Lecture 7
ROB-GY 7863 / CSCI-GA 3033 7863: Planning, Learning, and Control for Space Robotics

Benjamin Riviere

October 20, 2025

Logistics

- Project 1 Feedback:
  - Good job everyone!
  - Best presentation: Christian and Nathaniel
  - Best report: Nam and Haizhou
  - What to focus on next time:
    - Videos (concept of operations)
    - Validation (how do we know what you are showing us is correct)
    - Clarity of presentation: What is the problem you are trying to solve, what is your method, what is your specific innovation? "Whereas existing work does XX, we do XX, which is important because XX".
- Project 2 Deadlines:
  - Final proposals: November 24th
  - Final presentations: December 8th

Space Culture I

PLANETARY ROBOTS

Autonomous robotics is driving Perseverance rover's progress on Mars

Vandi Verma*, Mark W. Maimone, Daniel M. Gaines, Raymond Francis, Tara A. Estlin, Stephen R. Kuhn, Gregg R. Rabideau, Steve A. Chien, Michael M. McHenry, Evan J. Graser, Arturo L. Rankin, Ellen R. Thiel

NASA's Perseverance rover uses robotic autonomy to achieve its mission goals on Mars. Its self-driving autonomous navigation system (AutoNav) has been used to evaluate 88% of the 17.7-kilometer distance traveled during its first Mars year of operation. Previously, the maximum total autonomous distance evaluated was 2.4 kilometers by the Opportunity rover during its 14-year lifetime. AutoNav has set multiple planetary rover records, including the greatest distance driven without human review (699.9 meters) and the greatest single-day drive distance (347.7 meters). The Autonomous Exploration for Gathering Increased Science (AEGIS) system analyzes wide-angle imagery onboard to autonomously select targets for observations by the SuperCam instrument, a multimode sensor suite capable of millimeter-scale geochemical and mineralogical analysis. AEGIS enables observations of scientifically interesting targets during or immediately after long drives without the need for ground communication. OnBoard Planner (OBP) is a scheduling capability planned for operational use in September 2023 that has the potential to reduce energy usage by up to 20% and complete drive and arm-contact science campaigns in 25% fewer days on Mars. This paper presents an overview of the AutoNav, AEGIS, and OBP capabilities used on Perseverance.

- Reference: Verma et al., 2023

Space Culture II

- Case for autonomy: "One of the many constraints of robotic Mars exploration is the limited frequency of commanding (new plans are sent only once every 1 to 3 days). This constraint makes autonomy a particularly attractive strategy. Other operational constraints make it challenging for autonomous capabilities to approach human performance. These constraints include limited available time for robotic operation (typically about 3 hours for driving per sol) and limited computational processing (a result of the low clock rate CPUs available for flight-proven, radiation-hardened processor boards)"
- AutoNav (Self-Driving Navigation): Drives at 97.97 meters per hour, autonomous navigation for 88% of all distance traveled, avoids challenging terrain to overcome (e.g. loose sand)

Space Culture III

- AEGIS (Automated Science Collection): selects rocks to science objectives
- Onboard Planner: schedules activities for the rover autonomously: "driving, drilling, and coring, can fail or take substantially longer or shorter than expected, affecting both activity duration and energy consumption. Because the OBP can replan using execution status and onboard measurements of temperature and battery state of charge, the planner enables more efficient operations than current open-loop ground-based commanding."

Space Culture IV

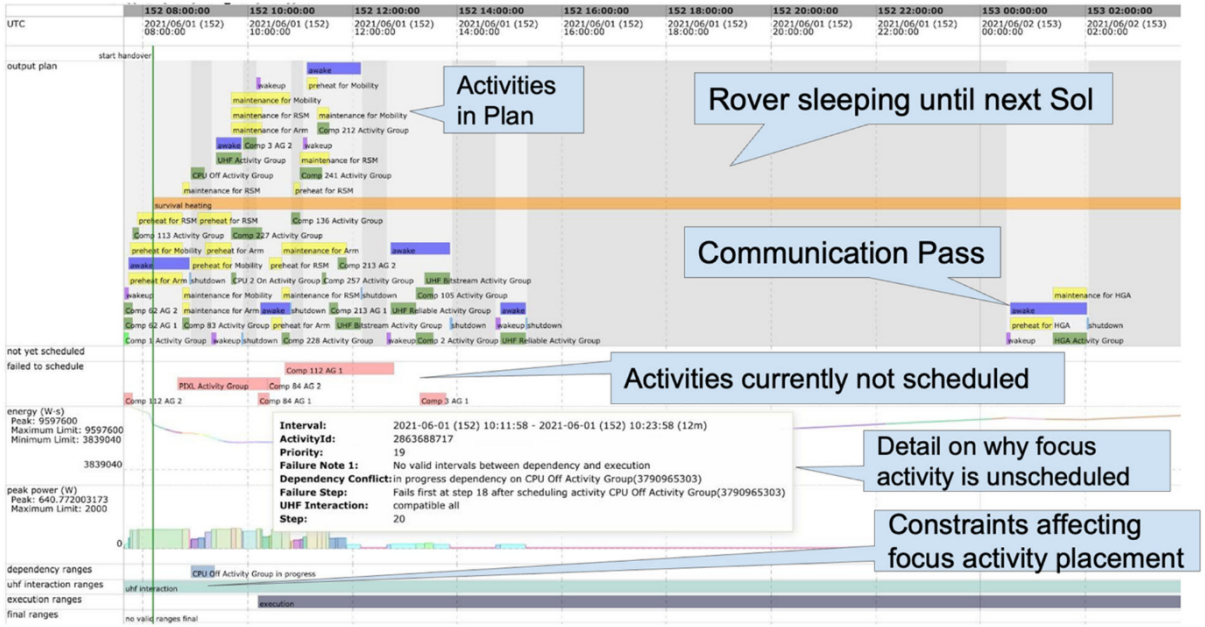
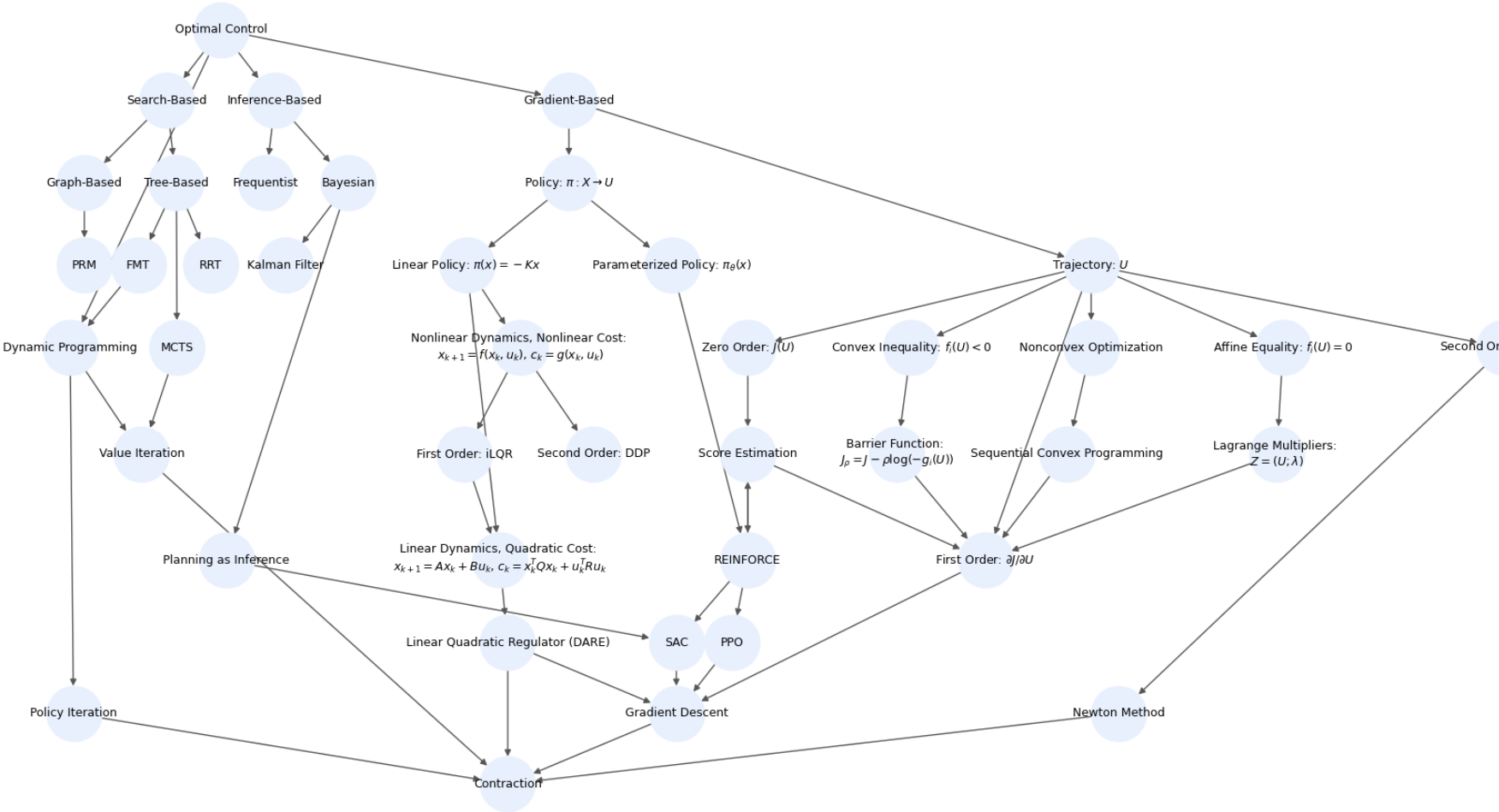


Figure S7: Sample rover plan in development in Crosscheck.

Agenda

- Connections
- Sampling-Based Optimization
  - Algorithm
  - Analysis: Sampling-based optimization $\approx$ gradient descent
  - Implementation:
  - Examples:
- Probability Review
  - Axioms of Probability
  - Bayes Rule
- Kalman Filter
- Planning as Inference

Connections



Purpose of Connections

- This organization is not perfect, and I encourage you to come up with your own or add your own connections, the figure code is on class github.
- "Organize your mind and you can hold more stuff up there!" - My high school physics teacher.
- Methods that rely on similar underlying mechanisms will have similar limitations: e.g. first and second order methods rely on convexity, and suffer from local minima.
- Making a connection is a very powerful contribution to research.
- The gold standard of algorithm analysis is **convergence analysis**: show your algorithm evolves towards the desired behavior/output. Contraction is a relatively easy and systematic way to do that.

Motivation: Sampling-Based Optimization I

- Previously, we learned about gradient-descent to solve optimization problems:

$$x^* = \min_x f(x)$$
$$x_{k+1} = x_k - \alpha \nabla_x f(x_k)$$

  but this requires knowledge of the gradient ∇f.
- Sampling-based optimization approximates the gradient ∇f using only calls to f. Our analysis today will prove this.
- Because sampling-based optimization approximates gradient descent, we can use our convergence analysis from lecture 5.
- Later, we will make connection between sampling-based optimization and policy gradient methods.
- Methods include MPPI **mppi**, CEM **cem**, PS **XX**.

Problem Setup

- Recall Optimal Control Problem: Given initial state, dynamics, and cost, select control inputs that minimize cost-over-time.
- Collocation form:

$$\min_{X,U} \sum_{k=1}^{H} c(x_k, u_k)$$
$$x_k = f(x_{k-1}, u_k), \quad \forall k \in [1, H]$$

- Shooting form:

$$J(U) = \sum_{k=1}^{H} c(\underbrace{f(\ldots f(f(x_0, u_1), u_2), \ldots, u_k)}_{x_k}, u_k)$$
$$\min_{U} J(U)$$

Algorithm

Algorithm Sampling-Based Optimization

- **Require:** Initial state x_0; horizon H; samples N; covariance Σ; dynamics $f(x, u)$; cost $c(\cdot)$; temperature λ; initial control trajectory $U_0 \in \mathbb{R}^{m \times H}$

for $t \in [0, ...,]$ **do**
  for $i \in [1, ..., N]$ **do**
    $\epsilon^i \sim N(0, \Sigma)$, $U^i = U_t + \epsilon^i \in \mathbb{R}^{m \times H}$, ▷ sample inputs.
    $\tau^i = \{(x_k^i, u_k^i, c_k^i)\}_{k=1}^H$ ▷ rollout.
    $J^i = \sum_k c_k^i$ ▷ compute value
  end for
  $\{w^i = \exp(-\frac{1}{\lambda}(J^i - \min_j J^j) \mid \forall i\}$ ▷ compute weights
  $U_{t+1} = U_t + \sum_i w^i \epsilon^i / \sum_j w^j$ ▷ update.
end for
Return U_t.

Preliminary to Analysis I

- References: Jordana et al., 2025; Kurtz et al., 2025; Pan et al., 2024
- Preliminary: Gaussian smoothing and log-sum-exp transform:

$$f_\Sigma(x) = \mathbb{E}_{z\sim\mathcal{N}(x,\Sigma)} f(z)$$

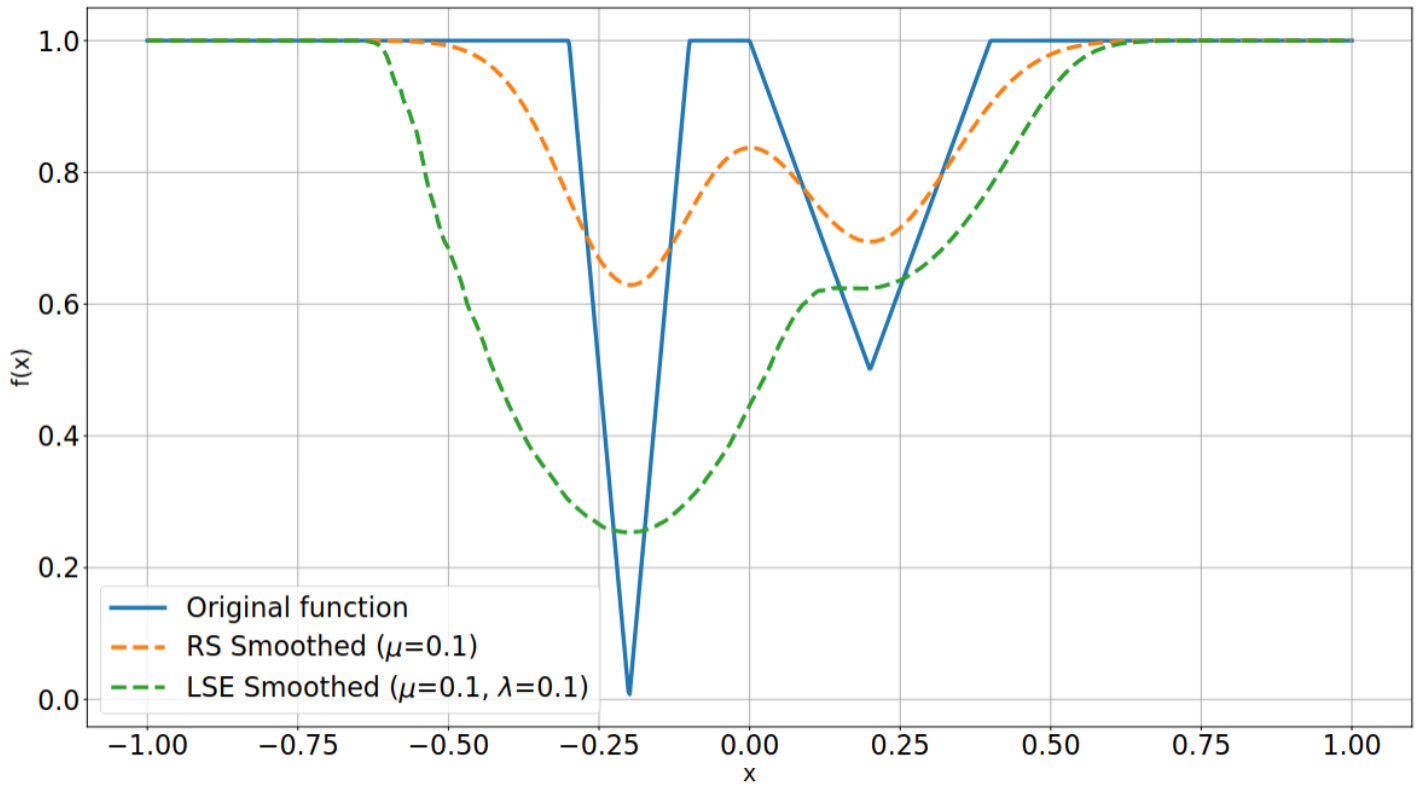
$$f_{\lambda,\Sigma}(x) = -\lambda \log\left(\mathbb{E}_{z\sim\mathcal{N}(x,\Sigma)} \exp(-\frac{1}{\lambda} f(z))\right)$$

- Limiting behavior:

$$\lim_{\Sigma\to 0} f_\Sigma(x) = f(x)$$

$$\lim_{\lambda\to\infty} f_{\lambda,\Sigma}(x) = f_\Sigma(x)$$

Preliminary to Analysis II



Analysis I

- Goal: show that sampling-based optimization is approximately gradient descent:

$$U_{t+1} = U_t - \alpha \nabla_U J_{\lambda,\Sigma}(U_t)$$

$$U_{t+1} = U_t + \frac{\sum_i \exp -\frac{1}{\lambda} J(U^i)\epsilon^i}{\sum_j \exp -\frac{1}{\lambda} J(U^j)}$$

- For notational simplicity, lets take the derivative of $f_{\lambda,\Sigma}(x)$ instead of $J_{\lambda,\Sigma}(U)$:

$$\nabla f_{\lambda,\Sigma}(x) = \nabla(-\lambda \log \left(\mathbb{E}_{z\sim\mathcal{N}(x,\Sigma)} \exp(-\frac{1}{\lambda} f(z)) \right)$$

$$= -\lambda \frac{\nabla \left(\mathbb{E}_{z\sim\mathcal{N}(x,\Sigma)} \exp(-\frac{1}{\lambda} f(z)) \right)}{\mathbb{E}_{z\sim\mathcal{N}(x,\Sigma)} \exp(-\frac{1}{\lambda} f(z))}$$

Analysis II

- Expand the numerator using **reparameterization trick**:

$$\nabla \left(\mathbb{E}_{z \sim \mathcal{N}(x,\Sigma)} \exp(-\frac{1}{\lambda} f(z))) \right) = \nabla \left(\mathbb{E}_{\epsilon \sim \mathcal{N}(0,\Sigma)} \exp(-\frac{1}{\lambda} f(x+\epsilon))) \right)$$
$$= -\frac{1}{\lambda} \mathbb{E}_{\epsilon \sim \mathcal{N}(0,\Sigma)} \exp(-\frac{1}{\lambda} f(x+\epsilon))) \nabla f(x+\epsilon)$$

  But this is bad because we do not have ∇f information. Let's try something else.

- Recall definition of expectation:

$$\mathbb{E}_{z \sim p} f(z) = \int p(z) f(z) dz$$

Analysis III

- Apply definition of expectation to the numerator:

$$\nabla \left(\mathbb{E}_{z \sim \mathcal{N}(x,\Sigma)} \exp(-\frac{1}{\lambda} f(z)) \right)$$
$$= \nabla_x \int_{-\infty}^{\infty} \underbrace{C \exp\left(\frac{1}{2}(x-z)^T \Sigma^{-1}(x-z)\right)}_{p(z)} \exp(-\frac{1}{\lambda} f(z))$$
$$= \int_{-\infty}^{\infty} C \exp\left(\frac{1}{2}(x-z)^T \Sigma^{-1}(x-z)\right) \Sigma^{-1}(x-z) \exp(-\frac{1}{\lambda} f(z))$$
$$= \mathbb{E}_{z \sim \mathcal{N}(x,\Sigma)} \exp(-\frac{1}{\lambda} f(z)) \Sigma^{-1}(x-z)$$

Analysis IV

- Compute numerator and denominator with finite samples:

$$\mathbb{E}_{z\sim\mathcal{N}(x,\Sigma)} \exp(-\frac{1}{\lambda}f(z))\Sigma^{-1}(x-z) = \frac{1}{N}\sum_{i=1}^{N} \exp(-\frac{1}{\lambda}f(x^i))\Sigma^{-1}(\epsilon^i)$$

$$\mathbb{E}_{z\sim\mathcal{N}(x,\Sigma)} \exp(-\frac{1}{\lambda}f(z)) = \frac{1}{N}\sum_{i=1}^{N} \exp(-\frac{1}{\lambda}f(x^i))$$

- Return to original gradient step and plug in our notation ($f \to J, x \to U$):

$$\begin{aligned}\nabla J_{\lambda,\Sigma}(U) &= -\lambda \frac{\frac{1}{N}\sum_{i=1}^{N} \exp(-\frac{1}{\lambda}J(U^i))\Sigma^{-1}(\epsilon^i)}{\frac{1}{N}\sum_{i=1}^{N} \exp(-\frac{1}{\lambda}J(U^i))} \\ &= -\lambda\Sigma^{-1}\frac{\sum_{i=1}^{N} \exp(-\frac{1}{\lambda}J(U^i))(\epsilon^i)}{\sum_{i=1}^{N} \exp(-\frac{1}{\lambda}J(U^i))}\end{aligned}$$

Analysis V

- Assuming $\Sigma = \sigma^2 I$, select $\alpha = \sigma^2/\lambda$, and recover the original goal of MPPI step is approximately gradient step:

$$-\alpha \nabla J_{\lambda,\Sigma}(U) = \frac{\sum_{i=1}^{N} \exp(-\frac{1}{\lambda} J(U^i))(\epsilon^i)}{\sum_{i=1}^{N} \exp(-\frac{1}{\lambda} J(U^i))}$$

- Why is this the correct choice for α?
- For an m-strongly convex and L-smooth function, we previously had $\alpha = \frac{1}{L}$ for standard optimization and $\alpha = \frac{2}{m+L}$ for contraction. So here $L = \lambda/\sigma^2$, i.e. Large smoothing and low temperature gives small L and vice versa.

Implementation

- Parallelization:
  - For robotics applications, GPUs generate thousands of sampled trajectories per clock cycle.
  - Algorithm performance will improve with GPU hardware.
- Language:
  - Compiled languages: Source code is translated entirely into machine-readable instructions by a compiler before execution begins, e.g. C++.
  - Interpretted: Source code is read line by line without a compiler, e.g. Python.
  - Just-in-time (JIT) compilation: Runs interpretted, but functions that are reused get compiled on-the-fly, e.g. Jax (Jax has other features like auto diff, hardware acceleration, functional programming)
- Here is an example of a repository that implements these ideas and should be easy to use with Mujoco: link. Show code.

Break

- Break

Motivation for Probability I

- Reference: Abbeel, 2019. Bigger picture Thrun, 2002
- Often state is unknown and only noisy sensors are available. Probability provides a framework to fuse information (e.g. Kalman filter)
- Dynamics are often easier to model as stochastic, so we cannot optimize for a single outcome, but we have to optimize to achieve a distribution of outcomes
- Example: Helicopter
  - State: position, orientation, velocity, angular rate
  - Sensors:
    - GPS: position / velocity
    - IMU: angular rates, accelerometer, magnetometer
  - Dynamics: noise from wind

Probability Review I

- Probability space: (Ω, F, P) where Ω is the sample space, F is the event space, and P is the probability measure
- Example (Flipping a coin):
  - Sample space: $\Omega = \{\text{Heads}, \text{Tails}\}$
  - Event space: $F = \{\emptyset, \{\text{Heads}\}, \{\text{Tails}\}, \Omega\}$
  - Probability measure: $P(\{\text{Heads}\}) = 0.5$, $P(\{\text{Tails}\}) = 0.5$, $P(\emptyset) = 0$, $P(\Omega) = 1$
- Axioms:
  - Non-negativity: $P(A) \geq 0, \quad \forall A \in F$
  - Unit measure: $P(\Omega) = 1$
  - Additivity: $P(A \cup B) = P(A) + P(B) - P(A \cap B)$
- De Morgan's Laws:

$$P((A \cup B)^c) = P(A^c \cap B^c) \tag{1}$$
$$P((A \cap B)^c) = P(A^c \cup B^c) \tag{2}$$

where $A^c = \Omega / A$ denotes set complement

Discrete/Continuous Random Variables

Discrete Random Variables

- X denotes a discrete random variable
- X can take on a countable number of values in $\{x_1, x_2, \ldots, x_n\}$
- $p(X = x_i)$ or $p(x_i)$ is the probability that X takes value x_i
- p is called the probability mass function
- Example: X models the outcome of flipping a coin: $P(x_1) = 0.5$, $P(x_2) = 0.5$

Continuous Random Variables

- X denotes a continuous random variable
- $p(X = x)$ or $p(x)$ is the probability density function
- Example: X models the outcome of sensor noise: $p(x) = ce^{-x^2/\sigma}$

Joint and Conditional Probability

- Joint probability: $p(X = x \text{ and } Y = y) = p(x, y)$
- X and Y are independent iff $p(x, y) = p(x)p(y)$
- $P(x|y)$ is the probability of x given y.

$$p(x, y) = p(x|y)p(y) = p(y|x)p(x) \tag{3}$$

- If X and Y are independent then $p(x|y) = p(x)$
- Law of total probability:

$$\sum_x p(x) = 1 \qquad \int p(x) = 1 \tag{4}$$

$$p(x) = \sum_y p(x, y) \qquad p(x) = \int_y p(x, y)dy \tag{5}$$

$$p(x) = \sum_y p(x|y)p(y) \qquad p(x) = \int_y p(x|y)p(y)dy \tag{6}$$

Bayes Rule

- Manipulation of product rule
  $p(x, y) = p(x|y)p(y) = p(y|x)p(x)$:

$$p(x|y) = \frac{p(y|x)p(x)}{p(y)} = \frac{\text{likelihood} \cdot \text{prior}}{\text{evidence}} \tag{7}$$

  where y is the data, x is what we are trying to estimate $p(x)$ is the prior, $p(y|x)$ is the likelihood function and $p(y)$ is the evidence.

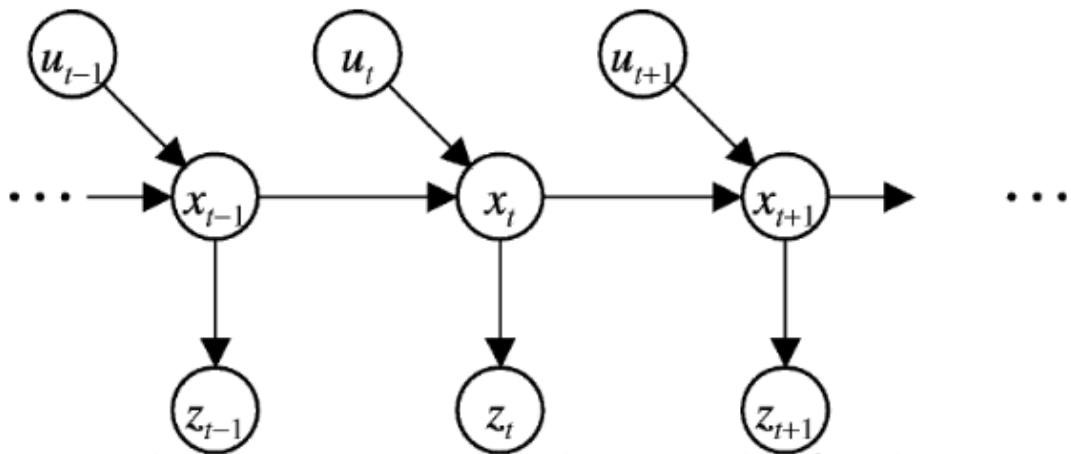
Conditional Bayes Rule

- Lets say I have extra information z that changes the prior, likelihood function, and evidence. How does that change Bayes rule?

$$P(x|y,z) = \textbf{TODO:}\textit{exercise} \tag{8}$$

Recursive Bayes Filter I

- Problem statement: Given sensor and actuator data: $d_t = \{u_1, y_1, u_2, y_2, \ldots, u_t, y_t\}$, sensor model: $p(y|x)$, process model: $p(x'|x, u)$, prior: $p(x)$, compute estimate of state x_t. The posterior of the state is called the belief: $b_t(x) = p(x_t|d_t)$
- Markov assumptions:



$$p(z_t \mid x_{0:t}, z_{1:t-1}, u_{1:t}) = p(z_t \mid x_t)$$

$$p(x_t \mid x_{1:t-1}, z_{1:t-1}, u_{1:t}) = p(x_t \mid x_{t-1}, u_t)$$

Recursive Bayes Filter II

$$b_t(x) = p(x_t|u_1, y_1, \ldots, u_t, y_t)$$

By Belief Definition

$$= \eta p(y_t|x_t, u_1, y_1, \ldots, u_t, y_t) p(x_t|u_1, y_1, \ldots, u_t, y_t)$$

By Bayes

$$= \eta p(y_t|x_t) p(x_t|u_1, y_1, \ldots, u_t, y_t)$$

By Markov

$$= \eta p(y_t|x_t) \int_{x_{t-1}} p(x_t|x_{t-1}, u_1, y_1, \ldots, u_t, y_t) p(x_{t-1}|u_1, y_1, \ldots, u_t, y_t) dx_{t-}$$

By Total Probability

$$= \eta p(y_t|x_t) \int_{x_{t-1}} p(x_t|x_{t-1}, u_t) p(x_{t-1}|u_1, y_1, \ldots, u_t, y_t) dx_{t-1}$$

By Markov

$$= \eta p(y_t|x_t) \int_{x_{t-1}} p(x_t|x_{t-1}, u_t) b_{t-1}(x) dx_{t-1}$$

By Belief definition

Kalman Filter I

- Special case of Bayes Filter when:

$$x_{t+1} = Ax_t + Bu_t + w \tag{9}$$
$$y_t = Cx_t + v \tag{10}$$

where $w \sim \mathcal{N}(0, \Sigma_x)$ and $v \sim \mathcal{N}(0, \Sigma_y)$. Using our new notation:

$$p(x'|x, u) = \mathcal{N}(Ax + Bu, \Sigma_x) \tag{11}$$
$$p(y|x) = \mathcal{N}(Cx, \Sigma_y) \tag{12}$$
$$p(x) = \mathcal{N}(\mu_0, P_0) \tag{13}$$

- Strategy is simple: plug in these equations into previous bayes filter equations, assuming prior $b_{t-1}(x) = \mathcal{N}(\mu_{t-1}, P_{t-1})$

Kalman Filter II

- First, lets consider $\overline{b}_t(x_t) = \int_{x_{t-1}} p(x_t|x_{t-1}, u_t) b_{t-1}(x_{t-1}) dx_{t-1}$ (in the literature this is called propagation):

$$\overline{b}_t(x_t) = \mathcal{N}(\overline{\mu}_t, \overline{P}_t) \tag{14}$$

$$\overline{\mu}_t = A\mu_{t-1} + Bu \tag{15}$$

$$\overline{P}_t = AP_{t-1}A^T + \Sigma_x \tag{16}$$

Derivation is kind of tedious but here are key steps: write both terms in the exponent, group the quadratic and linear terms in x_{t-1}, complete the square, perform the Gaussian integral, and then use Woodbury to rewrite the resulting term as a gaussian.

Kalman Filter III

- Next, lets consider $b_t(x_t) = p(z_t|x_t)\overline{b}_t(x_t)$ (in the literature this is called innovation):

$$b_t(x_t) = \mathcal{N}(\overline{\mu}_t + K_t(y_t - C\overline{\mu}_t), (I - K_t C)\overline{P}_t) \tag{17}$$

$$K_t = \overline{P}_t C^T (C\overline{P}_t C^T + \Sigma_y)^{-1} \tag{18}$$

Similar key steps: write exponential, multiply gaussians, group terms, complete the square.

Intuition Picture

- **TODO:** draw on board limiting behavior: $P0 = 0$, $Q = 0$, $V = 0$

Kalman Filter Contraction Analysis I

- Linear Gaussian system:

$$x_{k+1} = Ax_k + Bu_k + w_k, \tag{19}$$
$$y_k = Cx_k + v_k, \tag{20}$$

with $w_k \sim \mathcal{N}(0, \Sigma_x)$, $v_k \sim \mathcal{N}(0, \Sigma_y)$, $\Sigma_x, \Sigma_y \succ 0$.

- Kalman filter (predict/update):

$$\bar{x}_{k+1} = Ax_k + Bu_k \tag{21}$$
$$\bar{P}_{k+1} = AP_kA^T + \Sigma_x \tag{22}$$
$$S_{k+1} = C\bar{P}_{k+1}C^T + \Sigma_y \tag{23}$$
$$K_{k+1} = \bar{P}_{k+1}C^T S_{k+1}^{-1}, \tag{24}$$
$$x_{k+1} = \bar{x}_{k+1} + K_{k+1}(y_{k+1} - C\bar{x}_{k+1}), \tag{25}$$
$$P_{k+1} = (\bar{P}_{k+1}^{-1} + C^T\Sigma_y^{-1}C)^{-1}. \tag{26}$$

Kalman Filter Contraction Analysis II

- Assumption (bounded covariance under standard detectability/stabilizability):

$$\underline{p}\, I \preceq P_k \preceq \overline{p}\, I, \quad \forall k. \tag{27}$$

- Virtual system (two KF observers, same u, y):

$$z_{k+1} = Az_k + Bu_k + K_{k+1}\big(y_{k+1} - C(Az_k + Bu_k)\big). \tag{28}$$

Identify two solutions of this discrete system: $z = \hat{x}_k$ and $z = x_k$, so contraction gives us desired behavior. Let $\delta_k = \lim_{\epsilon \to 0} \frac{\partial z_k^\epsilon}{\partial \epsilon}$. Then

$$\bar{\delta}_{k+1} = A\delta_k \tag{29}$$

$$\delta_{k+1} = (I - K_{k+1}C)\ \bar{\delta}_{k+1}. \tag{30}$$

Kalman Filter Contraction Analysis III

- Select **precision matrix** metric: $M_k = P_k^{-1}$ and define differential discrete Lyapunov function: $V_k = \delta_k^T M_k \delta_k$.
- Key identity for the measurement update (with $\bar{M}_{k+1} = \bar{P}_{k+1}^{-1}$):

$$(I - K_{k+1}C)^T M_{k+1}(I - K_{k+1}C) = \bar{M}_{k+1} - C^T S_{k+1}^{-1} C. \quad (31)$$

- Propagate then update the energy:

$$V_{k+1} = \delta_{k+1}^T M_{k+1} \delta_{k+1} = \bar{\delta}_{k+1}^T (\bar{M}_{k+1} - C^T S_{k+1}^{-1} C)\bar{\delta}_{k+1} \quad (32)$$

$$= \delta_k^T \left[A^T \bar{M}_{k+1} A - A^T C^T S_{k+1}^{-1} C A \right] \delta_k. \quad (33)$$

Kalman Filter Contraction Analysis IV

- Bounding each term in the M_k-metric. By boundedness of P_k (hence of $M_k = P_k^{-1}$) and $\Sigma_x \succ 0$,

$$\delta_k^T (A^T (A P_k A^T + \Sigma_x)^{-1} A) \delta_k \leq \delta_k^T (A^T (A P_k A^T)^{-1} A) \delta_k \tag{34}$$

$$= \delta_k^T P_k^{-1} \delta_k = \delta_k^T M_k \delta_k \tag{35}$$

  By uniform observability/detectability (and bounded P_k),

$$\exists \sigma > 0 : \quad A^T C^T S_{k+1}^{-1} C A \succeq \sigma M_k \tag{36}$$

Kalman Filter Contraction Analysis V

- Contraction inequality:

$$V_{k+1} \leq \delta_k^T (M_k - \sigma M_k)\, \delta_k = (1-\sigma)\, V_k. \tag{37}$$

If $\rho \equiv 1 - \sigma \in (0,1)$, then

$$V_{k+1} \leq \rho\, V_k \quad \Rightarrow \quad V_k \leq \rho^{\,k-k_0}\, V_{k_0}. \tag{38}$$

Thus, in the precision metric $M_k = P_k^{-1}$, the discrete-time KF is an **exponentially contracting observer**:

$$\|\delta_k\|_{M_k} \to 0 \quad \text{exponentially at rate } \rho. \tag{39}$$

Important Variations I

- Extended Kalman filter (local linear approximation)
- Unscented Kalman filter (pick sigma points, push through nonlinear functions, compute weighted average update using spectrum of prior)
- Ensemble Kalman filter (sample points, push through nonlinear functions, fit normal)
- Particle filter (predict, update, resample)

Coding Exercise I

- See lab github

References I

- Abbeel, Pieter (2019). *Probability Review, Bayes Filters, Gaussians (Lecture 11)*. Lecture slides, EECS 287, University of California, Berkeley. https://people.eecs.berkeley.edu/~pabbeel/cs287-fa19/slides/Lec11-probability-bayes-filters-gaussians.pdf (accessed Oct 19 2025) (cit. on p. 22).
- Jordana, Armand, Jianghan Zhang, Joseph Amigo, and Ludovic Righetti (2025). *An Introduction to Zero-Order Optimization Techniques for Robotics*. arXiv: 2506.22087 [cs.RO]. URL: https://arxiv.org/abs/2506.22087 (cit. on p. 13).
- Kurtz, Vince and Joel W. Burdick (2025). *Generative Predictive Control: Flow Matching Policies for Dynamic and Difficult-to-Demonstrate Tasks*. arXiv: 2502.13406 [cs.RO]. URL: https://arxiv.org/abs/2502.13406 (cit. on p. 13).

References II

- Pan, Chaoyi, Zeji Yi, Guanya Shi, and Guannan Qu (2024). *Model-Based Diffusion for Trajectory Optimization*. arXiv: 2407.01573 [cs.RO]. URL: https://arxiv.org/abs/2407.01573 (cit. on p. 13).
- Thrun, Sebastian (2002). "Probabilistic robotics". In: *Communications of the ACM* 45.3, pp. 52–57 (cit. on p. 22).
- Verma, Vandi, Mark W. Maimone, Daniel M. Gaines, Raymond Francis, Tara A. Estlin, Stephen R. Kuhn, Gregg R. Rabideau, Steve A. Chien, Michael M. McHenry, Evan J. Graser, Arturo L. Rankin, and Ellen R. Thiel (2023). "Autonomous robotics is driving Perseverance rover's progress on Mars". In: *Science Robotics* 8.80, eadi3099. DOI: 10.1126/scirobotics.adi3099. eprint: https://www.science.org/doi/pdf/10.1126/scirobotics.adi3099. URL: https://www.science.org/doi/abs/10.1126/scirobotics.adi3099 (cit. on p. 3).