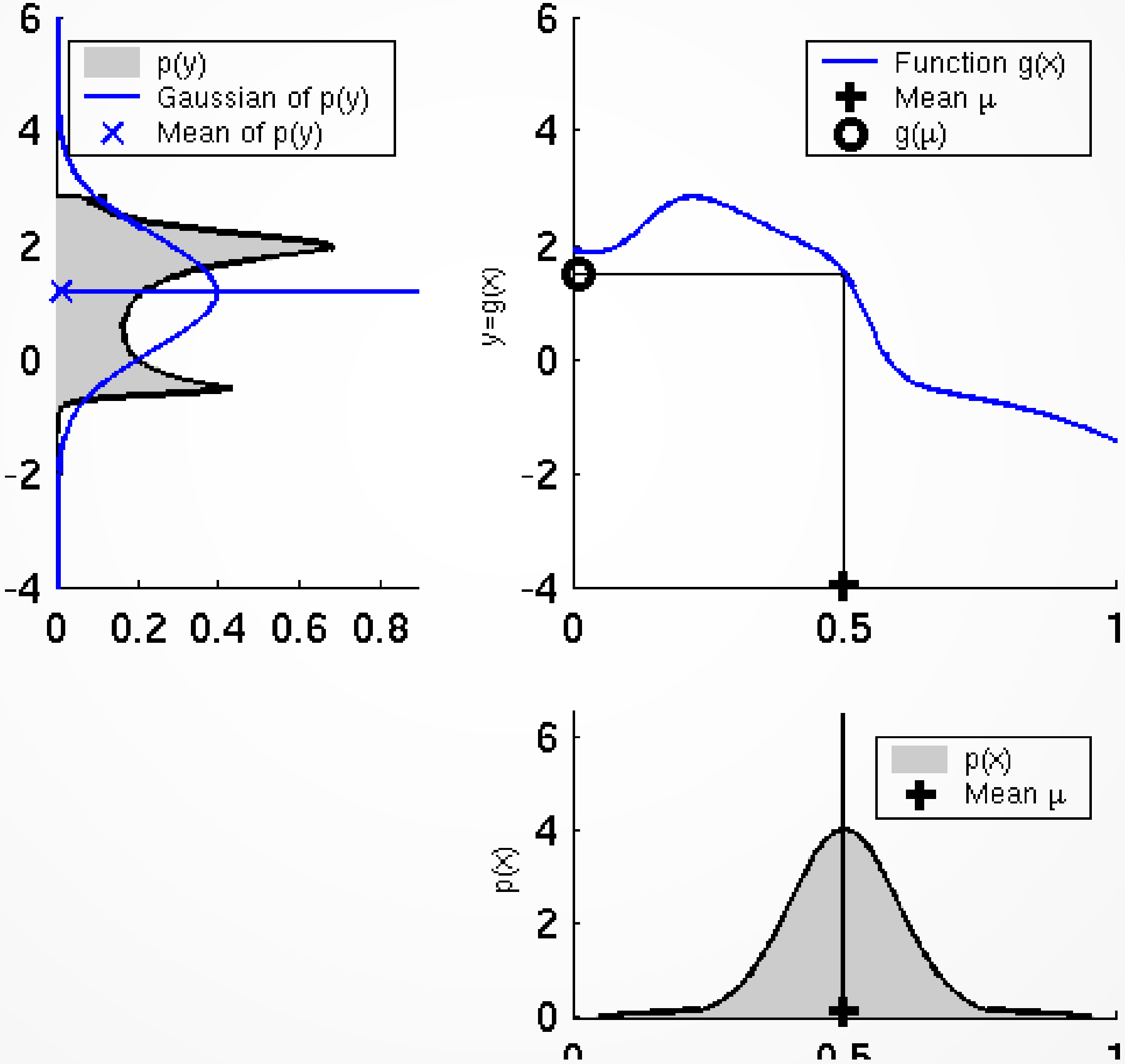
$$\mu_t = \bar{\mu}_t + K_t(z_t - h(\bar{\mu}_t))$$
$$\Sigma_t = (I - K_t H_t)\bar{\Sigma}_t$$

where $K_t = \bar{\Sigma}_t H_t^\top (H_t \bar{\Sigma}_t H_t^\top + R)^{-1}$ and $H_t = \dfrac{\partial h(\bar{\mu}_t)}{\partial x_t}$

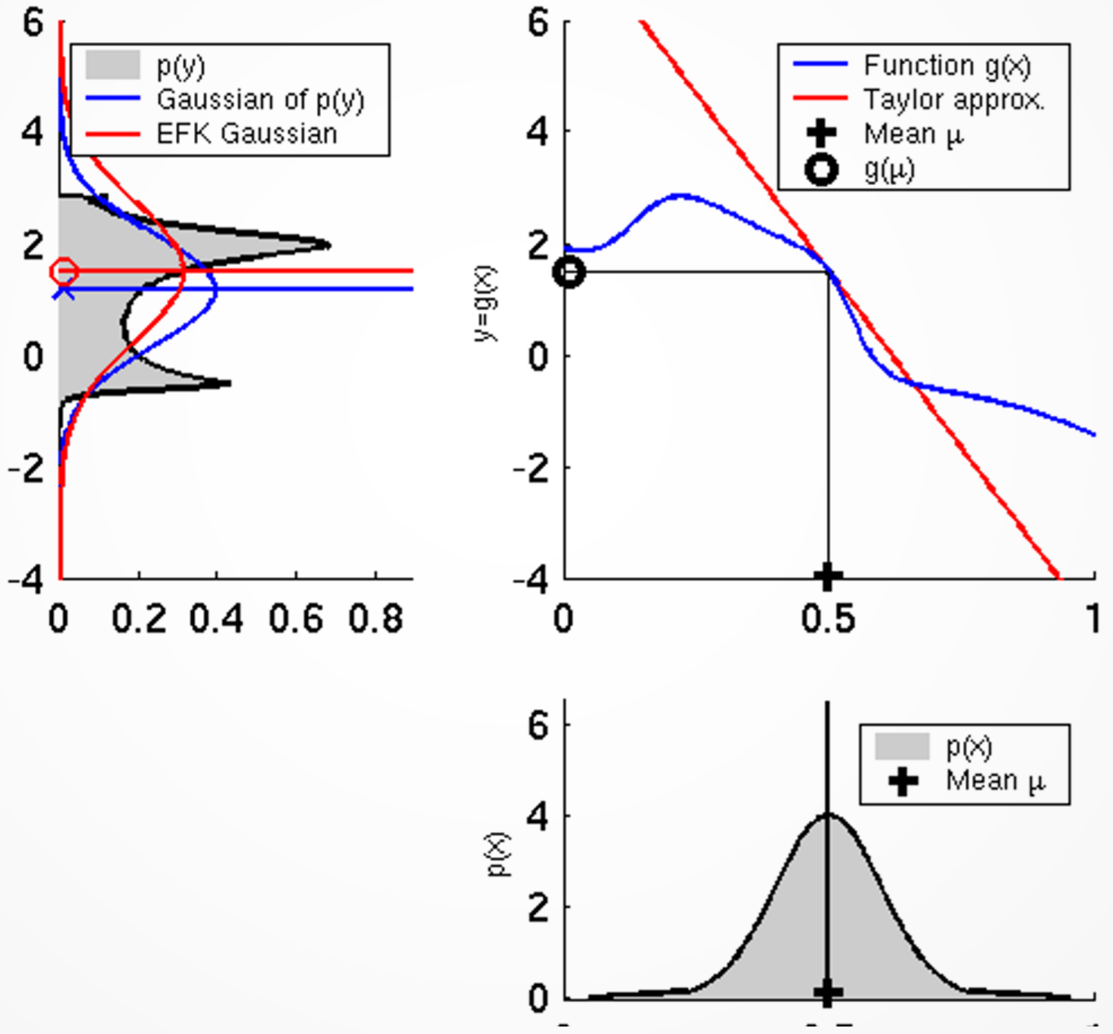
Linearity Assumption Revisited



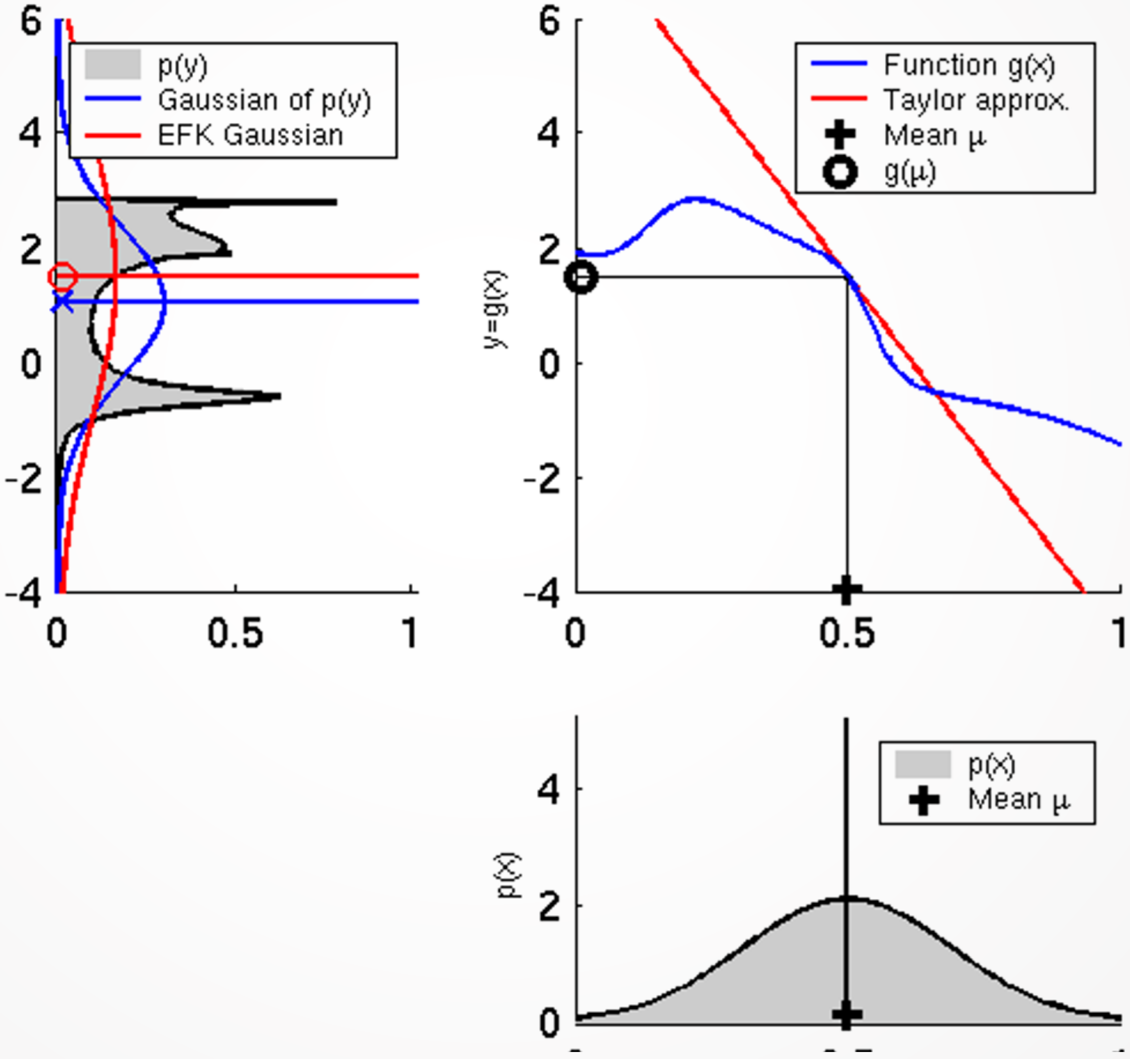
Nonlinear Function



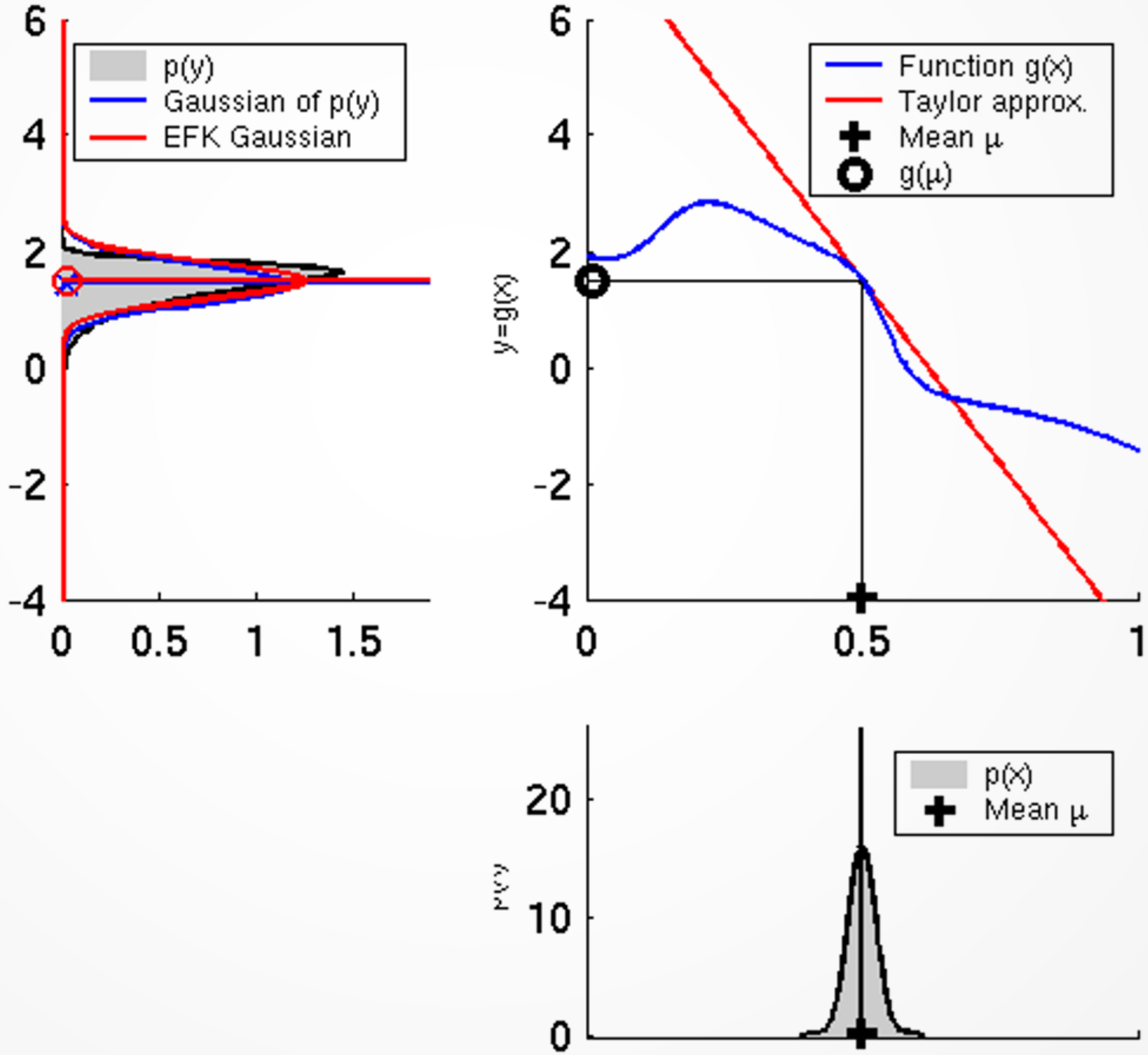
EKF Linearization (1)



EKF Linearization (2)



EKF Linearization (3)



Linearized Motion Model

- The linearized model leads to:

$$p(x_t \mid u_t, x_{t-1}) \approx \det\left(2\pi R_t\right)^{-\frac{1}{2}}$$
$$\exp\Big(-\frac{1}{2}\left(x_t - g(u_t, \mu_{t-1}) - G_t\left(x_{t-1} - \mu_{t-1}\right)\right)^T$$
$$R_t^{-1}\left(x_t - \underbrace{g(u_t, \mu_{t-1}) - G_t\left(x_{t-1} - \mu_{t-1}\right)}_{\text{linearized model}}\right)\Big)$$

- R_t describes the noise of the motion.

EKF Algorithm

1: $\textbf{Extended\_Kalman\_filter}(\mu_{t-1}, \Sigma_{t-1}, u_t, z_t)$**:**

2: $\bar{\mu}_t = \underline{g(u_t, \mu_{t-1})}$

3: $\bar{\Sigma}_t = G_t \, \Sigma_{t-1} \, G_t^T + R_t$

4: $K_t = \bar{\Sigma}_t \, H_t^T (H_t \, \bar{\Sigma}_t \, H_t^T + Q_t)^{-1}$

5: $\mu_t = \bar{\mu}_t + K_t(z_t - \underline{h(\bar{\mu}_t)})$

6: $\Sigma_t = (I - K_t \, H_t) \, \bar{\Sigma}_t$

7: $return \; \mu_t, \Sigma_t$

$A_t \leftrightarrow G_t$

$C_t \leftrightarrow H_t$

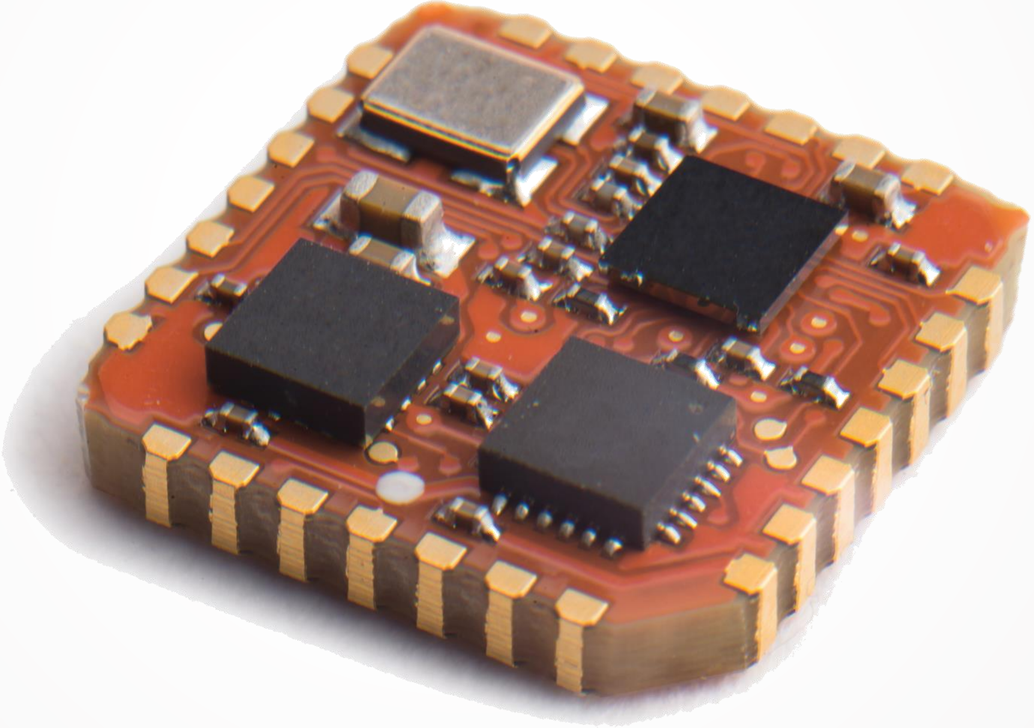
KF vs EKF

EKF Summary

- Extension of the Kalman Filter.
- One way to deal with nonlinearities.
- Performs local linearizations.
- Works well in practice for moderate nonlinearities.
- Large uncertainty leads to increased approximation error.

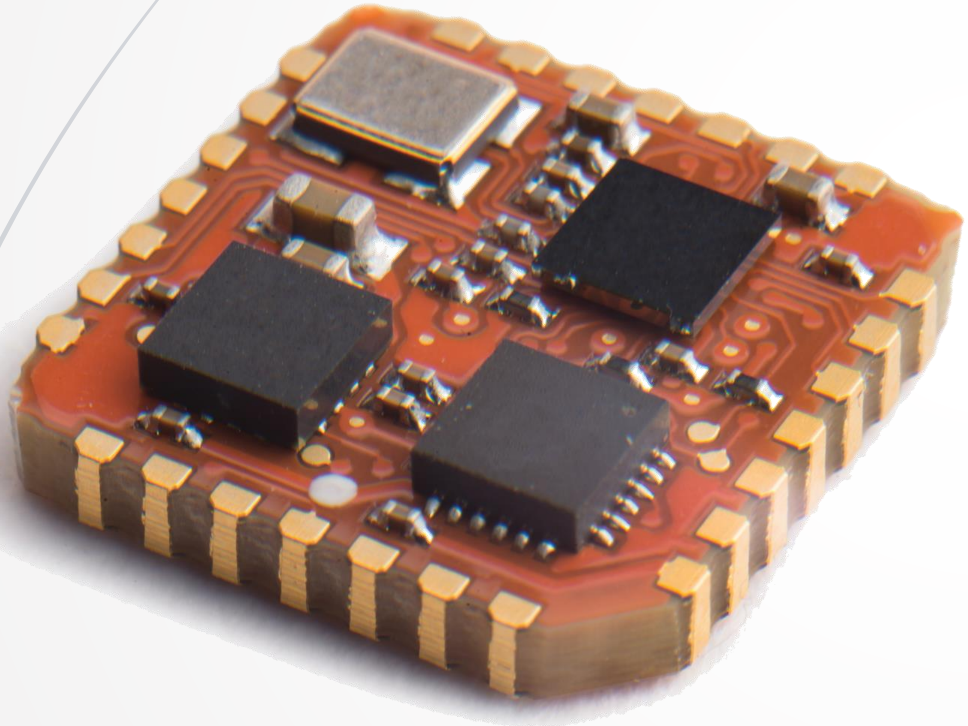
EKF Discussion

IMU



EKF Discussion

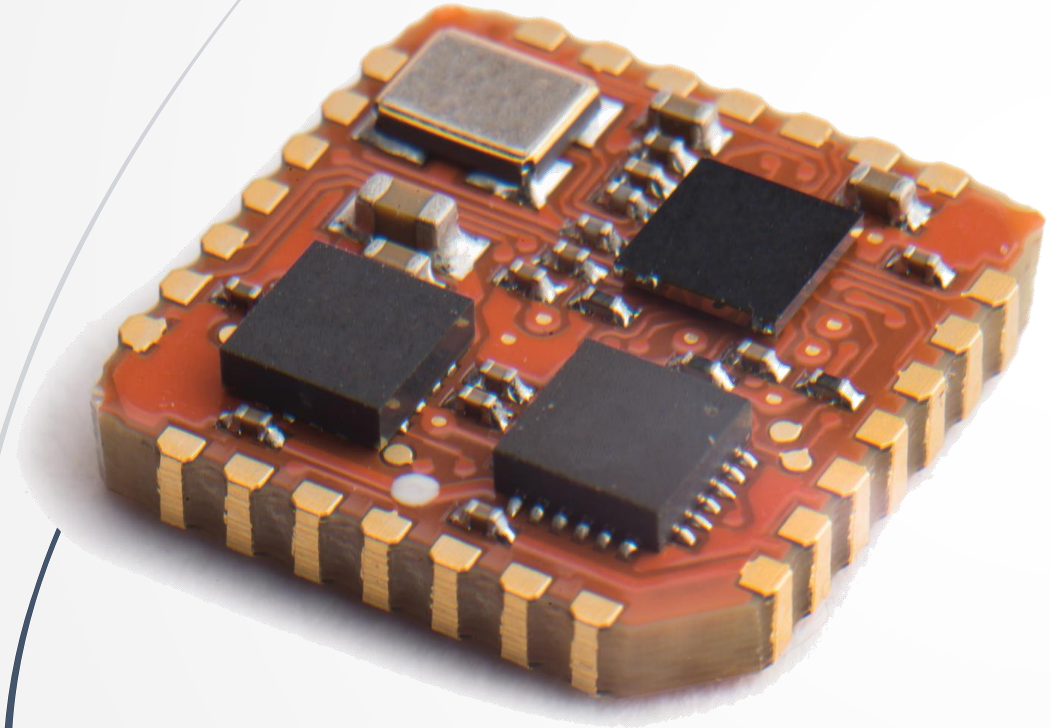
IMU + Compass



GPS

EKF Discussion

IMU + Compass



Camera

Thank you!

Please ask your question!