Autonomous Robotics

Winter 2025

Abhishek Gupta

TAs: Carolina Higuera, Entong Su, Bernie Zhu

Class Outline

State Estimation

- Robotic System Design
- Filtering
- Localization
- SLAM

Control

- Feedback Control
- PID Control
- MPC
- LQR

Planning

- Search
- Heuristic Search
- Motion Planning
- Lazy Search

Learning

- Imitation Learning
- Policy Gradient
- Actor-Critic
- Model-Based RL

Logistics

- Car pick up today 1/15: 4:00-5:00pm
- Project 1 due on Jan 21 EOD

- Post questions, discuss any issues you are having on Ed.
- Students with **no** access to 002, e-mail us with your student ID.
- Students that have not been added to the class, email abhgupta@cs.washington.edu with the subject-line "Waitlisted for CSE478"

Recap

Why do Bayesian filters need to be probabilistic?

✓ 0

You don’t know the measurement model	0%
You don’t know the motion model	0%
The motion/measurement model may be incorrect	0%
We can maximize entropy	0%
None of the above	0%

Fundamental Problem: State is hidden

But all decision making depends on knowing state

Solution: Estimate belief over state

$$bel(x_t) = P(x_t | z_{1:t}, u_{1:t})$$

Belief is a probability of each possible state given history

Also called Posterior / Information state / State of knowledge

Represent belief? Parametric (Gaussian), Non-parametric (Histogram)

Bayes Filters

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

We want to recursively express Bel(x_t) in terms of three entities

$$p(z_t | x_t)$$

Measurement

$$p(x_t | x_{t-1}, u_{t-1})$$

Dynamics

$$Bel(x_{t-1})$$

Previous Belief

Bayes filter in a nutshell

Key Idea: Apply Markov to get a recursive update!

Step 0. Start with the belief at time step t-1

$$bel(x_{t-1})$$

Step 1: Prediction - push belief through dynamics given action

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

Step 2: Correction - apply Bayes rule given measurement

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

Lecture Outline

Recap

↓

Bayesian Filtering Examples

↓

Motion Models

↓

Observation Models

Example: Opening a Door

$\mathcal{X} =$ **O**PEN, **C**LOSED

$\mathcal{A} =$ **P**ULL, **L**EAVE $P(x_t|x_{t-1}, u_t)$

$$P(O|C,P) = 0.7$$
$$P(C|C,P) = 0.3$$

HTTPS://YOUTU.BE/AFUA50H9UEK

Example: Opening a Door

$$\begin{bmatrix} P(x_t = \mathbf{O} | x_{t-1} = \mathbf{O}, u_t) & P(x_t = \mathbf{O} | x_{t-1} = \mathbf{C}, u_t) \\ P(x_t = \mathbf{C} | x_{t-1} = \mathbf{O}, u_t) & P(x_t = \mathbf{C} | x_{t-1} = \mathbf{C}, u_t) \end{bmatrix}$$

$$P(.|., \mathbf{P}) = \begin{bmatrix} 0.8 & 0.7 \\ 0.2 & 0.3 \end{bmatrix} \qquad P(.|., \mathbf{L}) = \begin{bmatrix} 0.5 & 0 \\ 0.5 & 1 \end{bmatrix}$$

HTTPS://YOUTU.BE/AFUA50H9UEK

Example: Opening a Door

$\mathcal{X} =$ **O**PEN, **C**LOSED

$\mathcal{A} =$ **P**ULL, **L**EAVE

$\mathcal{Z} =$ **O**PEN, **C**LOSED

$$P(z_t | x_t)$$

$$\begin{bmatrix} P(z_t|\mathbf{O}) \\ P(z_t|\mathbf{C}) \end{bmatrix}$$

$$P(\mathbf{O}|.) = \begin{bmatrix} 0.6 \\ 0.2 \end{bmatrix} \quad P(\mathbf{C}|.) = \begin{bmatrix} 0.4 \\ 0.8 \end{bmatrix}$$

HTTPS://YOUTU.BE/AFUA50H9UEK

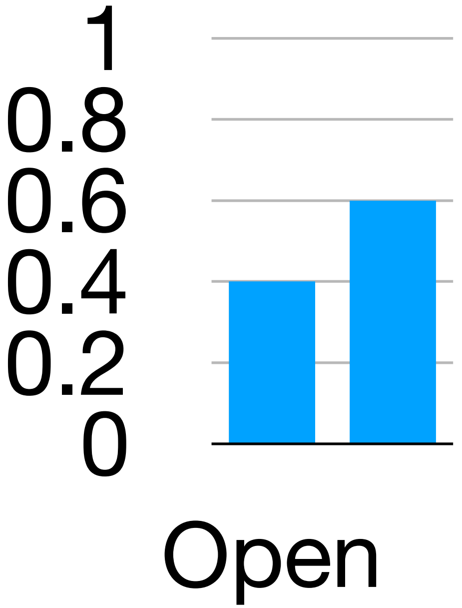
Example: Opening a Door

$\mathcal{X} =$ **O**PEN, **C**LOSED

$\mathcal{A} =$ **P**ULL, **L**EAVE

$\mathcal{Z} =$ **O**PEN, **C**LOSED

$$Bel(x_0) = \begin{bmatrix} 0.4 \\ 0.6 \end{bmatrix}$$



PULL

Example: Opening a Door

$\mathcal{X} =$ **O**PEN, **C**LOSED

$\mathcal{A} =$ **P**ULL, **L**EAVE

$\mathcal{Z} =$ **O**PEN, **C**LOSED

Prediction: Given action, propagate belief through dynamics

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

$$\begin{bmatrix} P(x_t = \mathbf{O}) \\ P(x_t = \mathbf{C}) \end{bmatrix} = \begin{bmatrix} P(x_t = \mathbf{O} | x_{t-1} = \mathbf{O}, u_t) & P(x_t = \mathbf{O} | x_{t-1} = \mathbf{C}, u_t) \\ P(x_t = \mathbf{C} | x_{t-1} = \mathbf{O}, u_t) & P(x_t = \mathbf{C} | x_{t-1} = \mathbf{C}, u_t) \end{bmatrix} \begin{bmatrix} P(x_{t-1} = \mathbf{O}) \\ P(x_{t-1} = \mathbf{C}) \end{bmatrix}$$

$\overline{Bel}(x_t)$ $\qquad$ $Bel(x_{t-1})$

Example: Opening a Door

$\mathcal{X} =$ **O**PEN, **C**LOSED

$\mathcal{A} =$ **P**ULL, **L**EAVE

$\mathcal{Z} =$ **O**PEN, **C**LOSED

Prediction: Given action, propagate belief through dynamics

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

$$\underbrace{\begin{bmatrix} 0.74 \\ 0.26 \end{bmatrix}}_{\overline{Bel}(x_t)} = \underbrace{\begin{bmatrix} 0.8 & 0.7 \\ 0.2 & 0.3 \end{bmatrix}}_{P(.|., \mathbf{P})} \underbrace{\begin{bmatrix} 0.4 \\ 0.6 \end{bmatrix}}_{Bel(x_{t-1})}$$

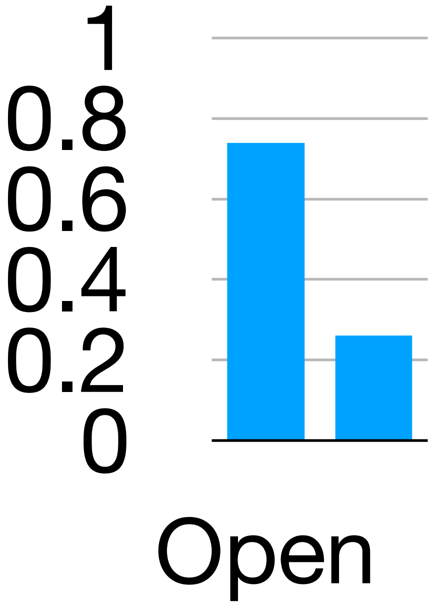
Example: Opening a Door

$\mathcal{X} =$ **O**PEN, **C**LOSED

$\mathcal{A} =$ **P**ULL, **L**EAVE

$\mathcal{Z} =$ **O**PEN, **C**LOSED

$$\overline{Bel}(x_t) = \begin{bmatrix} 0.74 \\ 0.26 \end{bmatrix}$$



CLOSED

Example: Opening a Door

$\mathcal{X} =$ **O**PEN, **C**LOSED

$\mathcal{A} =$ **P**ULL, **L**EAVE

$\mathcal{Z} =$ **O**PEN, **C**LOSED

Correction: Given measurement, apply Bayes' rule

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

$$\begin{bmatrix} P(x_t = \mathbf{O}) \\ P(x_t = \mathbf{C}) \end{bmatrix} = \eta \begin{bmatrix} P(z_t|\mathbf{O}) \\ P(z_t|\mathbf{C}) \end{bmatrix} * \begin{bmatrix} P(x_t = \mathbf{O}) \\ P(x_t = \mathbf{C}) \end{bmatrix}$$

$Bel(x_t)$ $\quad$ $P(\mathbf{C}|.)$ $\quad$ $\overline{Bel}(x_t)$

Example: Opening a Door

$\mathcal{X} =$ **O**PEN, **C**LOSED

$\mathcal{A} =$ **P**ULL, **L**EAVE

$\mathcal{Z} =$ **O**PEN, **C**LOSED

Correction: Given measurement, apply Bayes' rule

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

$$\underbrace{\begin{bmatrix} P(x_t = \mathbf{O}) \\ P(x_t = \mathbf{C}) \end{bmatrix}}_{Bel(x_t)} = \eta \begin{bmatrix} 0.4 \\ 0.8 \end{bmatrix} * \underbrace{\begin{bmatrix} 0.74 \\ 0.26 \end{bmatrix}}_{\overline{Bel}(x_t)} = \eta \begin{bmatrix} 0.296 \\ 0.208 \end{bmatrix} = \begin{bmatrix} 0.58 \\ 0.42 \end{bmatrix}$$

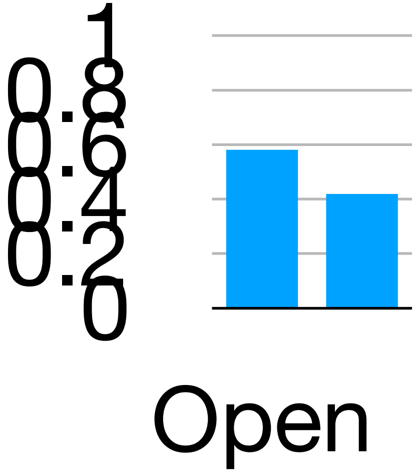
Example: Opening a Door

$\mathcal{X}$ = **O**PEN, **C**LOSED

$\mathcal{A}$ = **P**ULL, **L**EAVE

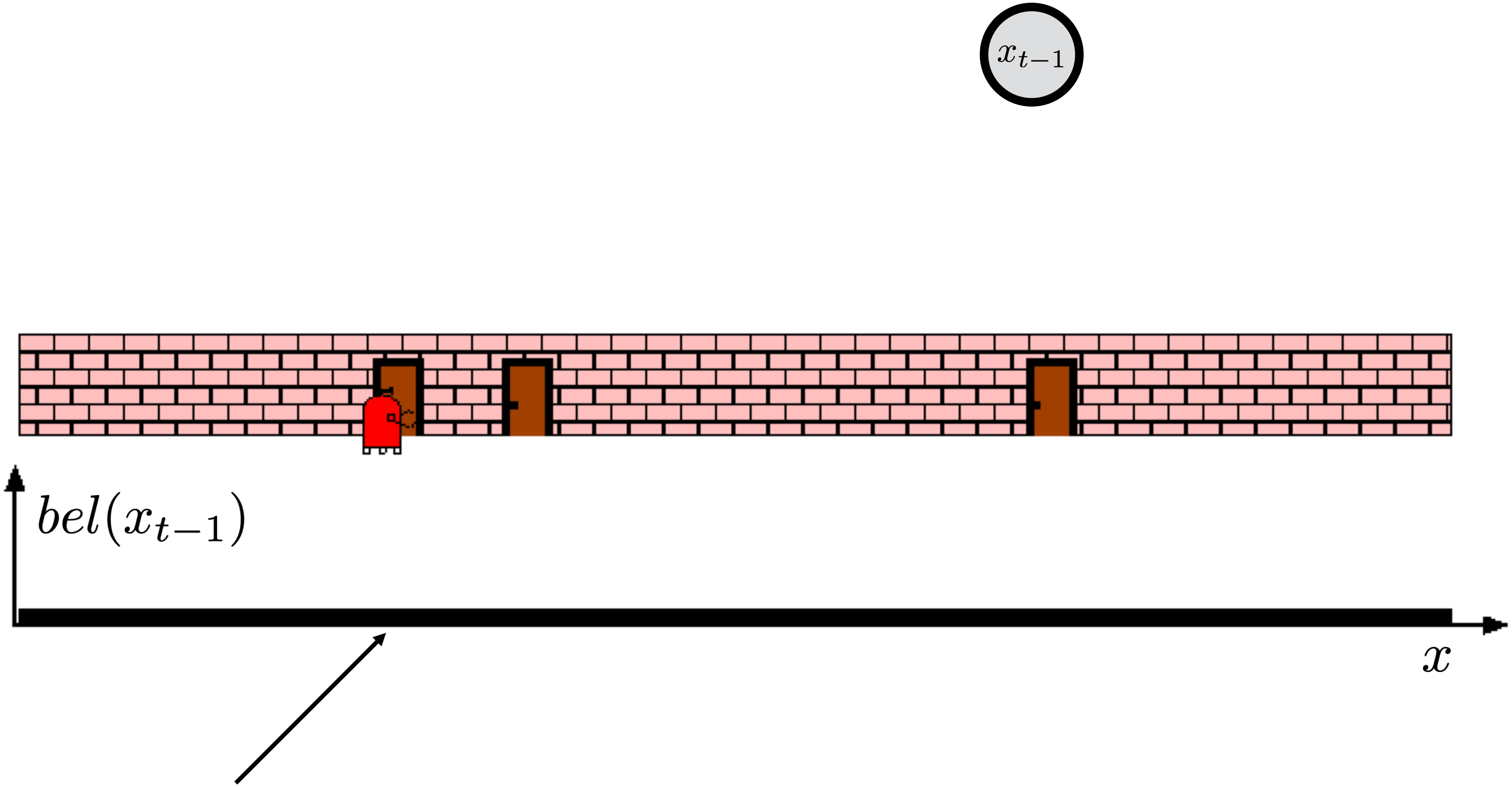
$\mathcal{Z}$ = **O**PEN, **C**LOSED

$$Bel(x_t) = \begin{bmatrix} 0.58 \\ 0.42 \end{bmatrix}$$



- Robot initially thought the door was open with 0.4 prob
- Robot took the PULL action, then thought the door was open with 0.74 prob
- Robot received a CLOSED measurement, now thinks open with 0.58 prob

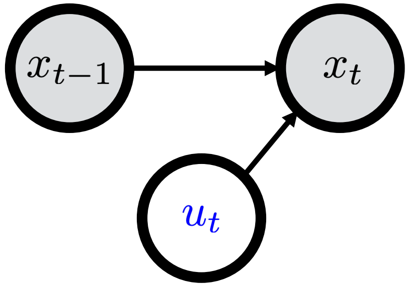
Robot lost in a 1-D hallway



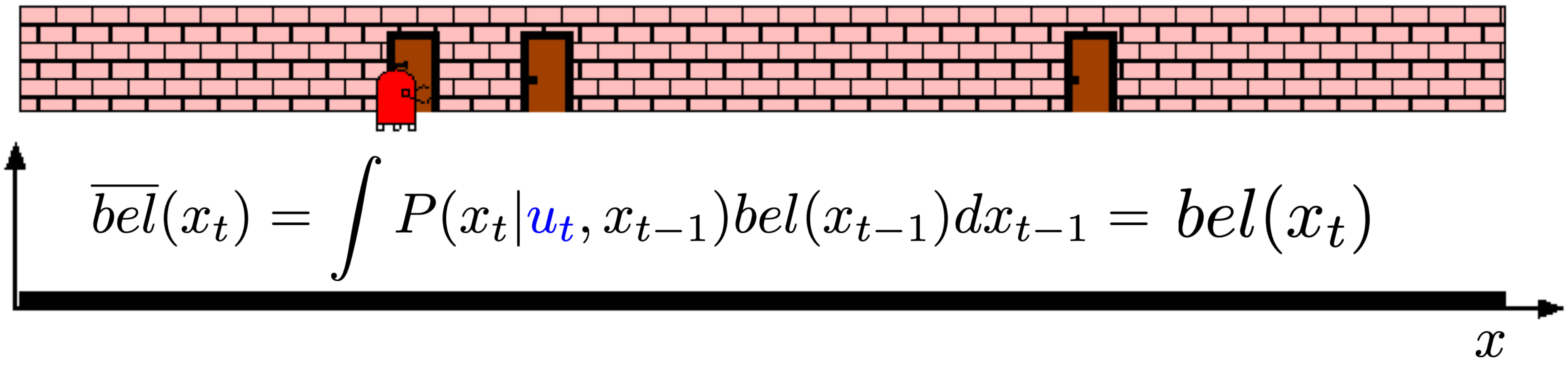
Uniform (robot could be anywhere)

Action at time t: NOP

$u_t = \text{NOP}$



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



$$\overline{bel}(x_t) = \int P(x_t | u_t, x_{t-1}) bel(x_{t-1}) dx_{t-1} = bel(x_t)$$

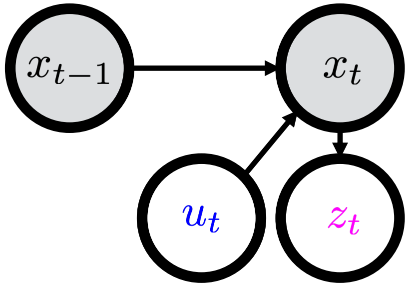
NOP action implies belief remains the same!

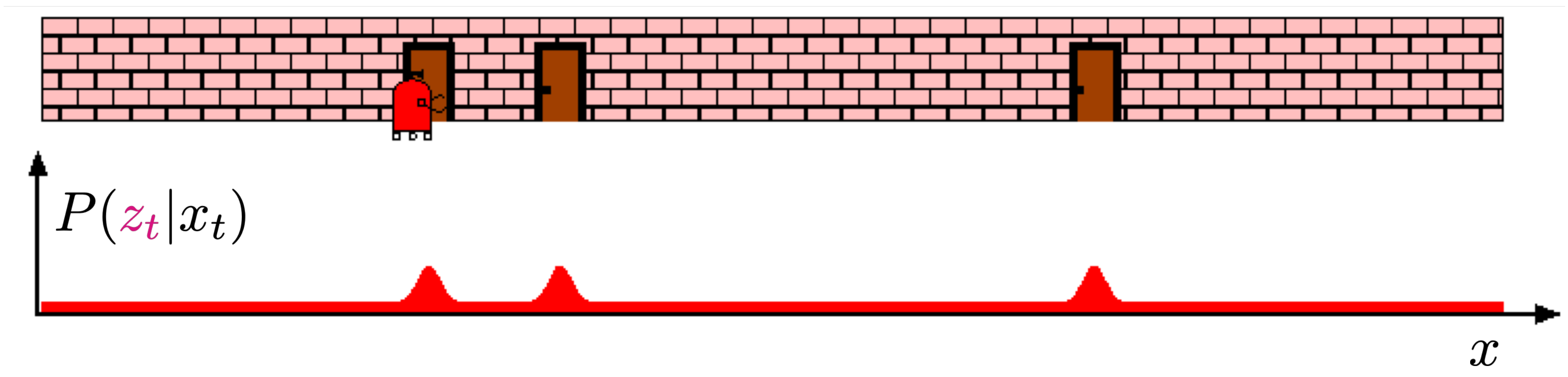
(still uniform — no idea where I am)

Measurement at time t: “Door”

$z_t = \text{Door}$

$P(z_t|x_t) = \mathcal{N}(\text{door centre}, 0.75m)$

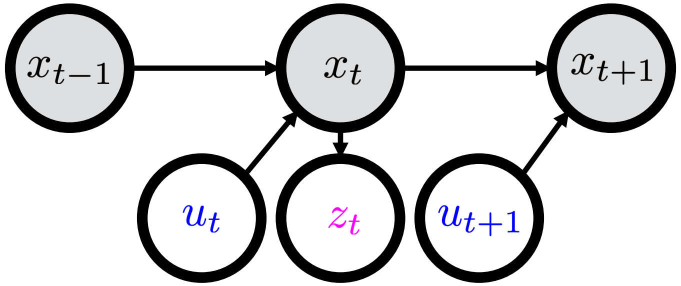


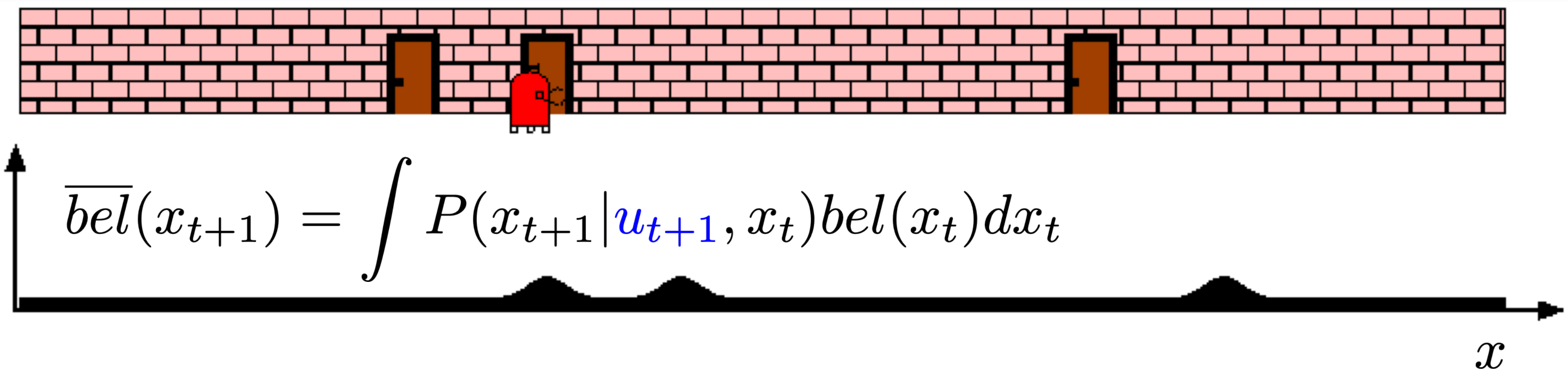


Action at time t+1: Move 3m right

$u_{t+1} = \text{3m right}$

$$P(x_{t+1}|u_{t+1}, x_t) = \mathcal{N}(x_t + u_{t+1}, 0.25m)$$



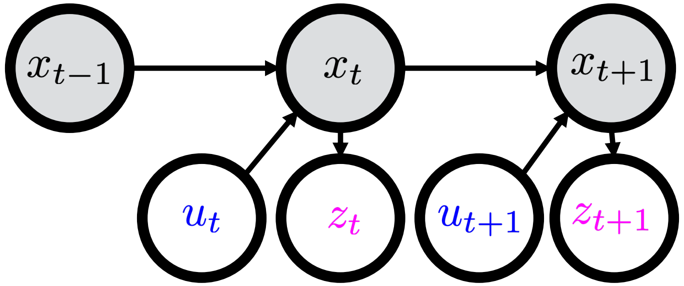


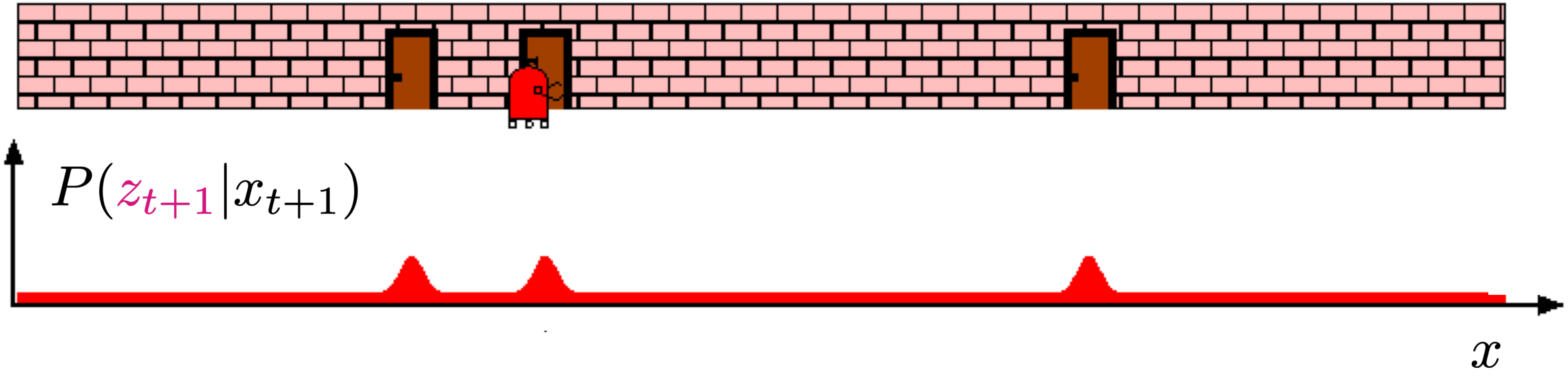
$$\overline{bel}(x_{t+1}) = \int P(x_{t+1}|u_{t+1}, x_t) bel(x_t) dx_t$$

Measurement at time t+1: "Door"

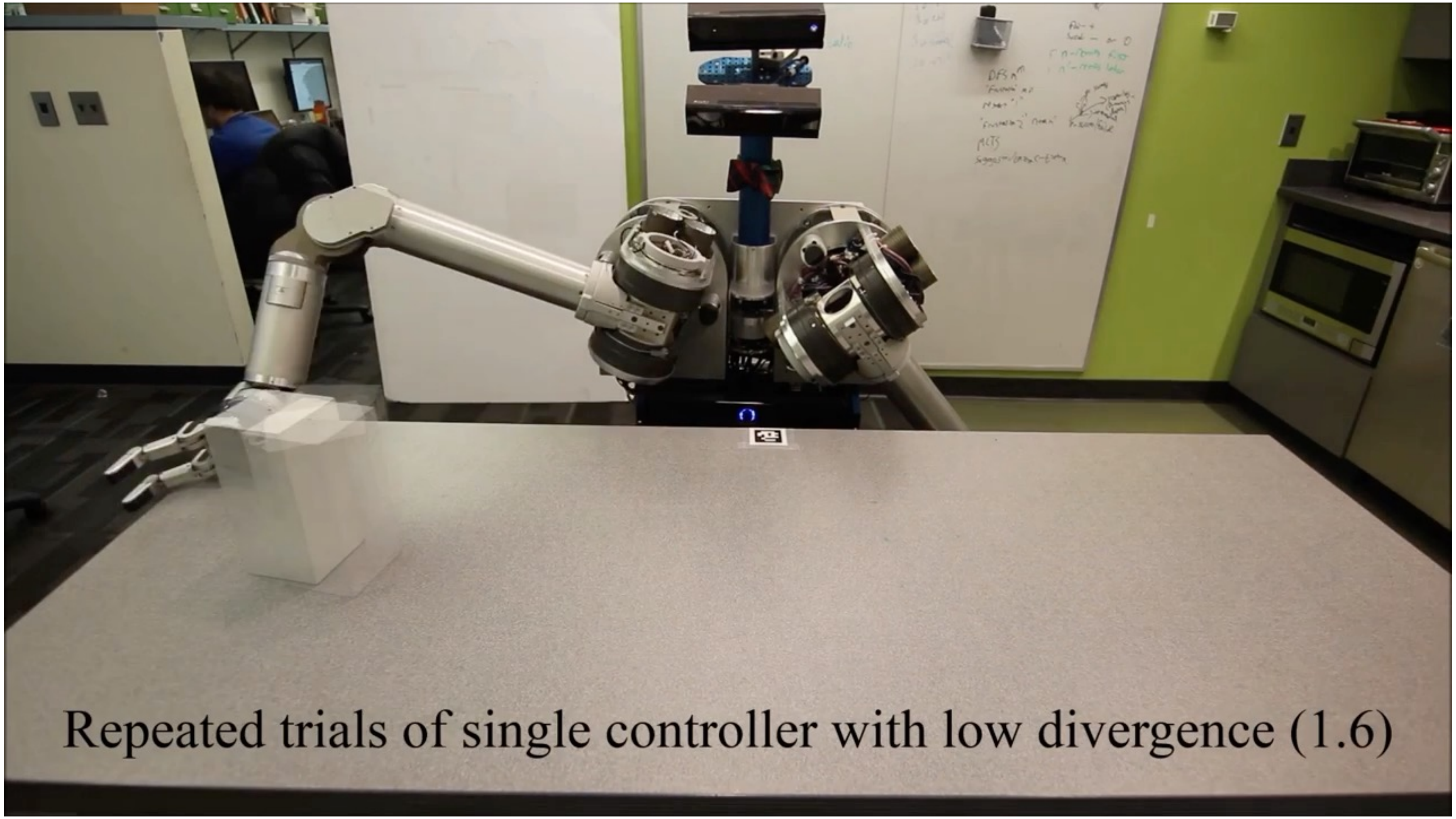
$z_{t+1} = \text{Door}$

$$P(z_{t+1}|x_{t+1}) = \mathcal{N}(\text{door centre}, 0.75m)$$





Do actions always increase uncertainty?



HTTPS://YOUTU.BE/BC1ATPRPPJC

Do measurements always reduce uncertainty?

- Level of uncertainty can be formalized as **entropy**
  - Low entropy if belief is tightly concentrated (e.g., concentrated on one state)
  - High entropy if belief is very spread out (e.g., uniform distribution)
- What if you reach into your pocket and can't find your keys?
  - Initially: low entropy (belief concentrated around pocket, some probability in other states around the house)
  - After: high entropy (very little probability in pocket, other states around the house have increased probability)

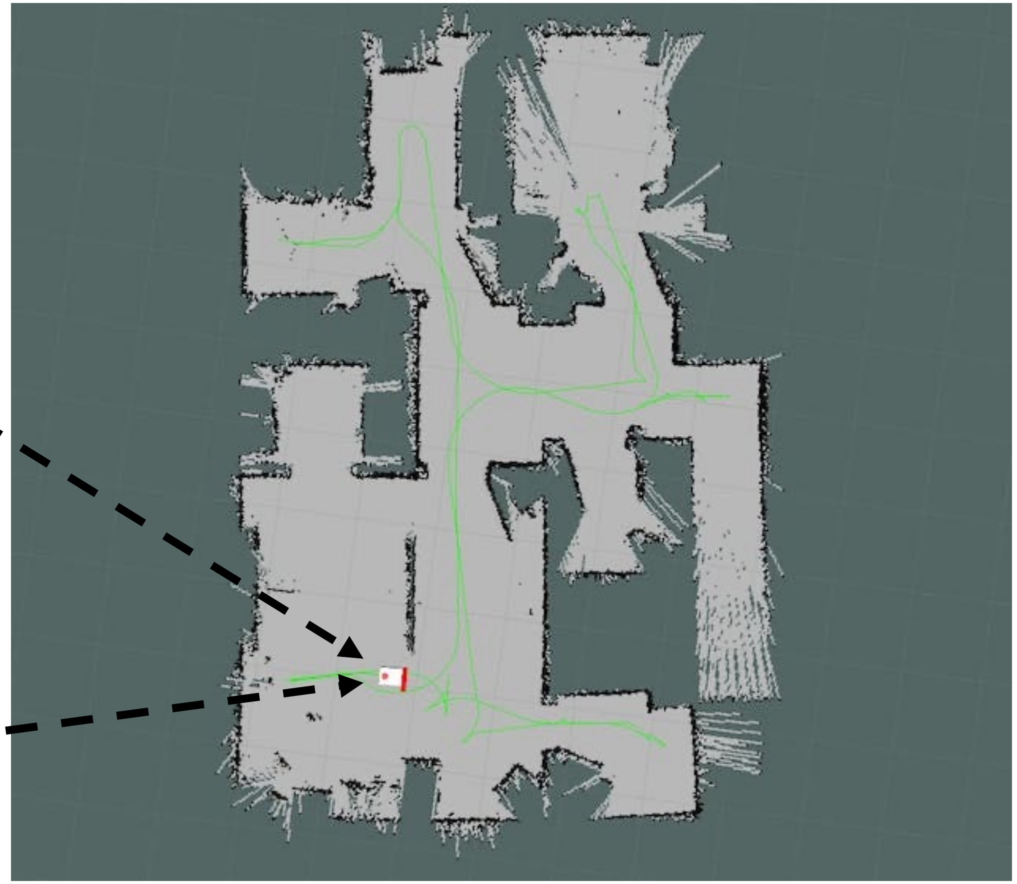
Ok this seems simple? What makes this hard!

$$Bel(x_t) = \eta \ P(z_t \mid x_t) \int P(x_t \mid u_t, x_{t-1}) \ Bel(x_{t-1}) \ dx_{t-1}$$

Tractable Bayesian inference is challenging in the general case

We will work out the conjugate prior and discrete case, leaving the MCMC/VI cases as an exercise

How does this connect back to our racecar?



Where am I in the world?

Lecture Outline

Recap

↓

Bayesian Filtering Examples

↓

Motion Models

↓

Observation Models

So what do we need to define to instantiate this?

Key Idea: Apply Markov to get a recursive update!

Step 0. Start with the belief at time step t-1

$$bel(x_{t-1})$$

Step 1: Prediction - push belief through dynamics given action

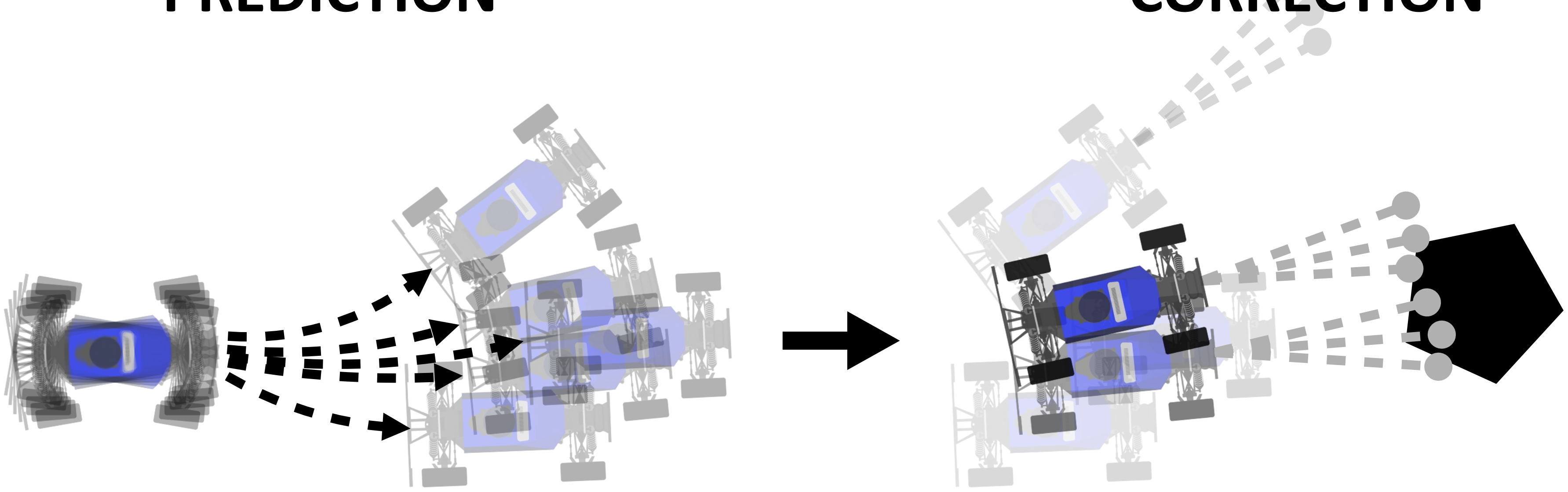
$$\overline{bel}(x_t) = \sum P(x_t|u_t, x_{t-1}) bel(x_{t-1})$$

Step 2: Correction - apply Bayes rule given measurement

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

Let’s ground this in the context of the car

PREDICTION **CORRECTION**



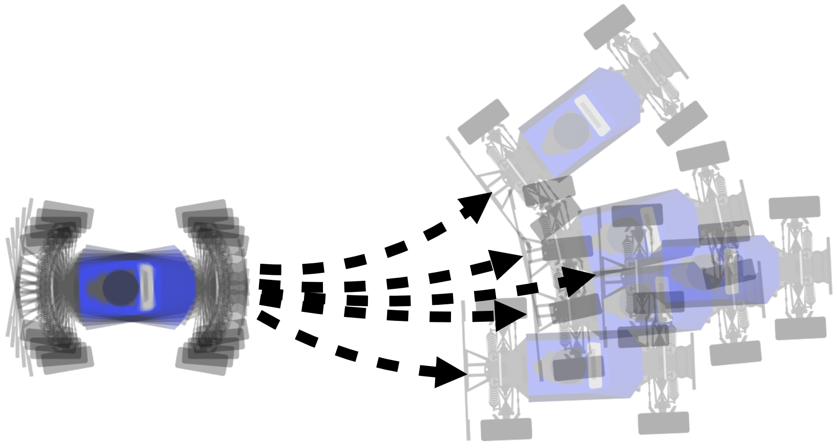
PREDICTION **CORRECTION**

$$P(x_t | u_t, x_{t-1})$$

$$P(z_t | x_t)$$

Motion Model

How do we know this?
→ it's just physics!



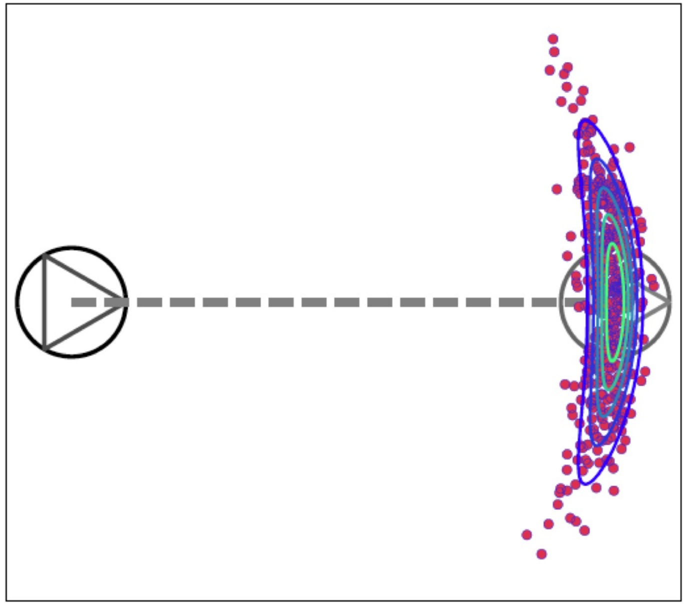
$$P(x_t | u_t, x_{t-1})$$

A Spectrum of Motion Models

Highest-fidelity models capturing everything we know

(Red Bull F1 Simulator)

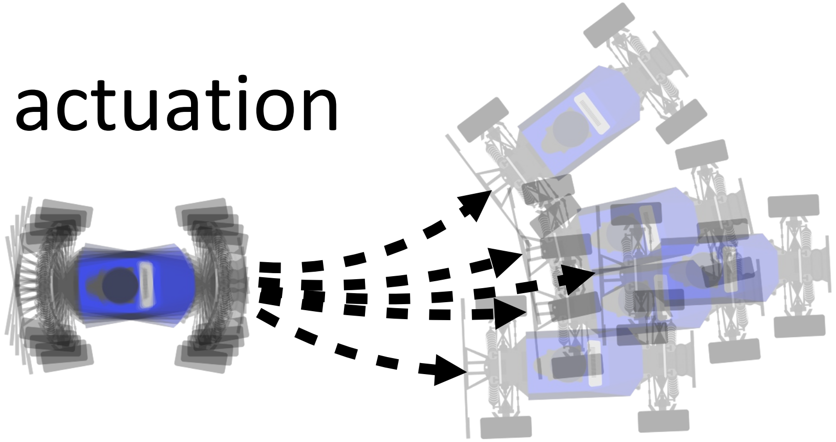
vs



Simple model
with lots of noise

Why is the motion model probabilistic?

- If we know how to write out equations of motion, shouldn't we be able to predict exactly where an object ends up?
- "All models are wrong, but some are useful" — George Box
  - Examples: ideal gas law, Coulomb friction
- Stochasticity is a catch-all for model error, actuation error, …



What defines a good motion model?

- In theory: try to accurately model the uncertainty (e.g., actuation errors)
- In practice…
  - We need just enough stochasticity to **explain any measurements** we'll see (Bayes filter uses measurements to hone in on the right state)
  - We need a model that can deal with **unknown unknowns** (No matter the model, we need to overestimate uncertainty)
  - We would like a model that is **computationally cheap** (Bayes filter repeatedly invokes this model to predict state after actions)
- Key idea: simple model + stochasticity

What motion model should I use for MuSHR?

- A **kinematic model** governs how wheel speeds map to robot velocities
- A **dynamic model** governs how wheel torques map to robot accelerations
- For MuSHR, we'll ignore dynamics and focus on kinematics (assuming the wheel actuators can set speed directly)
- Other assumptions: wheels roll on hard, flat, horizontal ground without slipping

Kinematic Car Model

$$\dot{x} = f(x, u)$$

$$\begin{bmatrix} x \\ y \\ \theta \end{bmatrix}$$

X-COORDINATE

Y-COORDINATE

HEADING

$$\begin{bmatrix} v \\ \delta \end{bmatrix}$$

SPEED

STEERING ANGLE

Kinematic Car Model

$$\dot{x} = f(x, u)$$

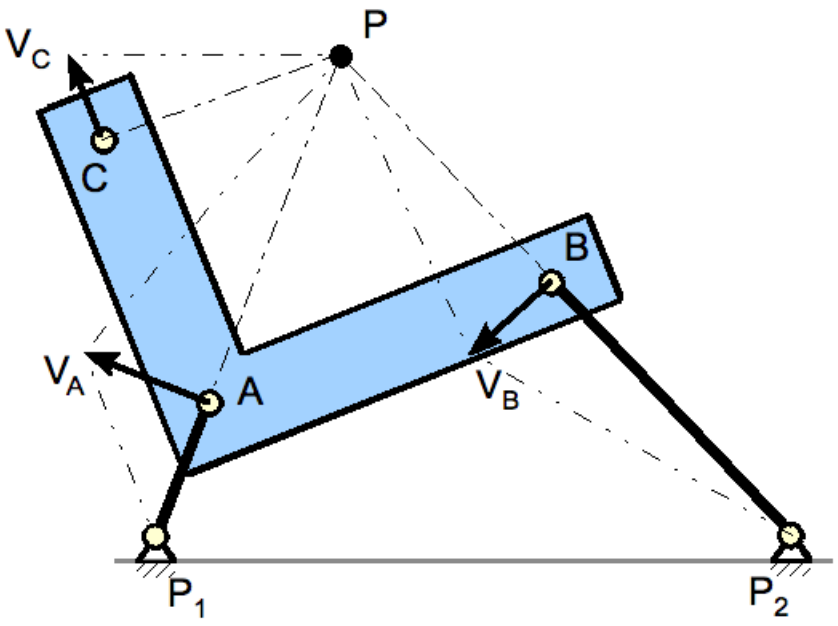
→ INTEGRATE

$$\begin{bmatrix} x_{t-1} + \Delta x \\ y_{t-1} + \Delta y \\ \theta_{t-1} + \Delta \theta \end{bmatrix} = \begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix}$$

→ ADD NOISE

$$P(x_t | u_t, x_{t-1})$$

Definition: Instant Center of Rotation (CoR)



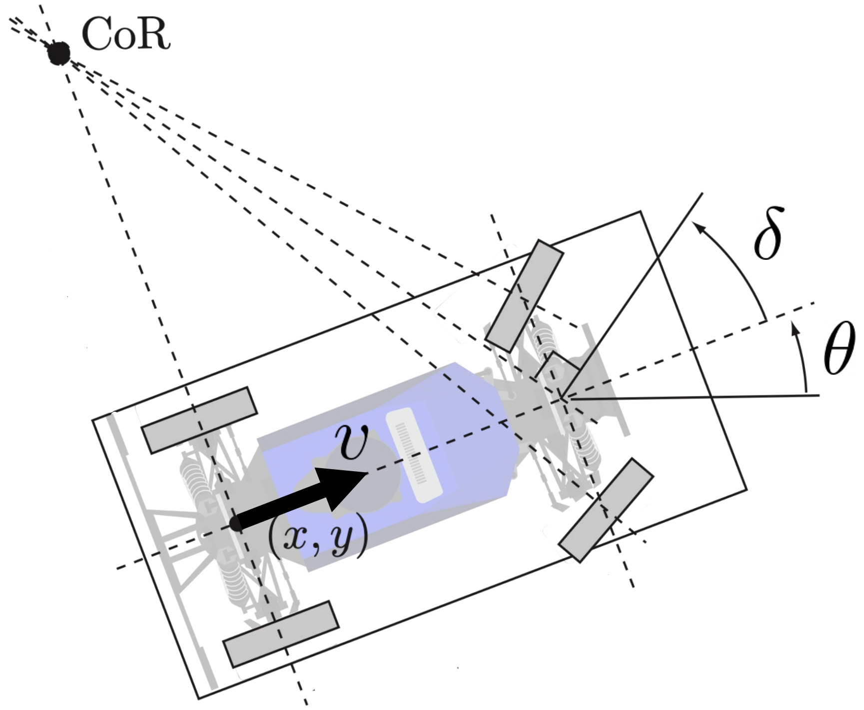
A planar **rigid body** undergoing a **rigid transformation** can be viewed as undergoing a **pure rotation** about an instant center of rotation.

rigid body: a non-deformable object

rigid transformation: a combined rotation and translation

HTTPS://EN.WIKIPEDIA.ORG/WIKI/INSTANT_CENTRE_OF_ROTATION

Equations of Motion

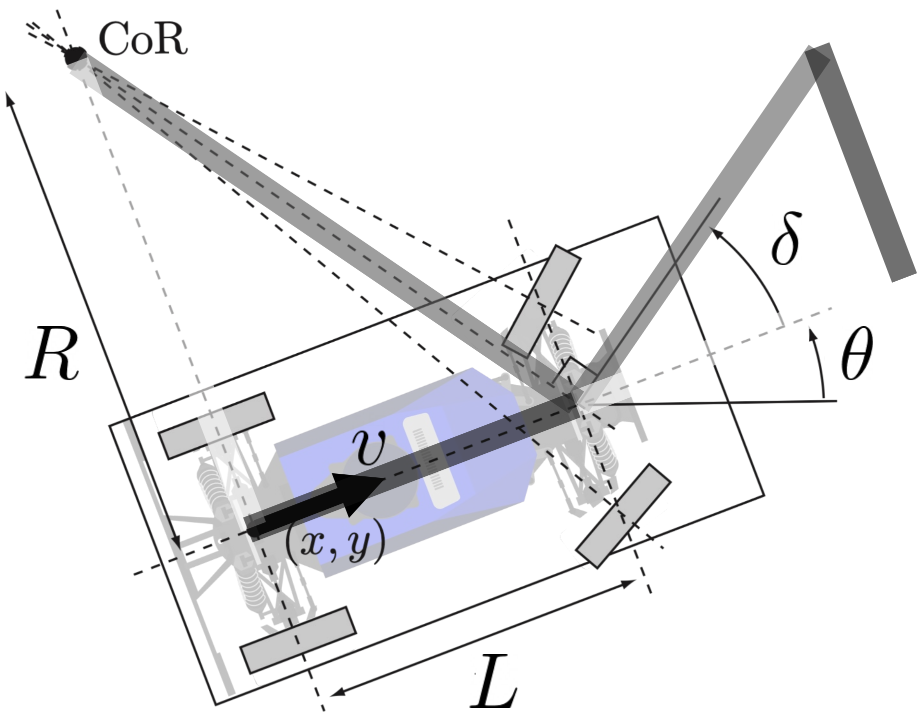


$$\dot{x} = v\cos\theta$$

$$\dot{y} = v\sin\theta$$

$$\dot{\theta} = \textbf{?}$$

Equations of Motion



$$\dot{x} = v \cos\theta$$

$$\dot{y} = v \sin\theta$$

$$\dot{\theta} = \omega = \frac{v}{R} = \frac{v \tan\delta}{L}$$

$$\tan\delta = \frac{L}{R} \rightarrow R = \frac{L}{\tan\delta}$$

Kinematic Car Model

$$\dot{x} = f(x, u) \quad \xrightarrow{\text{INTEGRATE}} \quad \begin{bmatrix} x_{t-1} + \Delta x \\ y_{t-1} + \Delta y \\ \theta_{t-1} + \Delta\theta \end{bmatrix} = \begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix}$$

Integrate the Kinematics Numerically

$$\dot{x} = v\cos\theta$$
$$\dot{y} = v\sin\theta$$
$$\dot{\theta} = \frac{v}{L}\tan\delta$$

Assume that steering angle is **piecewise constant** between t and t’

Integrate the Kinematics Numerically

$$\Delta x = \int_t^{t'} v\cos\theta(t)dt = \int_t^{t'} \frac{v\cos\theta}{\dot\theta}\frac{d\theta}{dt}dt = \frac{v}{\dot\theta}\int_\theta^{\theta'} \cos\theta d\theta$$

$$= \frac{L}{\tan\delta}(\sin\theta' - \sin\theta)$$

$$\Delta y = \frac{L}{\tan\delta}(\cos\theta - \cos\theta')$$

$$\Delta\theta = \int_t^{t'} \dot\theta dt = \frac{v}{L}\tan\delta\Delta t$$

$$\dot x = v\cos\theta$$
$$\dot y = v\sin\theta$$
$$\dot\theta = \frac{v}{L}\tan\delta$$

Assume that steering angle is **piecewise constant** between t and t'

Kinematic Car Update

$$\theta_t = \theta_{t-1} + \Delta\theta = \theta_{t-1} + \frac{v}{L} \tan\delta \Delta t$$

$$x_t = x_{t-1} + \Delta x = x_{t-1} + \frac{L}{\tan\delta}(\sin\theta_t - \sin\theta_{t-1})$$

$$y_t = y_{t-1} + \Delta y = y_{t-1} + \frac{L}{\tan\delta}(\cos\theta_{t-1} - \cos\theta_t)$$

Kinematic Car Model

$$\dot{x} = f(x, u)$$

→ INTEGRATE

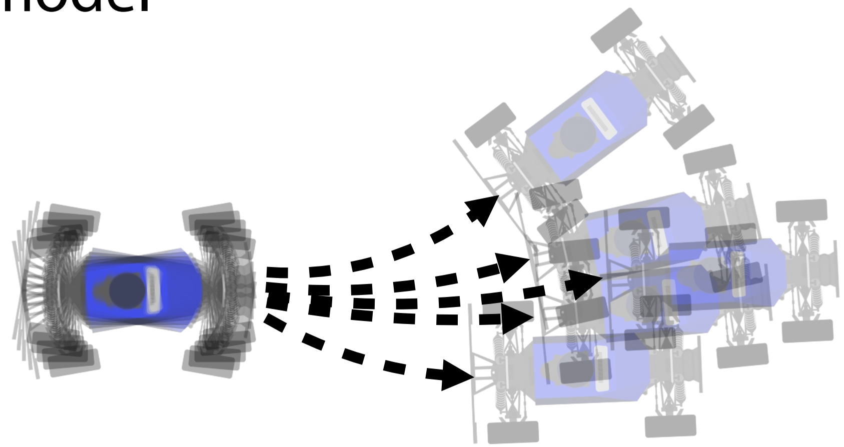
$$\begin{bmatrix} x_{t-1} + \Delta x \\ y_{t-1} + \Delta y \\ \theta_{t-1} + \Delta \theta \end{bmatrix} = \begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix}$$

→ ADD NOISE

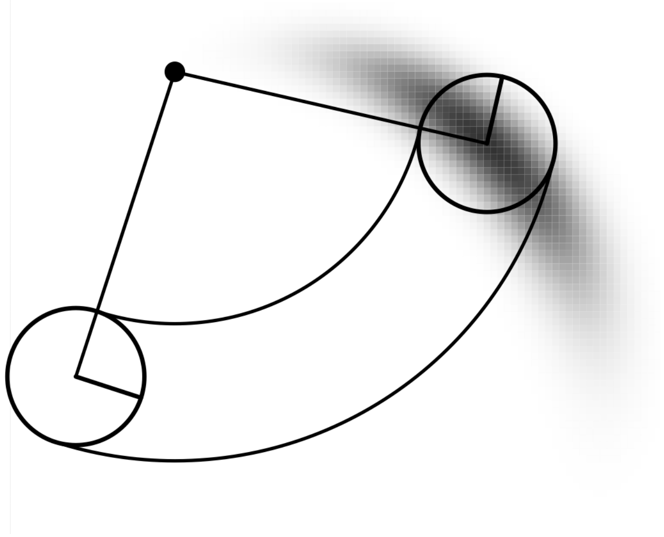
$$P(x_t | u_t, x_{t-1})$$

Why is the kinematic car model probabilistic?

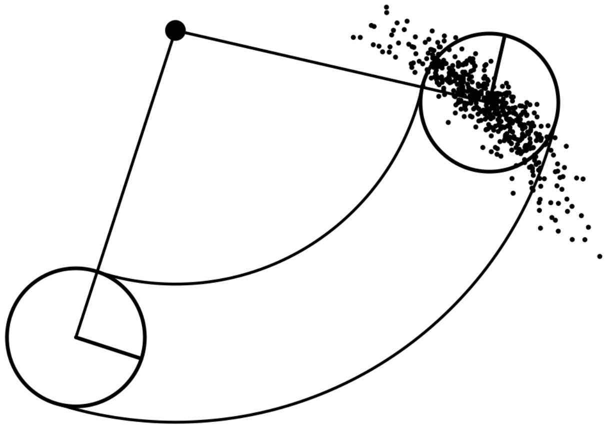
- Control signal error: voltage discretization, communication lag
- Unmodeled physics parameters: friction of carpet, tire pressure
- Incorrect physics: ignoring tire deformation, ignoring wheel slippage
- Our probabilistic motion model
  - Add noise to control before propagating through model
  - Add noise to state after propagating through model



Motion Model Summary



MOTION MODEL
PROB. DENSITY FUNCTION



MOTION MODEL
SAMPLES

- Write down the deterministic equations of motion (kinematic car model)
- Introduce stochasticity to account against various factors

Lecture Outline

Recap

↓

Bayesian Filtering Examples

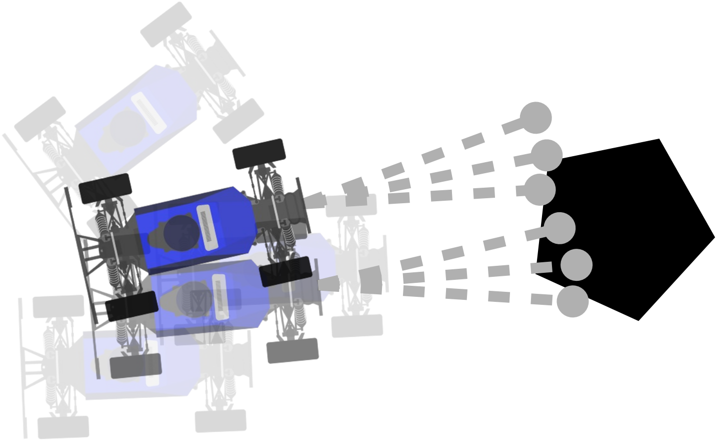
↓

Motion Models

↓

Observation Models

Sensor Model

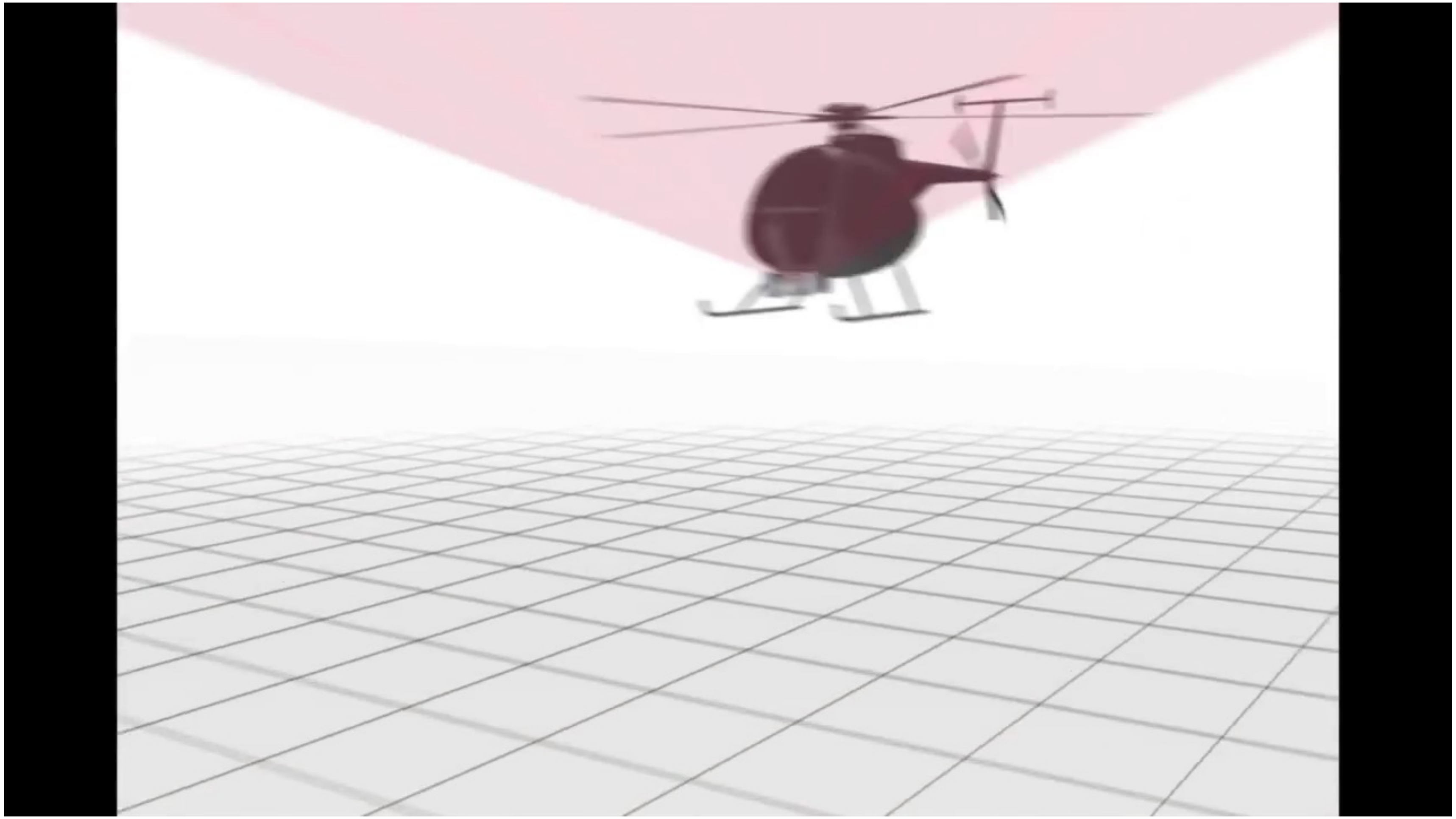


$$P(z_t | x_t)$$

How Does LIDAR Work?

HTTPS://YOUTU.BE/NZKVF1CXE8S

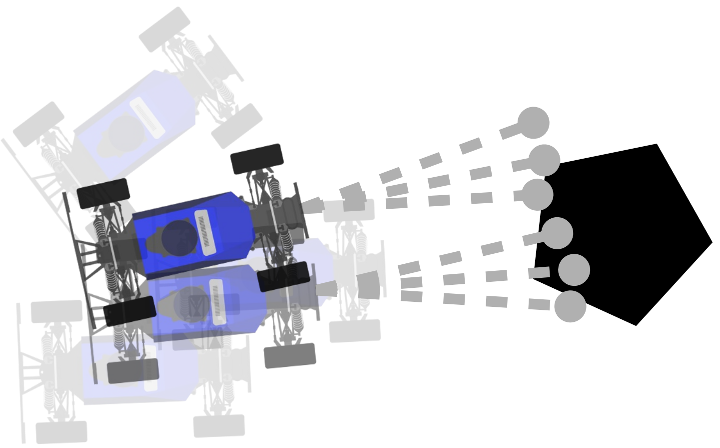
LIDAR in the Real World



HTTPS://YOUTU.BE/I8YV5D8CPOC

Why is the sensor model probabilistic?

- Incomplete/incorrect map: pedestrians, objects moving around
- Unmodeled physics: lasers go through glass
- Sensing assumptions: light interference from other sensors, multiple laser returns (bouncing off multiple objects)



What defines a good sensor model?

- Overconfidence can be catastrophic for Bayes filter
- LIDAR is very precise, but has distinct modes of failure
  - Anticipate specific types of failures, and add stochasticity accordingly

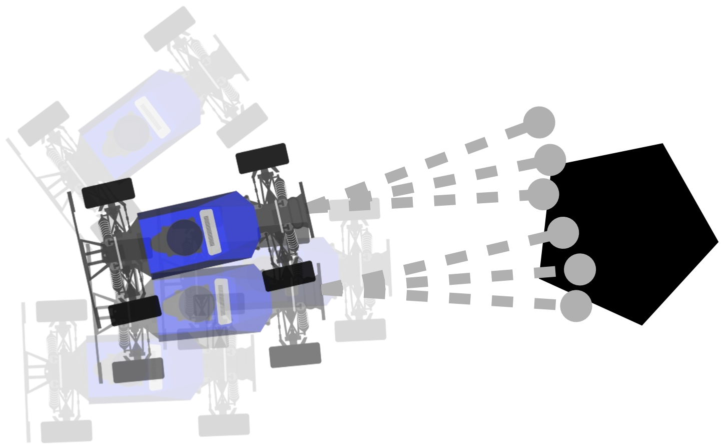
What sensor model should I use for MuSHR?

$$P(z_t|x_t) \rightarrow P(z_t|x_t, m)$$

LASER SCAN

STATE

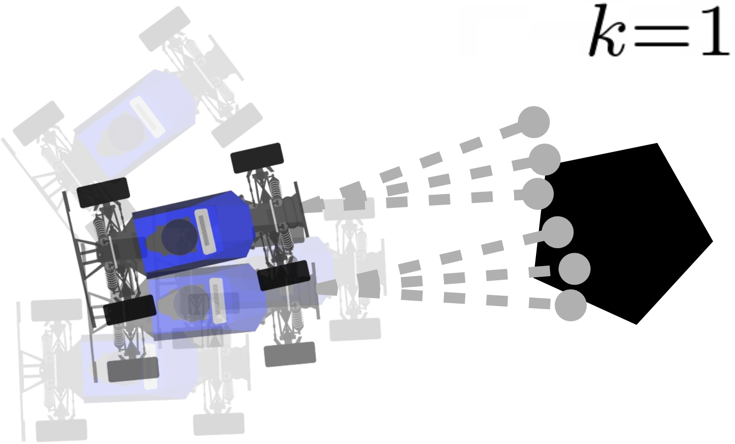
MAP



Assumption: Conditional Independence

$$P(z_t | x_t, m) = P(z_t^1, z_t^2, \cdots, z_t^K | x_t, m)$$

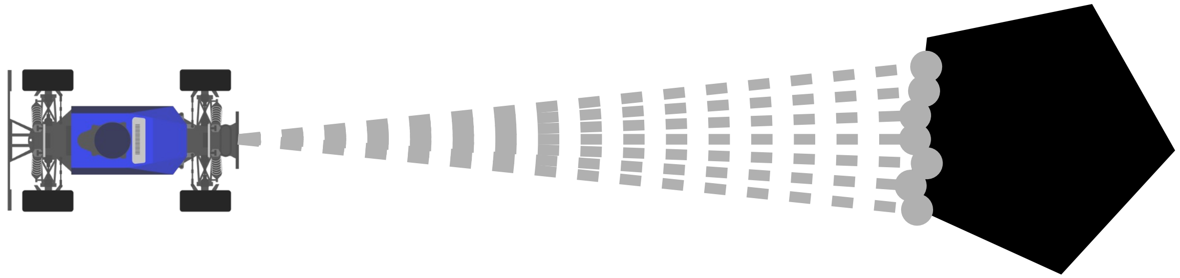
$$= \prod_{k=1}^{K} P(z_t^k | x_t, m)$$



Assumption: Conditional Independence

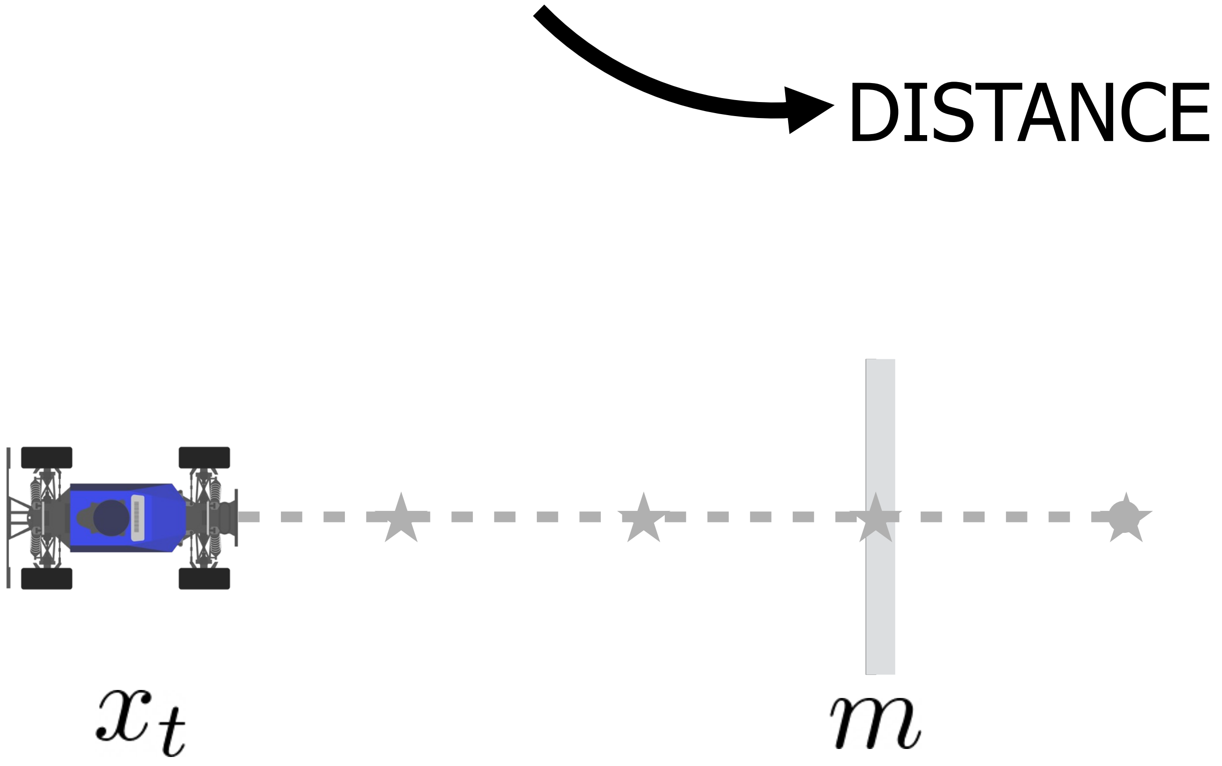
$$P(z_t|x_t, m) = P(z_t^1, z_t^2, \cdots, z_t^K|x_t, m)$$

$$= \prod_{k=1}^{K} P(z_t^k|x_t, m)$$



Single Beam Sensor Model

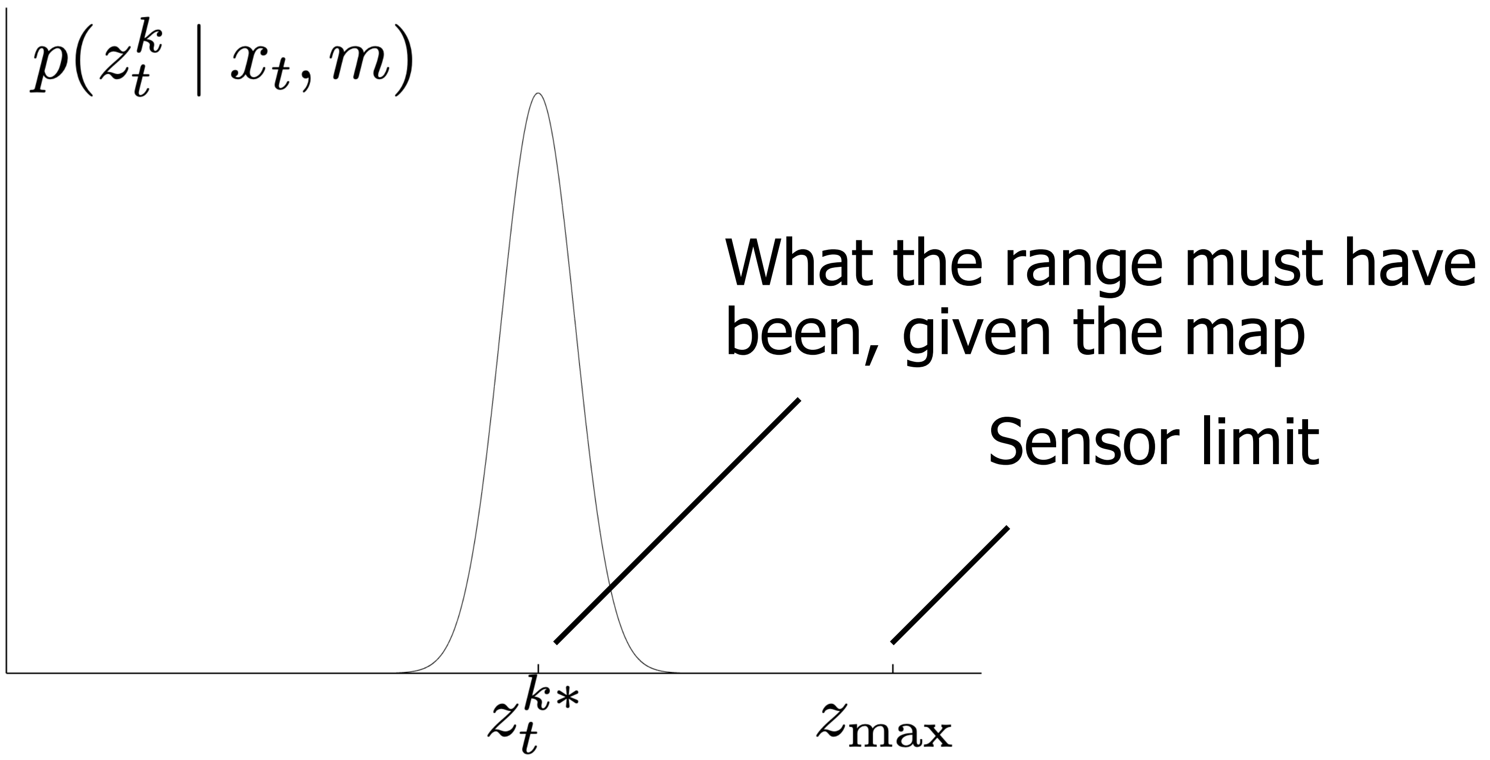
$$P(z_t^k | x_t, m)$$



Typical Sources of Stochasticity

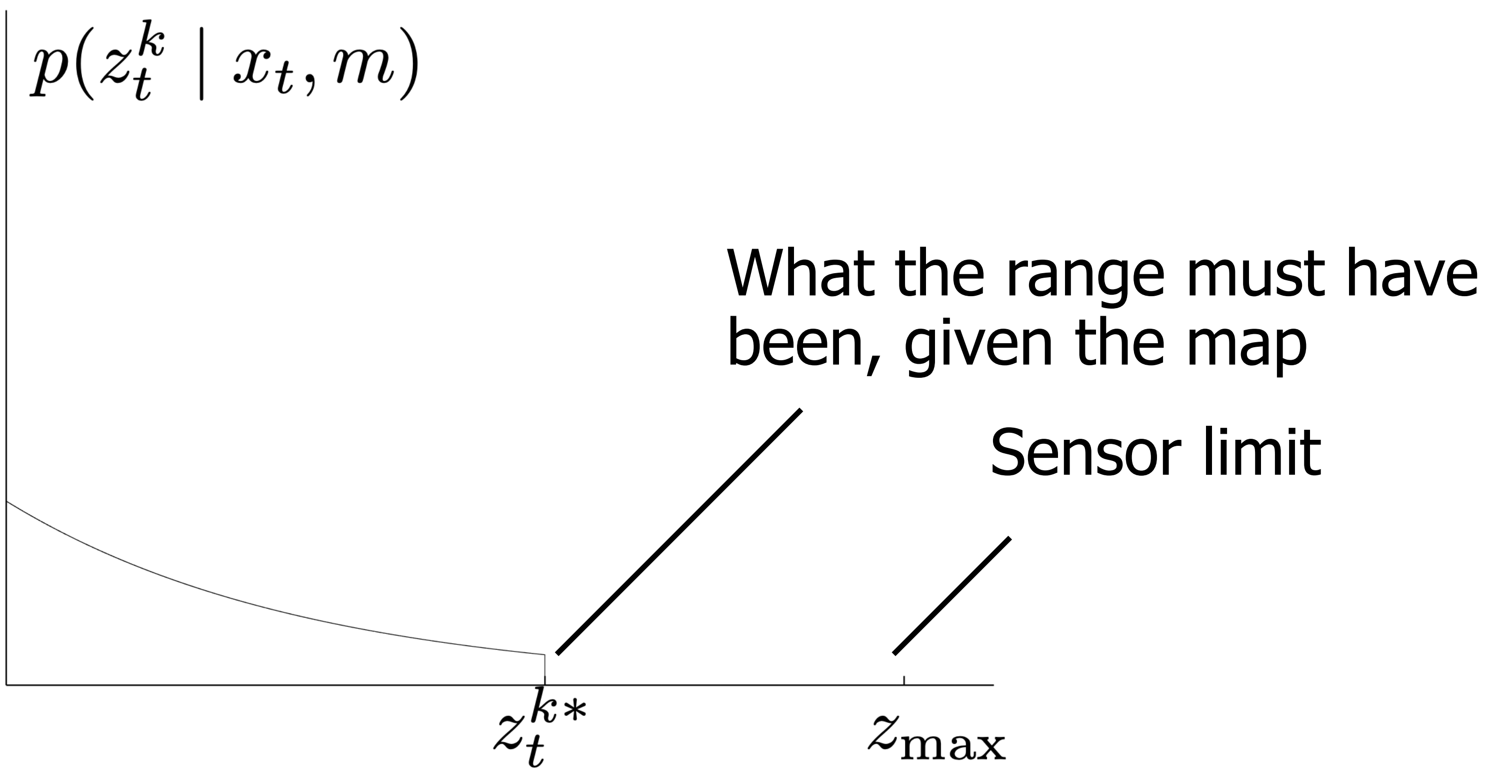
1. Correct range (distance) with local measurement noise
2. Unexpected objects
3. Sensor failures
4. Random measurements

Factor 1: Local Measurement Noise



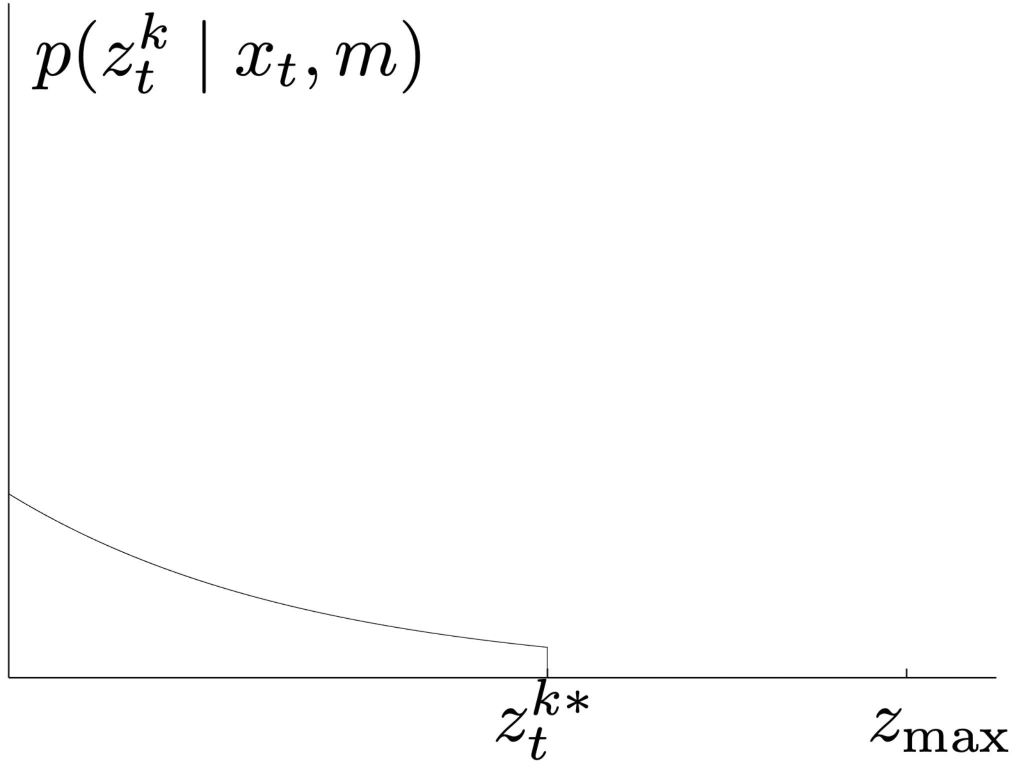
$$p_{\text{hit}}(z_t^k \mid x_t, m) = \begin{cases} \eta\, \mathcal{N}(z_t^k; z_t^{k*}, \sigma_{\text{hit}}^2) & \text{if } 0 \le z_t^k \le z_{\max} \\ 0 & \text{otherwise} \end{cases}$$

Factor 2: Unexpected Objects



$$p_{\text{short}}(z_t^k \mid x_t, m) \;=\; \begin{cases} \eta\ \lambda_{\text{short}}\ e^{-\lambda_{\text{short}}\, z_t^k} & \text{if } 0 \le z_t^k \le z_t^{k*} \\ 0 & \text{otherwise} \end{cases}$$

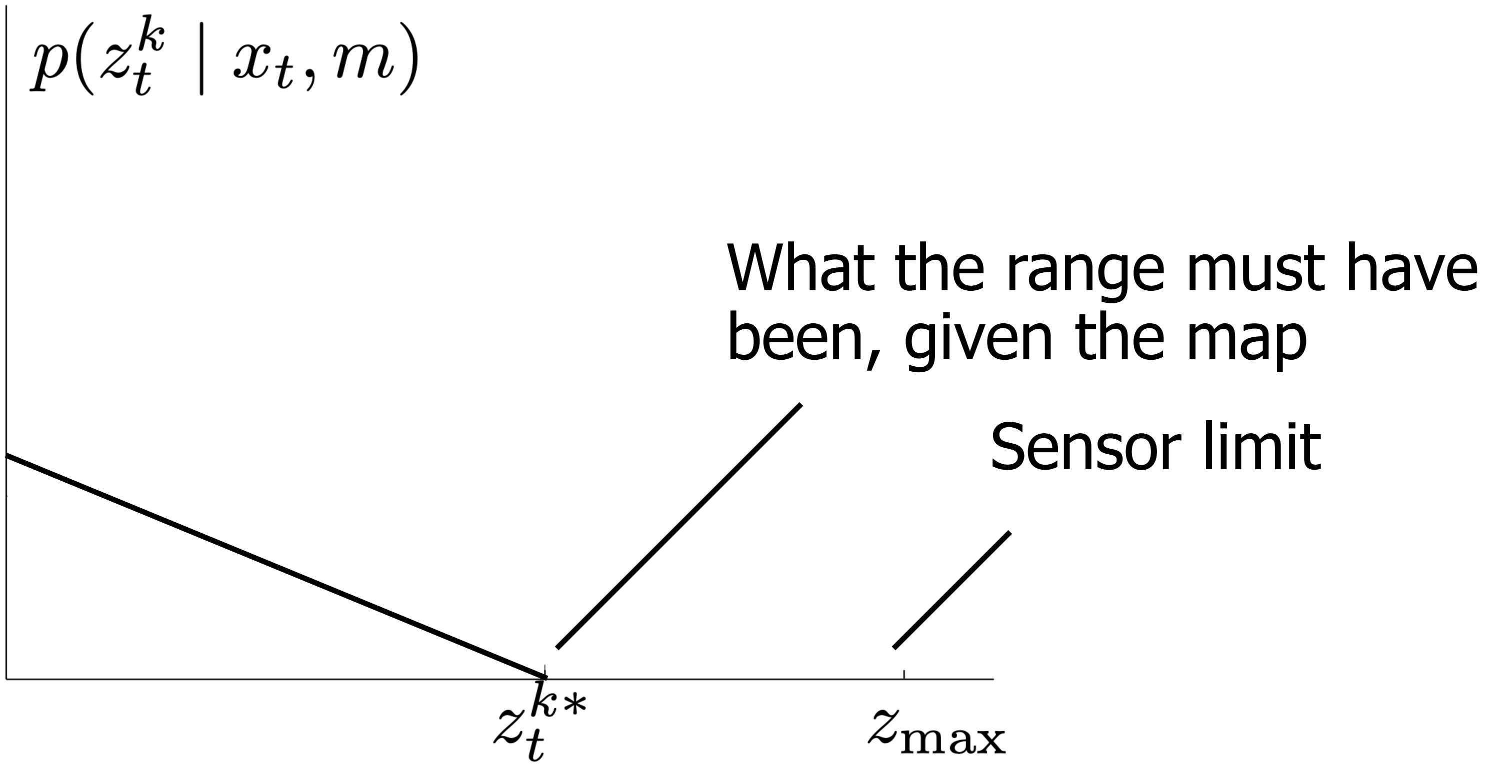
Factor 2: Unexpected Objects



1								128
0	1							64
0	0	1						32
0	0	0	1					16
0	0	0	0	1				8
0	0	0	0	0	1			4
0	0	0	0	0	0	1		2
0	0	0	0	0	0	0	1	1

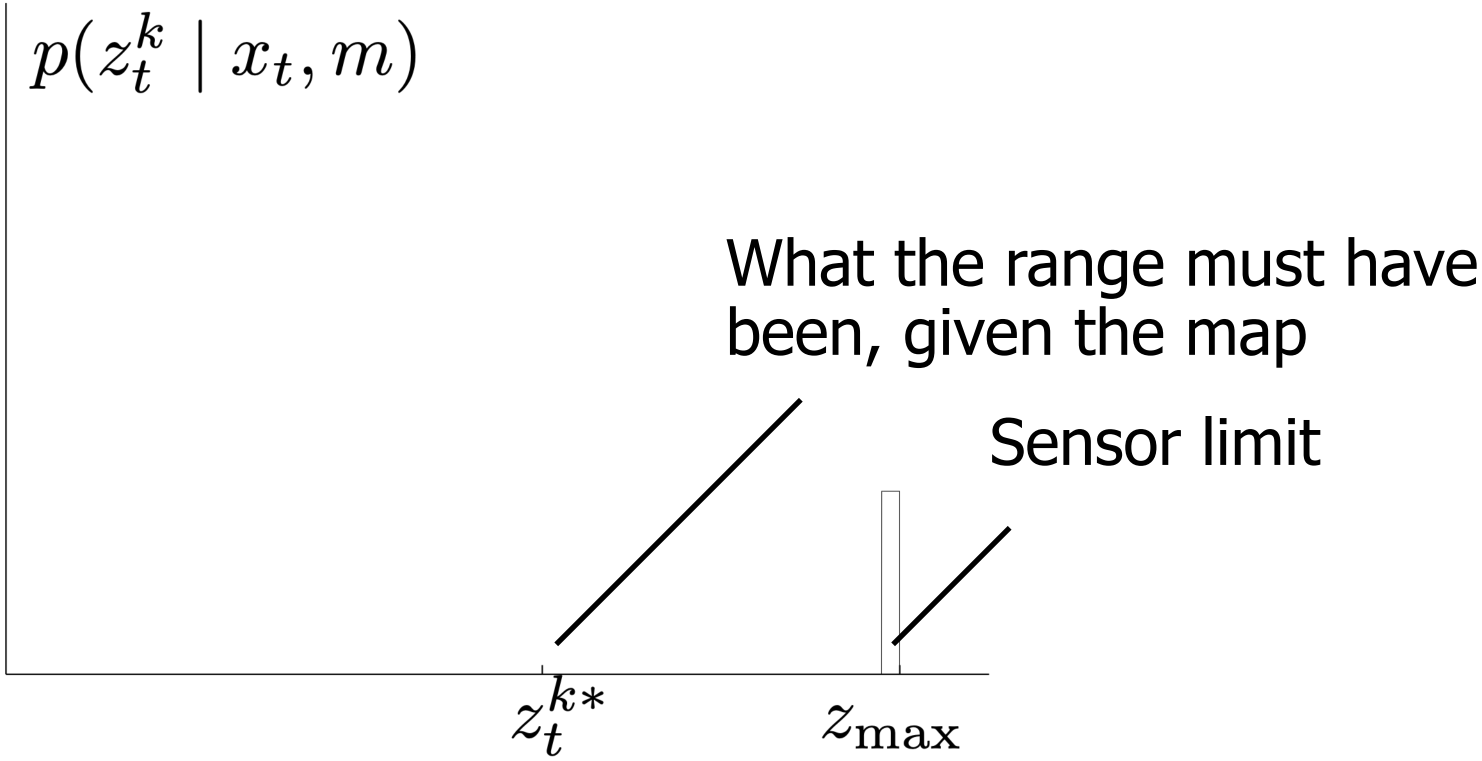
$$p_{\text{short}}(z_t^k \mid x_t, m) = \begin{cases} \eta\, \lambda_{\text{short}}\, e^{-\lambda_{\text{short}} z_t^k} & \text{if } 0 \le z_t^k \le z_t^{k*} \\ 0 & \text{otherwise} \end{cases}$$

Factor 2: Unexpected Objects (Project)



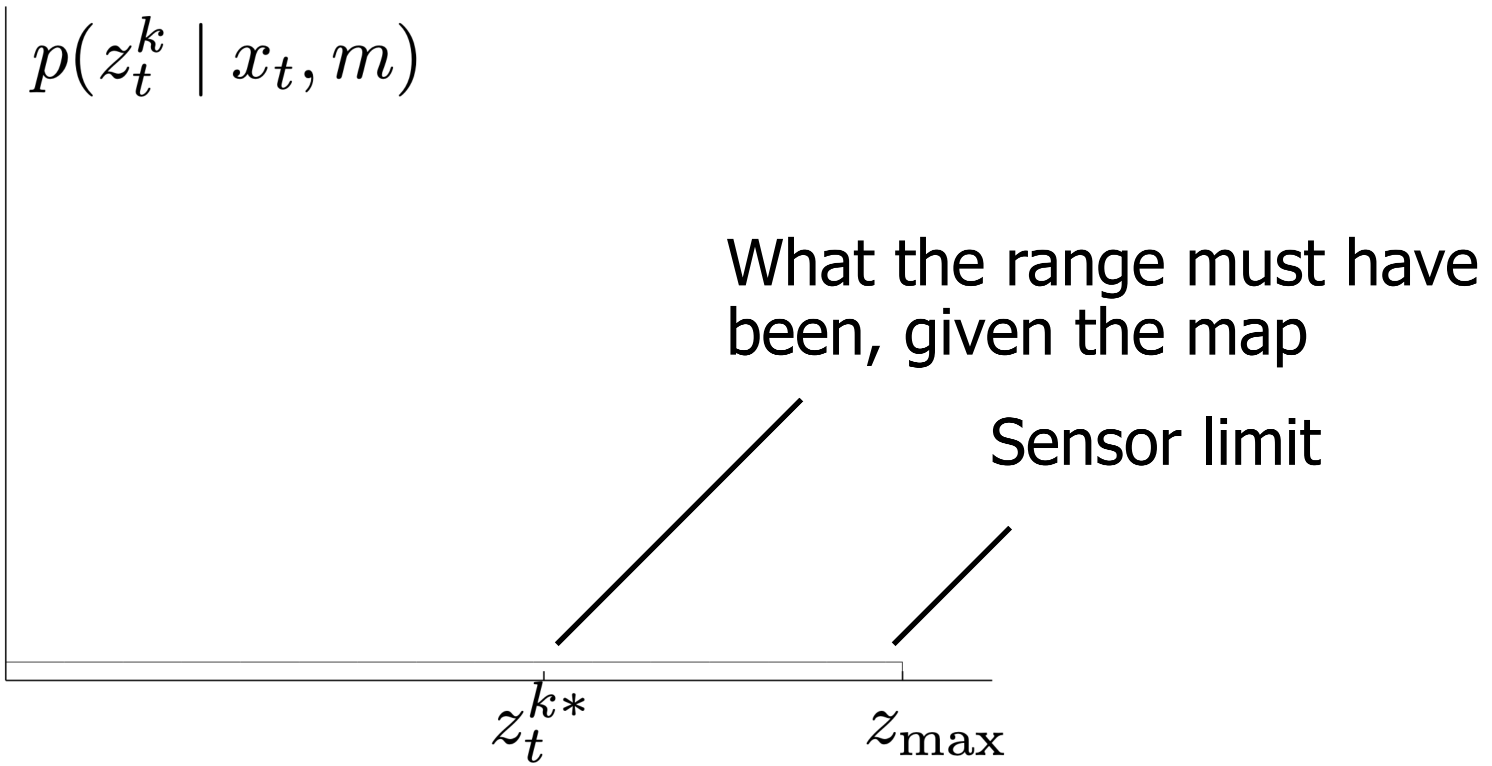
$$p_{\text{short}}(z_t^k | x_t, m) = \begin{cases} 2\frac{z_t^{k^*} - z_t^k}{z_t^{k^*}} & \text{if } z_t^k < z_t^{k^*} \\ 0 & \text{otherwise} \end{cases}$$

Factor 3: Sensor Failures



$$p_{\max}(z_t^k \mid x_t, m) \quad = \quad I(z = z_{\max}) \quad = \quad \begin{cases} 1 & \text{if } z = z_{\max} \\ 0 & \text{otherwise} \end{cases}$$

Factor 4: Random Measurements

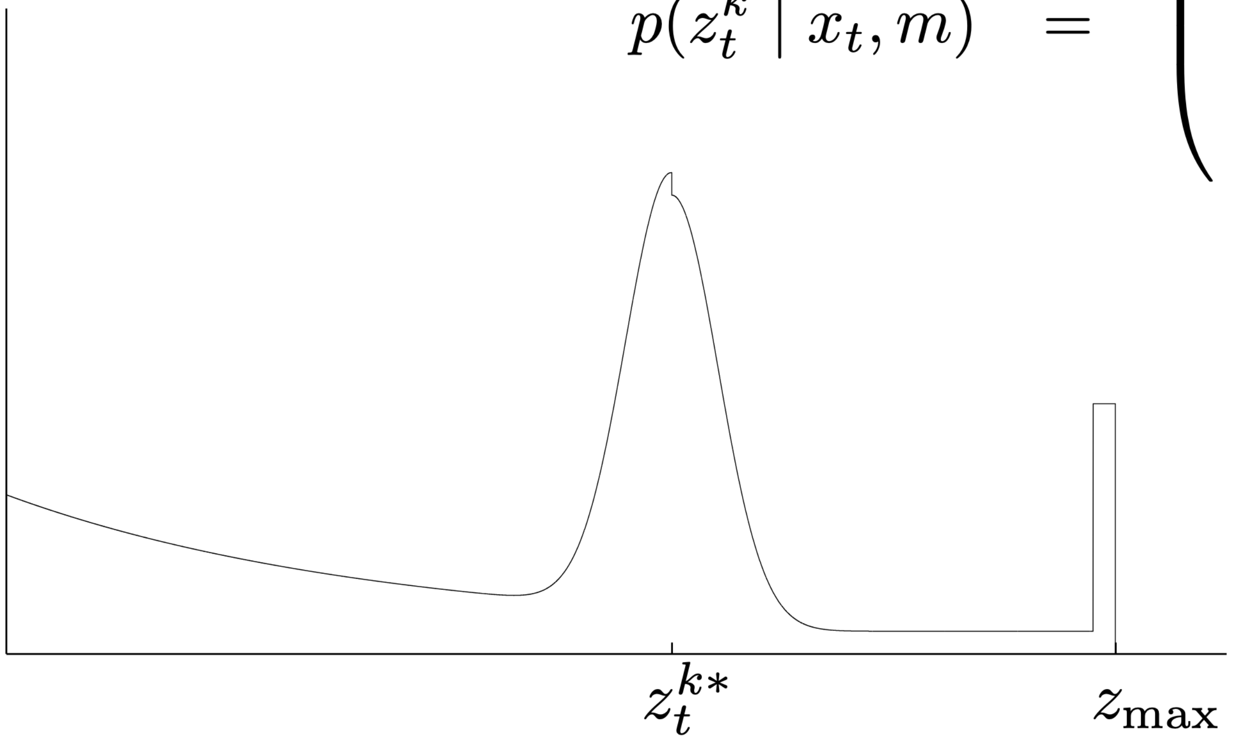


$$p_{\text{rand}}(z_t^k \mid x_t, m) \quad = \quad \begin{cases} \frac{1}{z_{\max}} & \text{if } 0 \le z_t^k < z_{\max} \\ 0 & \text{otherwise} \end{cases}$$

Putting It All Together

$$p(z_t^k \mid x_t, m) \quad = \quad \begin{pmatrix} z_{\text{hit}} \\ z_{\text{short}} \\ z_{\text{max}} \\ z_{\text{rand}} \end{pmatrix}^T \cdot \begin{pmatrix} p_{\text{hit}}(z_t^k \mid x_t, m) \\ p_{\text{short}}(z_t^k \mid x_t, m) \\ p_{\text{max}}(z_t^k \mid x_t, m) \\ p_{\text{rand}}(z_t^k \mid x_t, m) \end{pmatrix}$$

Weights sum to 1



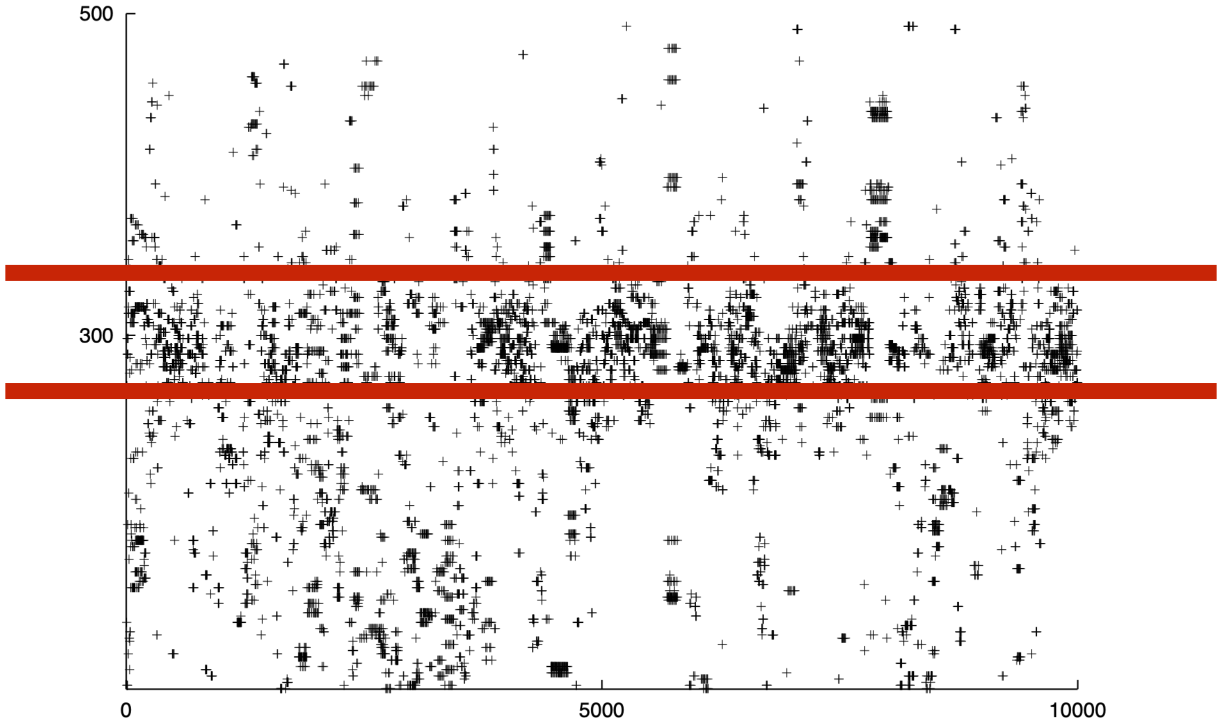
LIDAR Model Algorithm

$$P(z_t|x_t, m) = \prod_{k=1}^{K} P(z_t^k|x_t, m)$$

1. Use robot **state** to compute the sensor's pose on the **map**
2. Ray-cast from the sensor to compute a simulated laser scan
3. For each beam, compare ray-casted distance to **real laser scan distance**
4. Multiply all probabilities to compute the likelihood of that real laser scan

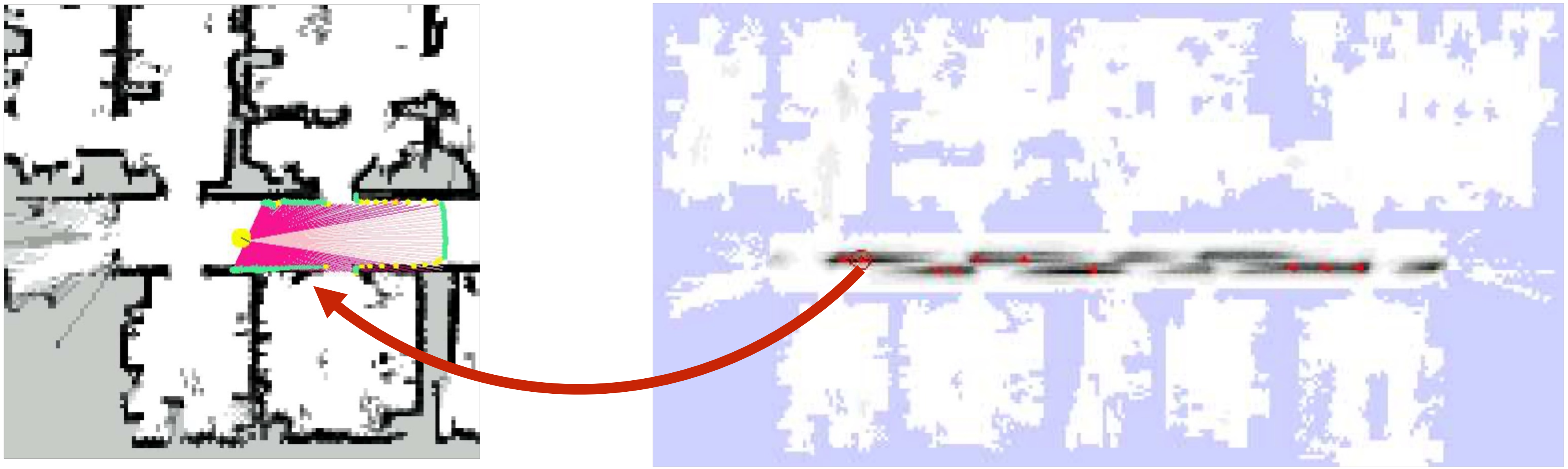
Tuning Single Beam Parameters

- Offline: collect lots of data and optimize parameters



Tuning Single Beam Parameters

- Online: simulate a scan and plot the likelihood from different positions



Actual scan | Likelihood at various locations

Dealing with Overconfidence

$$P(z_t|x_t, m) = \prod_{k=1}^{K} P(z_t^k|x_t, m)$$

- Subsample laser scans: convert 180 beams to 18 beams
- Force the single beam model to be less confident

$$P(z_t^k|x_t, m) \rightarrow P(z_t^k|x_t, m)^\alpha, \alpha < 1$$

MuSHR Localization Project

- Implement kinematic car motion model
- Implement different factors of single-beam sensor model
- Combine motion and sensor model with the Particle Filter algorithm

Lecture Outline

Recap

↓

Bayesian Filtering Examples

↓

Motion Models

↓

Observation Models

Class Outline

State Estimation
- Robotic System Design
- Filtering
- Localization
- SLAM

Control
- Feedback Control
- PID Control
- MPC
- LQR

Planning
- Search
- Heuristic Search
- Motion Planning
- Lazy Search

Learning
- Imitation Learning
- Policy Gradient
- Actor-Critic
- Model-Based RL