

MEAM 520

Lecture 25: Sensing and State Estimation

Cynthia Sung, Ph.D.

Mechanical Engineering & Applied Mechanics

University of Pennsylvania

Final Projects

poll @483

24 views

Actions

Final Project Timing

We've gotten some feedback that the timing of the final project is a bit tight. We'd like to get your thoughts on the best way to schedule the final tournament. Part of the squeeze is due to 12/10 (a Thursday) being a Monday schedule due to the strange academic calendar this semester, so we do not officially have class that day.

Which option do you prefer?

[Option A] Round Robin: 12/3 in class, Top 8 Final Round: 12/8 in class (as written in handout)

[Option B] Round Robin: 12/8 in class, Top 8 Final Round: 12/10 12-1:30 (what would have been class time)

[Option C] Round Robin: 12/8 in class, Top 8 Final Round: some time during reading days

Please answer this poll within the next two days to have your input considered! Thanks!

☐ Option A (stick to the plan)

☐ Option B

☐ Option C

Submit

Final Project

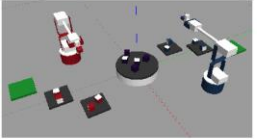
MEAM 520, University of Pennsylvania

November 15, 2020

Teams will use the concepts learned during the semester to control their simulated Lynx robot in a head-to-head competition with their opponent robot. The robots will manipulate objects in the simulated environment to score points, culminating in a class-wide tournament!

Instructions: Just as in lab, this final project is an opportunity for you to explore the concepts we learned in class in a more complicated environment. Equipped on previous labs, pull techniques from the literature, or 22 more experiments of your own. You should document your approach through a report similar to the reports you have written throughout the semester.

The final project is worth 20 pts. Bonus points will be awarded to teams who perform particularly well during the tournament: 5 pts to 1st place, 4 pts to 2nd place, 3 pts to 3rd place.



1 Competition Rules

1.1 Ground Rules

- Students are required to work in teams of four. If you would like, you may also be randomly matched with other students by the teaching staff. Regardless, you must fill out the form on Piazza to either register your team or ask to be matched by November 20 at 12 noon. Any student who does not register their team will be automatically assigned to a team. A few students will likely end up in teams of 3 but this will be sorted out by the teaching staff.
- Teams will submit their code through Gradescope before the competition. During the competition, TA's will run the game on a physical Ubuntu machine (not the provided Virtual Machine) while ensuring the distribution live on Zoom.

1

Updated Schedule

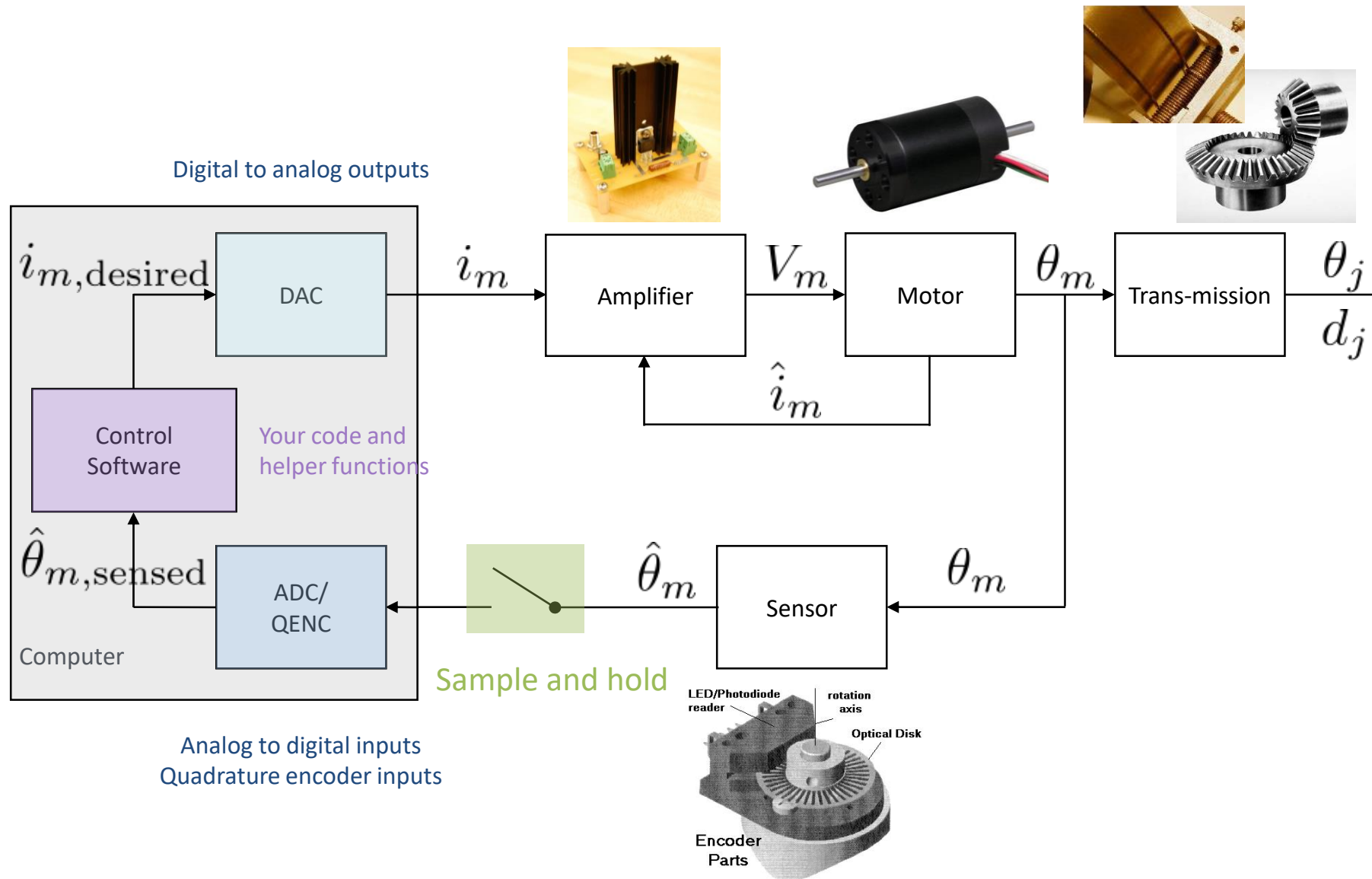
- Bookable 15 min slots on Dec. 3 during class and some other time (will be announced on Piazza)
- Round Robin on Dec. 8 during class and at 8pm ET (schedule based on Piazza post @510)
- Elimination on Dec. 10 during class
- Regular non-bookable OH throughout the weeks

Robotics-Relevant Courses Spring 2021 (Adv Reg 11/30-12/7)

- **MEAM 510:** Design of Mechatronic Systems (Mark Yim)
- **MEAM 513/ESE 505:** Feedback Control (Bruce Kothmann)
- **MEAM 620:** Advanced Robotics (James Paulos, CJ Taylor)
- **ESE 531:** Digital Signal Processing (Tiana Khanna)
- **ESE 547:** Introduction to Legged Locomotion (Daniel Koditschek)
- **ESE 605:** Modern Convex Optimization (Nikolai Matni)
- **ESE 619:** Model Predictive Control (Manfred Morari)
- **ESE 650:** Learning in Robotics (Pratik Chaudhari)
- **CIS 519:** Applied Machine Learning (Dinesh Jayaraman)
- **CIS 520:** Machine Learning (Shivani Agarwal)
- **CIS 580:** Machine Perception (Kostas Daniilidis)
- **CIS 581:** Computer Vision and Computational Photography (Jianbo Shi)
- **IPD 515:** Product Design (Karl Ulrich, Taylor Caputo)
- IPD 545:** Engineering Entrepreneurship I (Jeffrey Babin Vanesa Chan, Thomas Cassel)
- ENM 511:** Foundations of Engineering Math II (Michael Carchidi)
- **ENM 512:** Nonlinear Dynamics & Chaos (Michael Carchidi)
- ENM 520:** Principles & Techniques of Applied Math I (Prashant Purohit)
- **ENM 531:** Data-Driven Modeling (Paris Perdikaris)

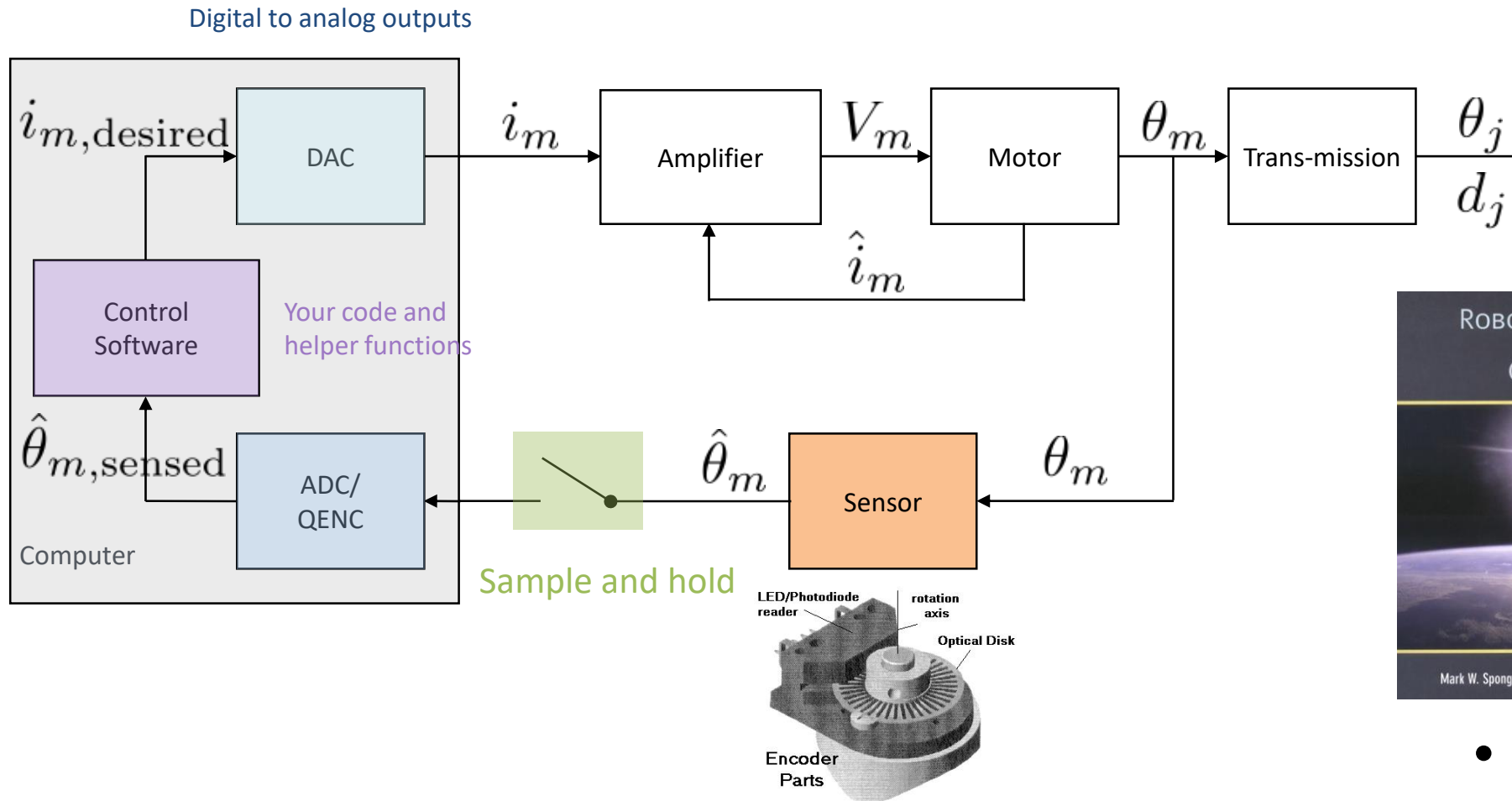
Design
Control
Sensing
Dynamics
Planning
Learning

Previously: How most real robots work



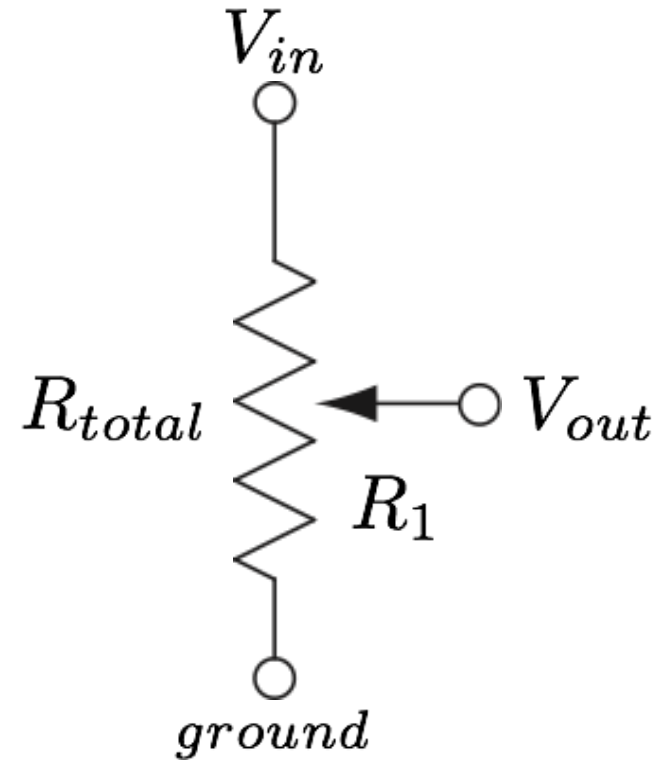
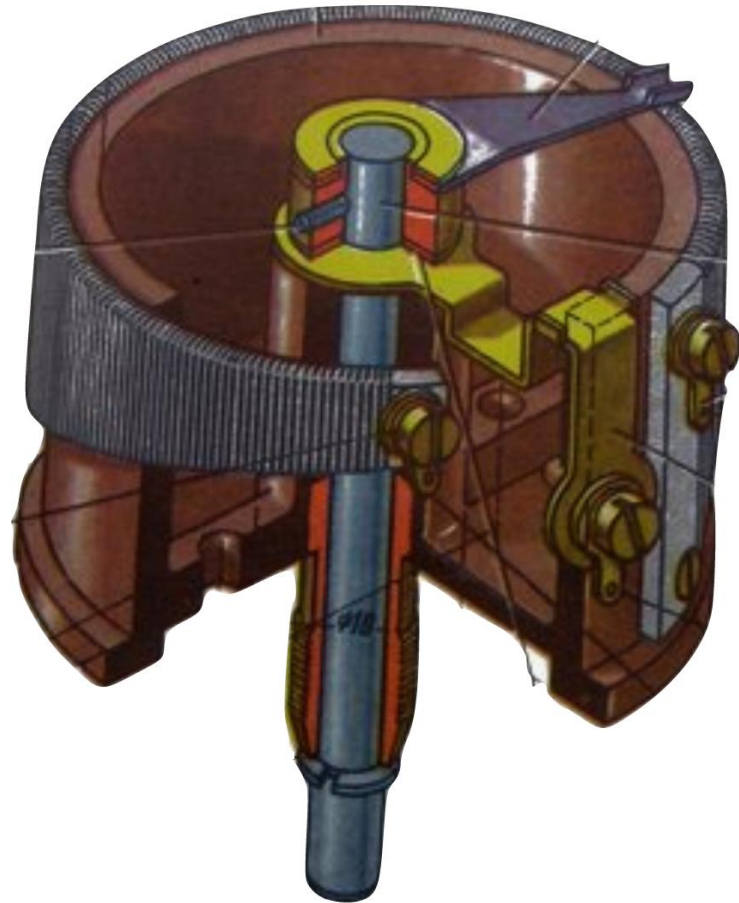
i : current
 V : voltage
 m : motor
 j : joint
 θ : angle
 d : displacement
 $\hat{}$: estimate

Today: Sensing and State Estimation

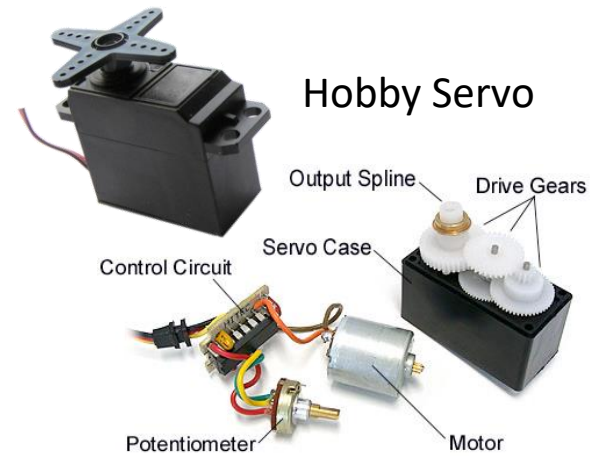


- [AKKK](#): 8.1 – 8.3

Common Sensor: Potentiometer

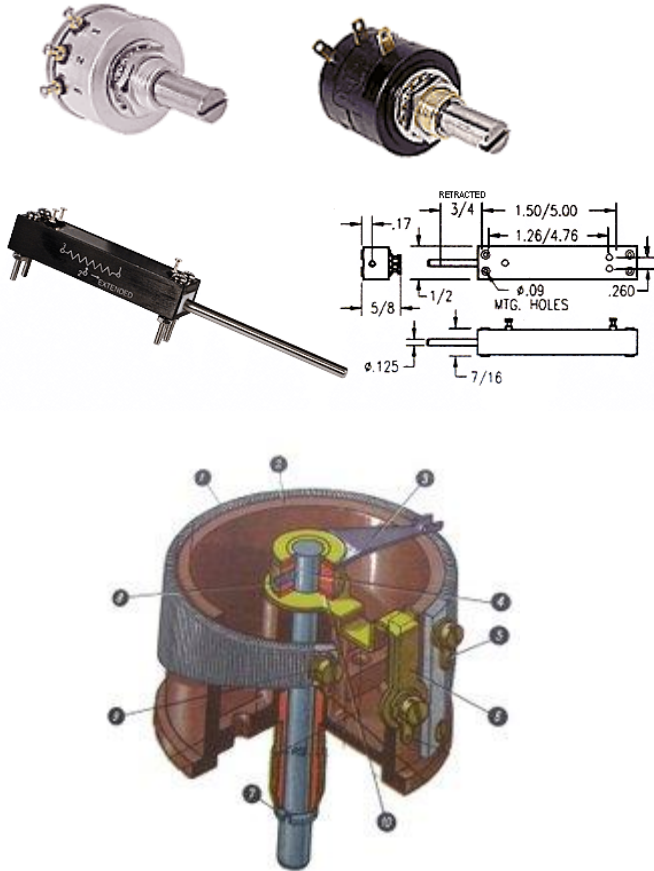


$$V_{out} = \frac{R_1}{R_{total}} V_{in}$$





Potentiometers

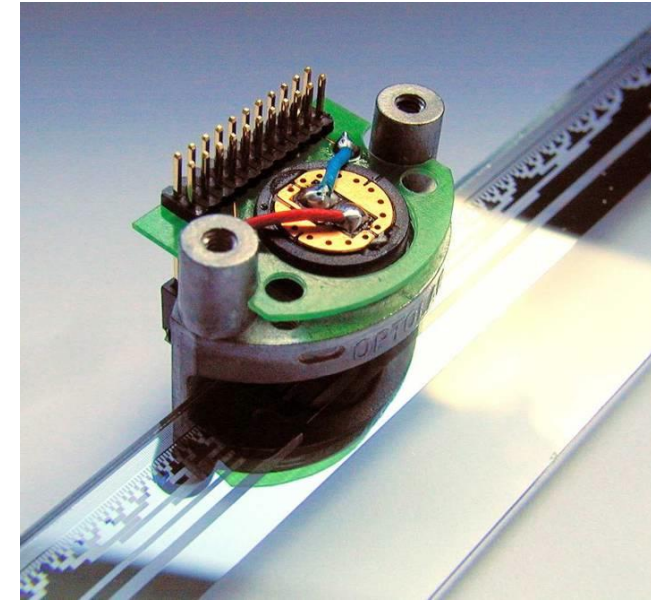
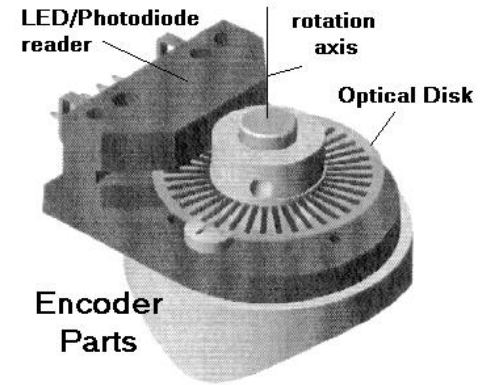


- Typically rotary, but linear exist.
- Cheap and easy.
- Measures absolute position.
- Moving parts means it can wear out.
- Hard to waterproof or dustproof.
- Susceptible to electrical noise from other system elements.
- Has non-negligible friction.
- Usually has finite range.

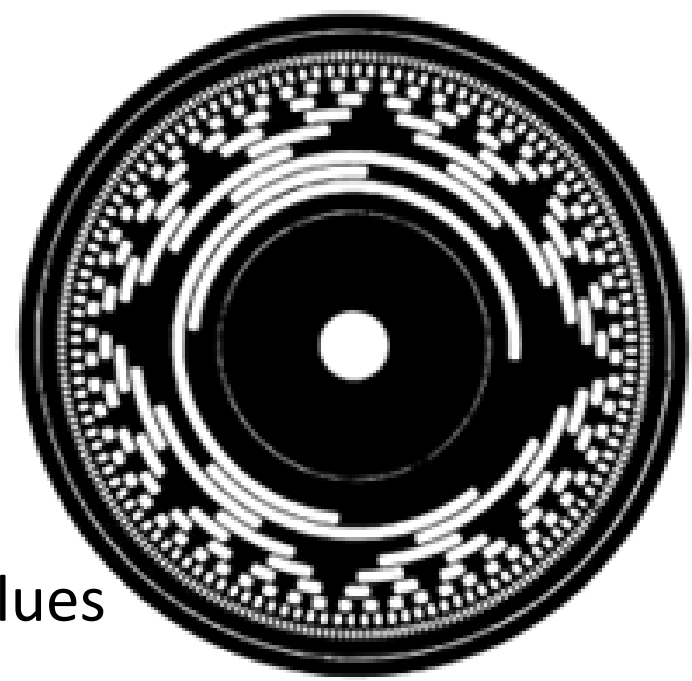
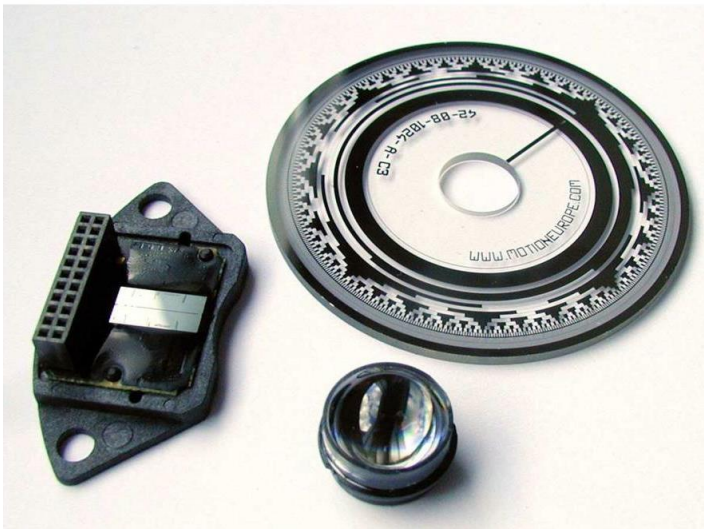
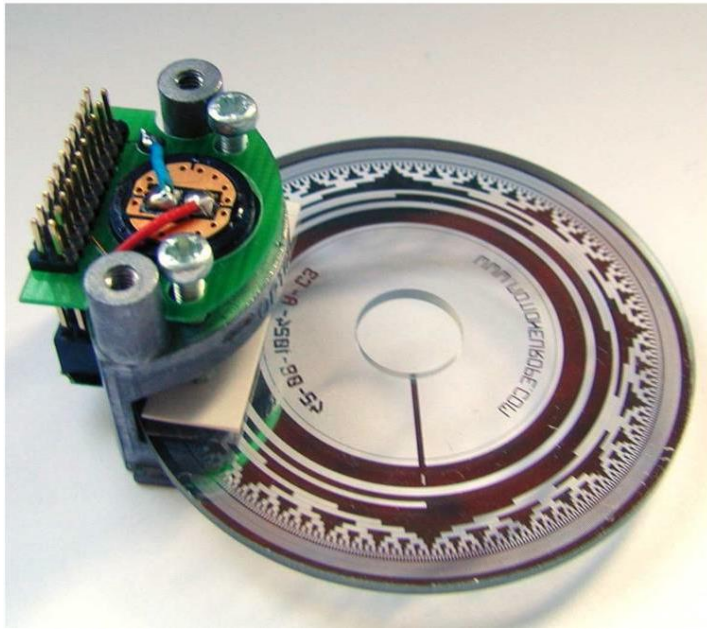
Due to these disadvantages, most robots don't use potentiometers alone.

Encoders

- A thin disk is attached to the rotating shaft whose angle you want to measure, usually the motor.
- The disk has slits cut into it in a regular pattern.
- A light shines on the disk on one side, and photo sensors are located on the opposite side.
- Produces a number of pulses per revolution, with higher resolution being more expensive.



Absolute Encoders



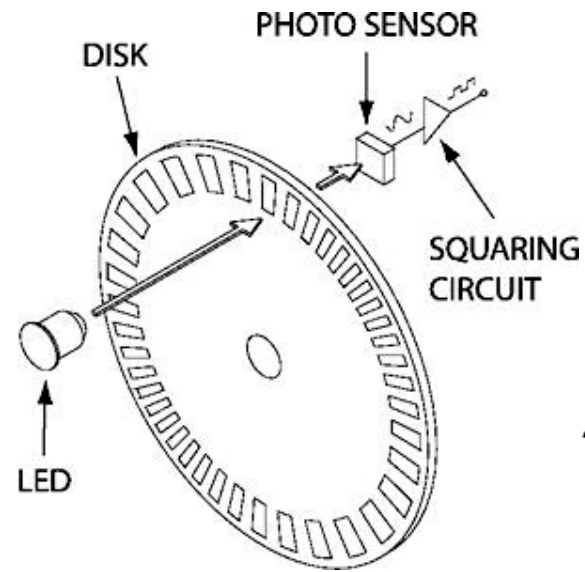
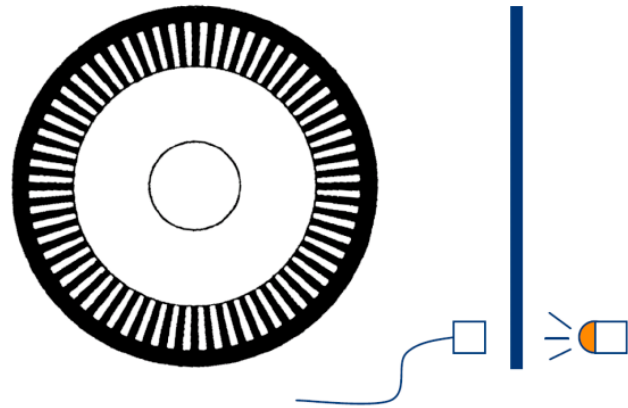
Unique combination of values
for each orientation.

Drawbacks: complexity, cost, wiring

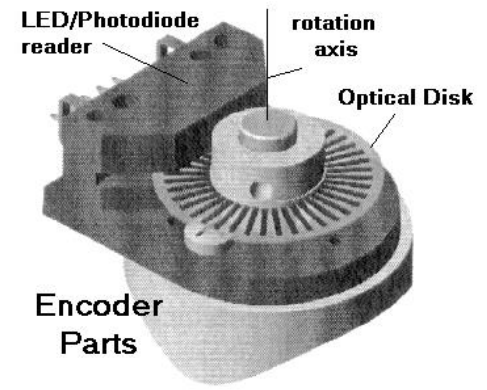
Due to these disadvantages, most
robots don't use absolute encoders.

Instead they use
incremental encoders.

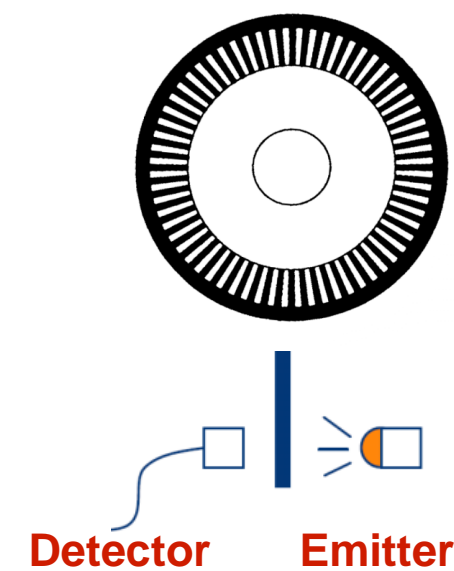
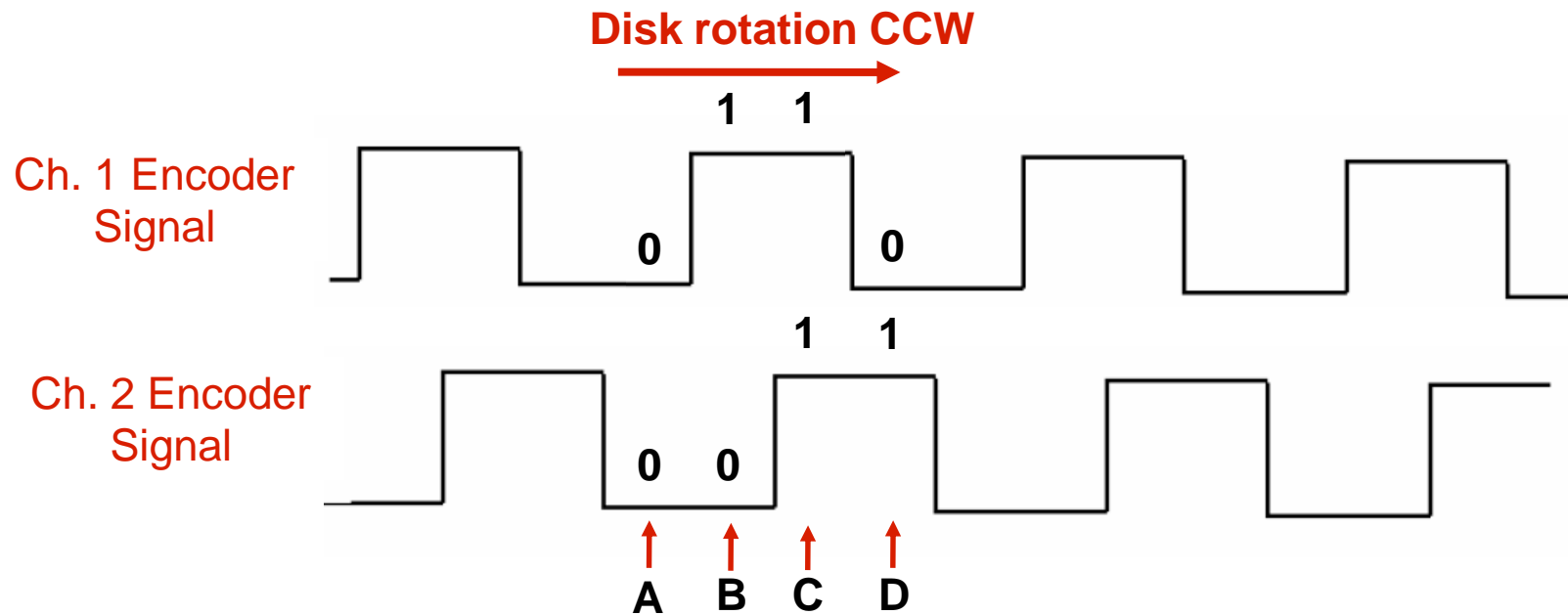
Encoders



$$\Delta = \frac{2\pi}{4n}$$

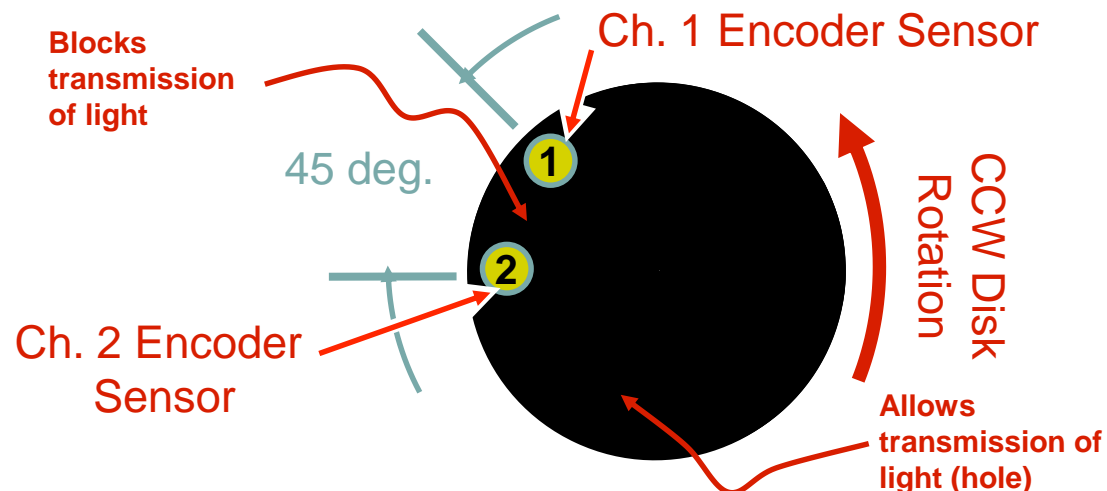


Quadrature Encoder States & Decoding



Simplified Encoder Disk
(2 CPR, 8 PPR) (shown in state A)

Two channels of pulses,
90 degrees out of phase:
quadrature



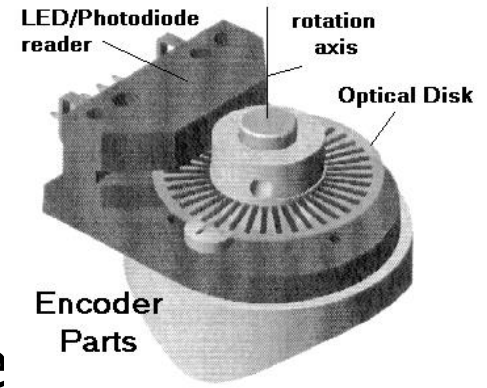
Encoder States

Disk rotation CCW

A	B	C	D
0	1	1	0
0	0	1	1

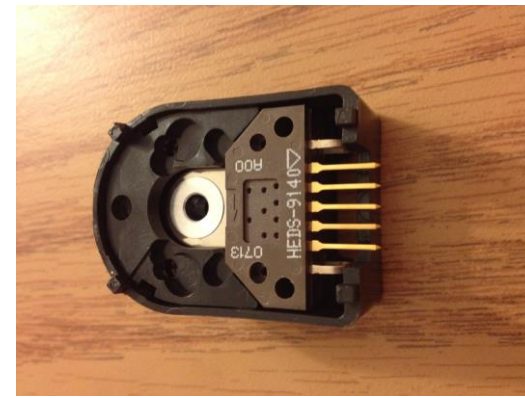
Disk rotation CW

Encoders

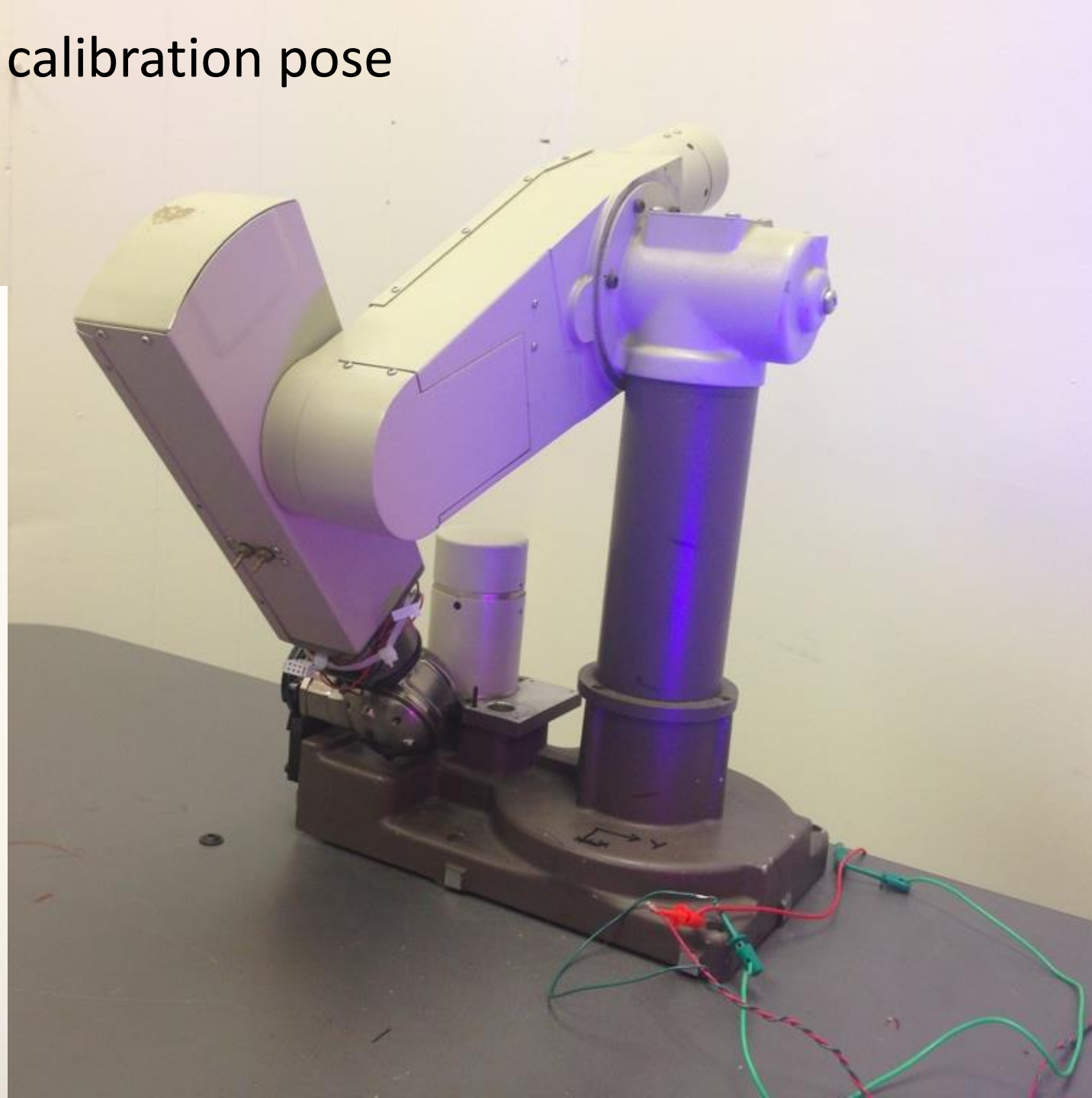


Ramifications of using incremental of optical encode

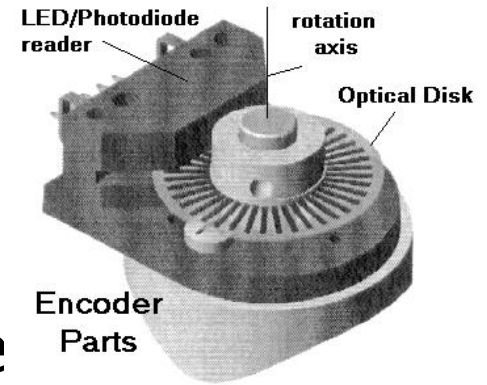
- The system has no knowledge of absolute position, because it's always just counting pulses.
- How can you solve this?
 - Calibration pose



PUMA calibration pose



Encoders



Ramifications of using incremental of optical encode

- The system has no knowledge of absolute position, because it's always just counting pulses.
- How can you solve this?
 - Calibration pose
 - Secondary sensors with absolute readings
- Sometimes problems occur at high velocities.
- No noise on position, but uncertainty due to resolution, and significant noise on velocity.

$$\theta_m = \underset{\substack{\text{resolution} \\ \text{(rad/count)}}}{\Delta} \overset{\substack{\text{measurement} \\ \text{(count)}}}{(Q - \underset{\substack{\text{offset} \\ \text{(count)}}}{Q_{zero}})}$$

Internal sensors tell you about the robot's current state.

External sensors are needed to tell you about the environment
and other interactions.

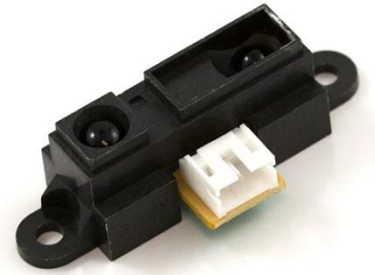
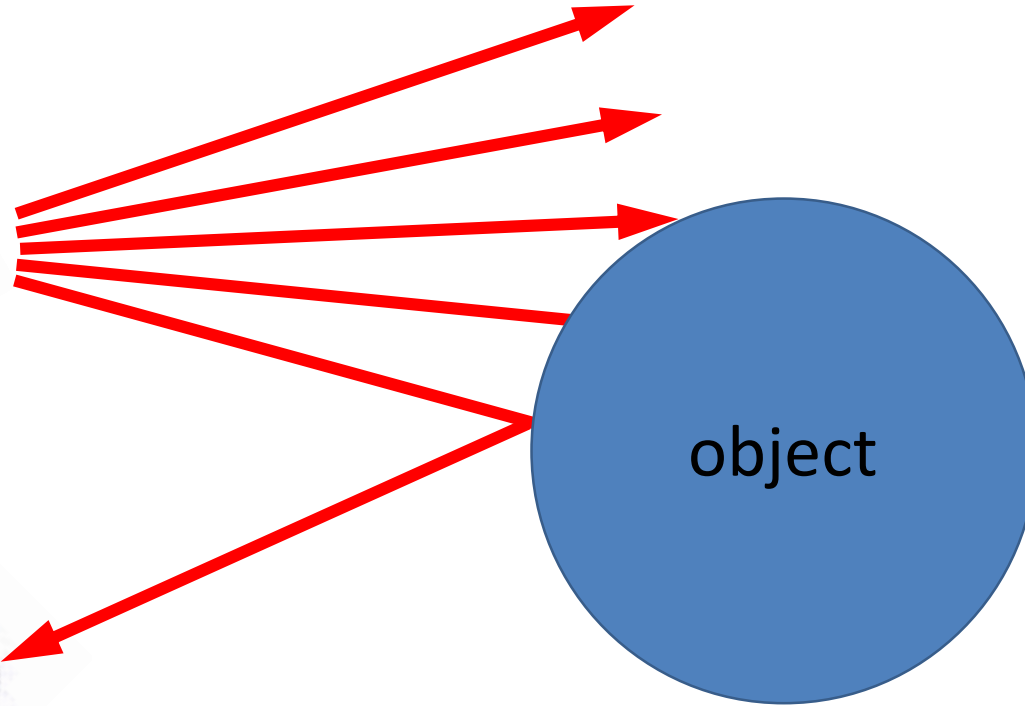
Proximity Sensors



LED (usually IR)



Photodiode



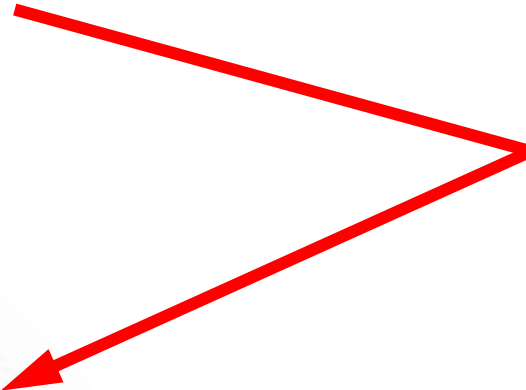
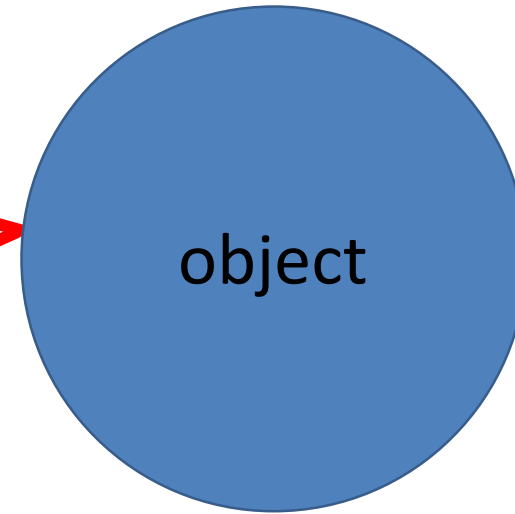
Distance Sensor



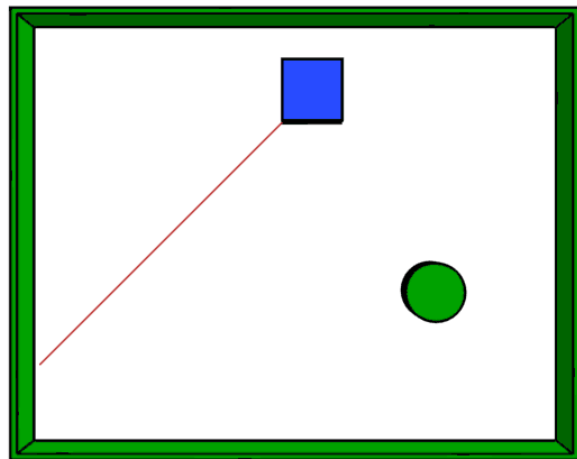
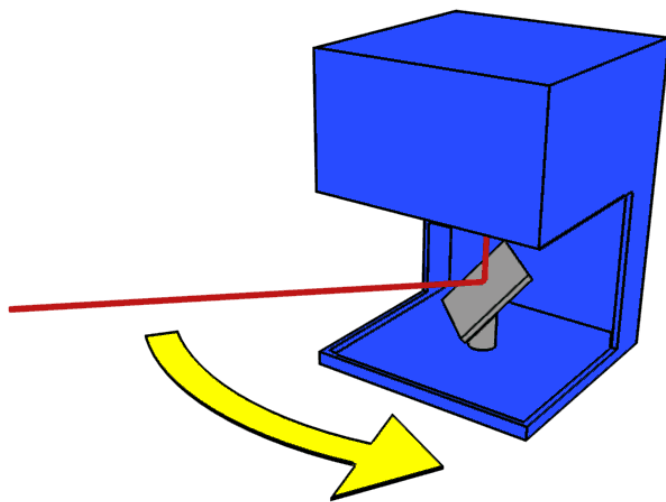
Laser (usually IR)



Photodiode



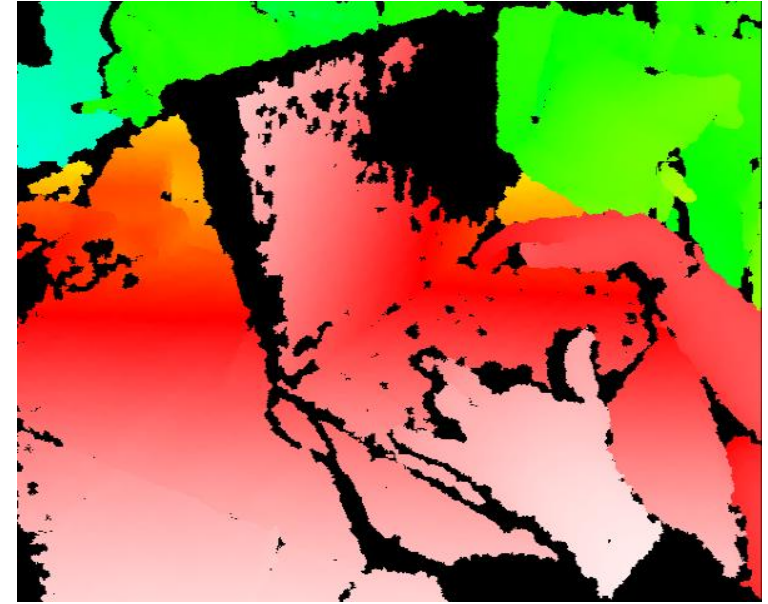
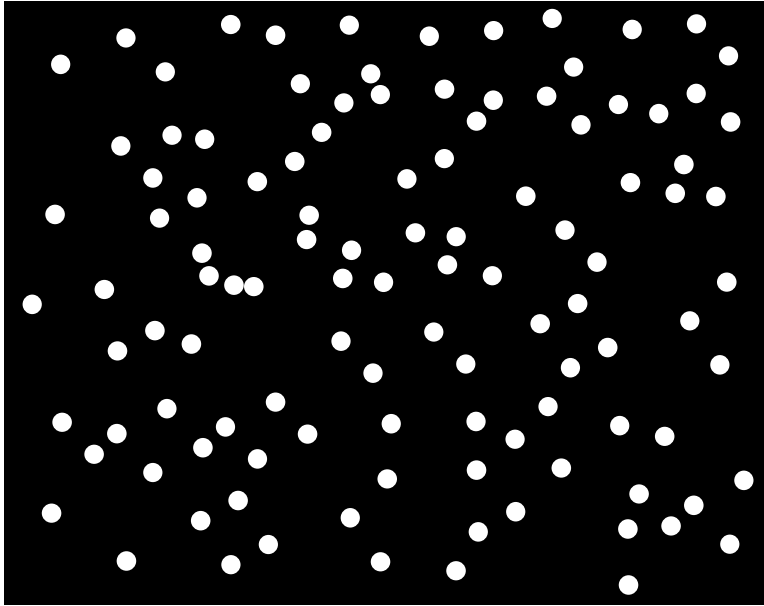
Scanning LIDAR



+
+
+
+
+
+
+
+
+
+
+
+
+
+

•

Depth Camera



Noise and Uncertainty

? question ☆

Lyric position tracking issue

The position tracking on Lyric doesn't seem to be working. My position tracking works perfectly fine in simulation, but is very problematic with Lyric.

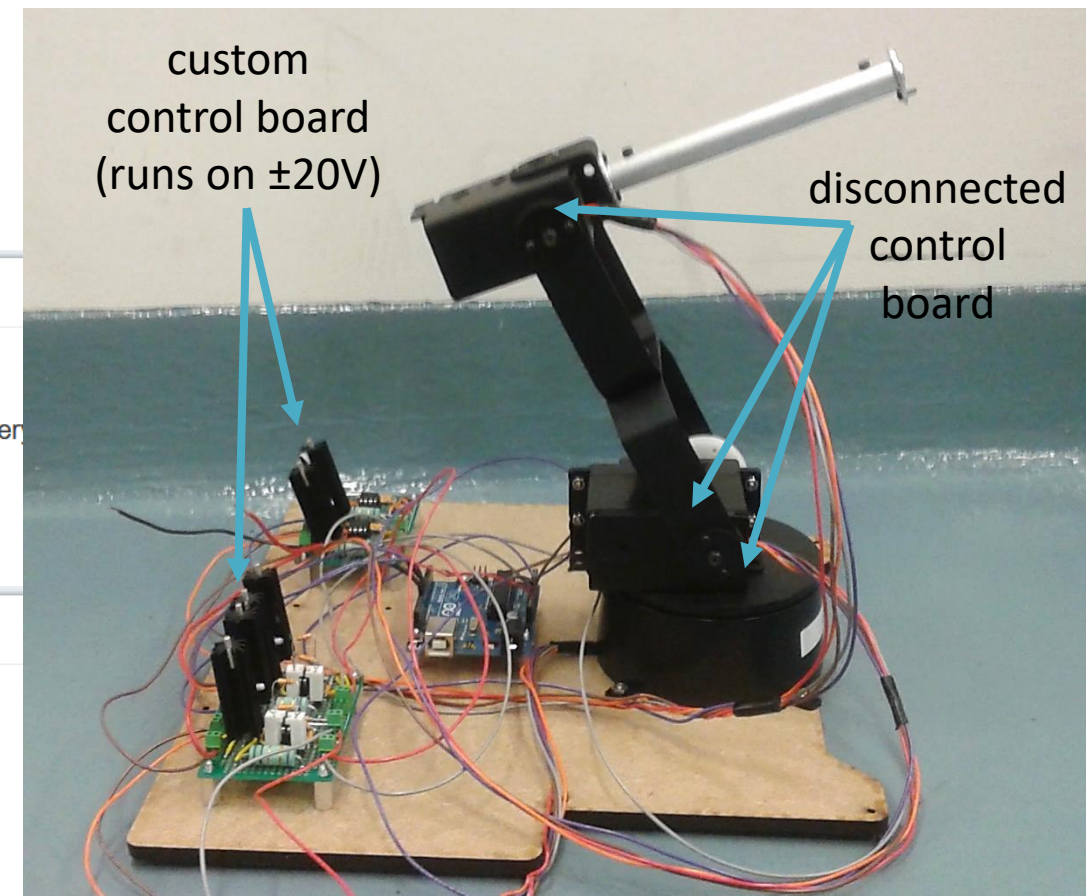
lab5

? question ☆

Lyric bot's sensors awry

Looks like Lyric's joint 1 sensor is non-functional and the other two are misreporting values. What do we do?

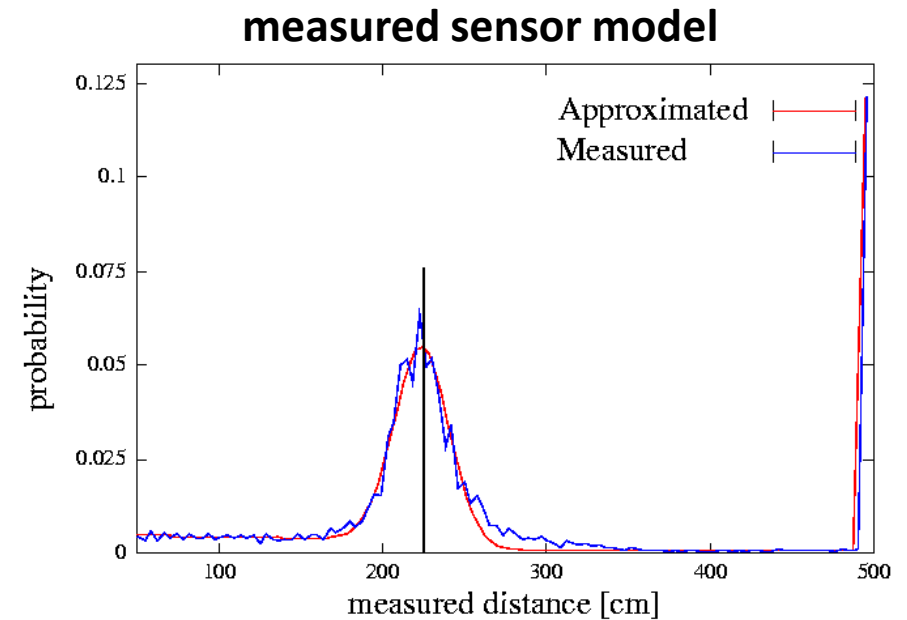
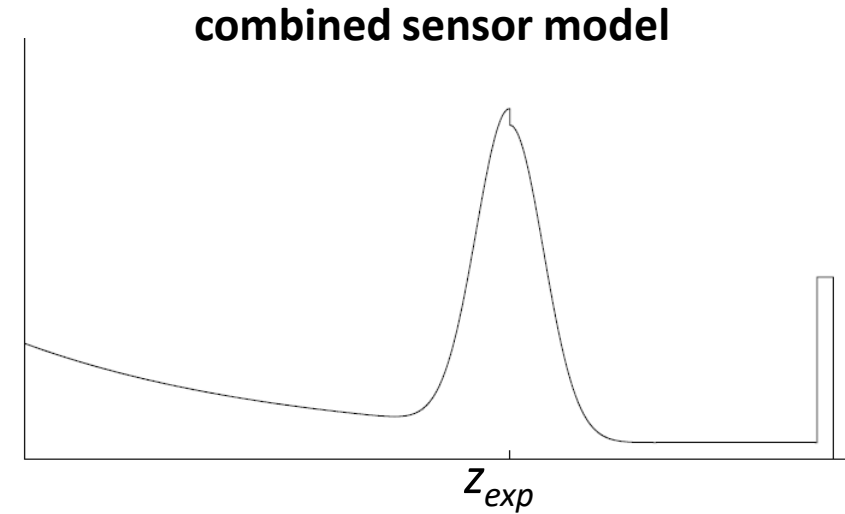
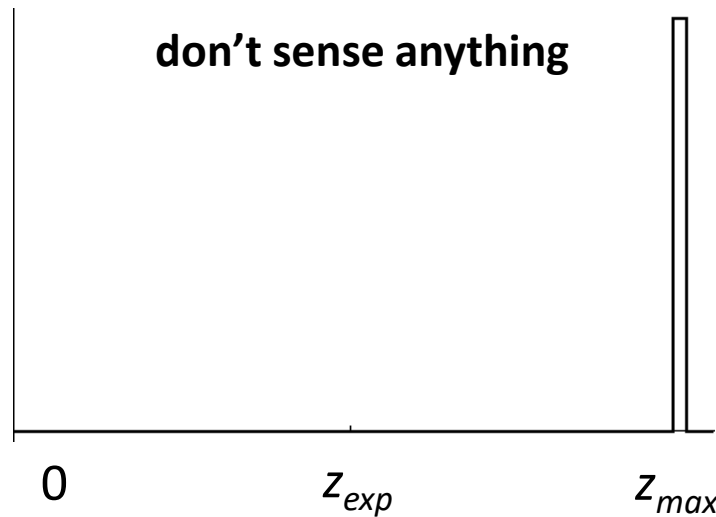
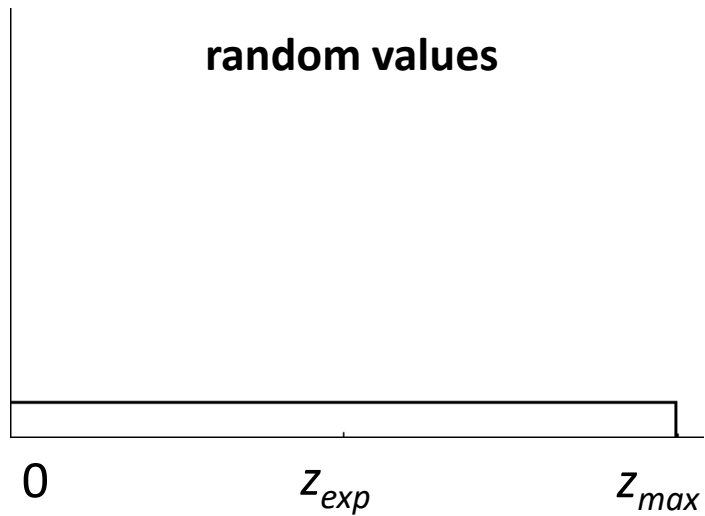
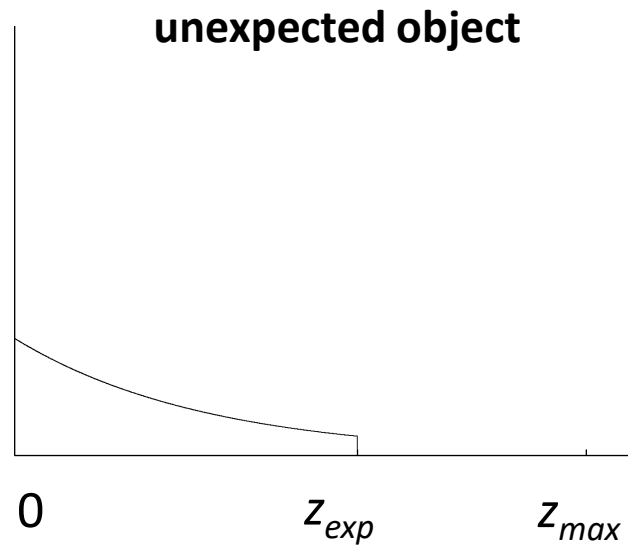
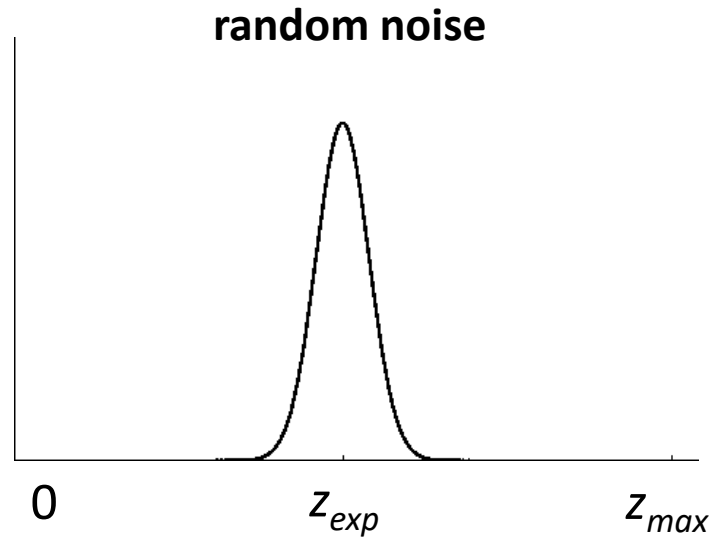
lab5



It can be dangerous to rely on just your sensor measurements.

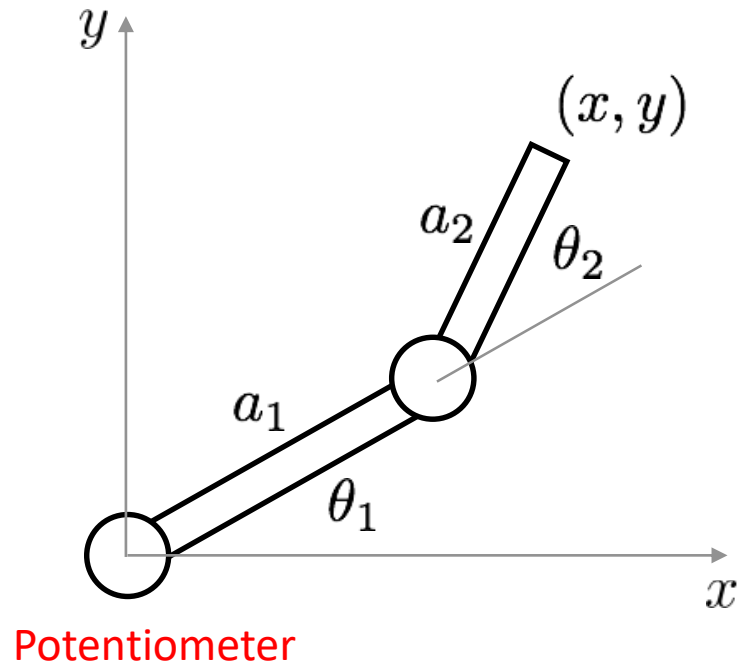
Sensors can malfunction, be uncalibrated, have noisy readings, ...

Sample LIDAR sensor model

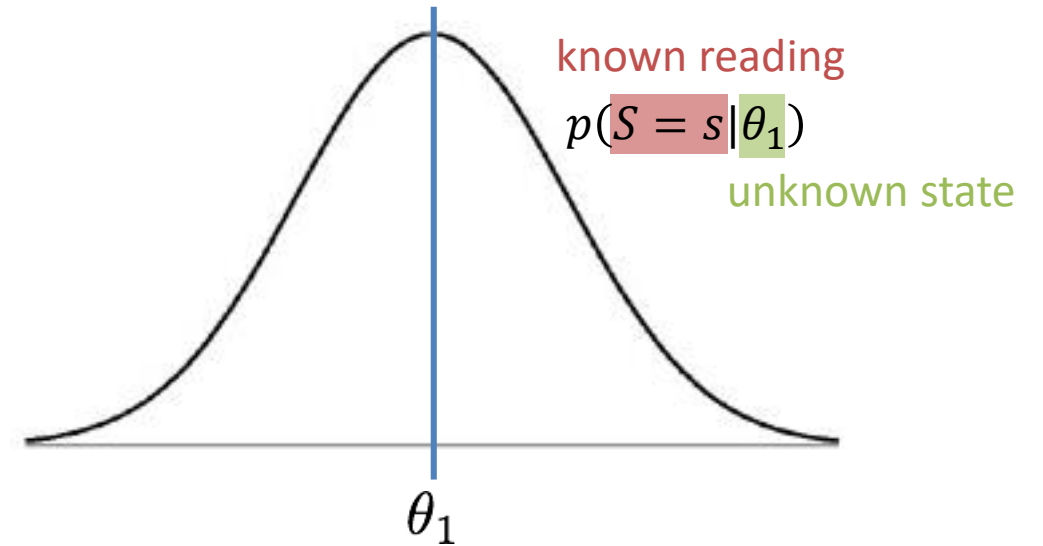


State Estimation

State estimation is the problem of estimating quantities from sensor data

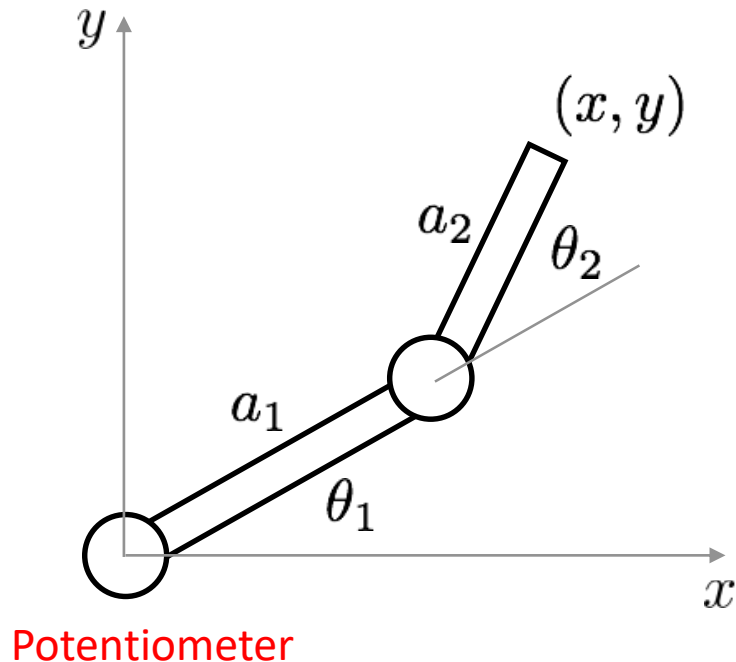


The sensor has some noise distribution



We want to find $p(\theta_1 | S = s)$

Bayes Rule



$$p(\theta_1 | S = s) = \frac{p(S = s | \theta_1) p(\theta_1)}{p(S = s)}$$

$p(S = s \text{ AND } \theta_1)$
independent of θ_1 ,
depends only on
environment/robot

Probabilities must sum to **1**

$$p(\theta_1 | S = s) = \eta p(S = s | \theta_1) p(\theta_1)$$

where the **normalizer** $\eta = (\sum_{\theta_1} p(\theta_1 | S = s))^{-1}$

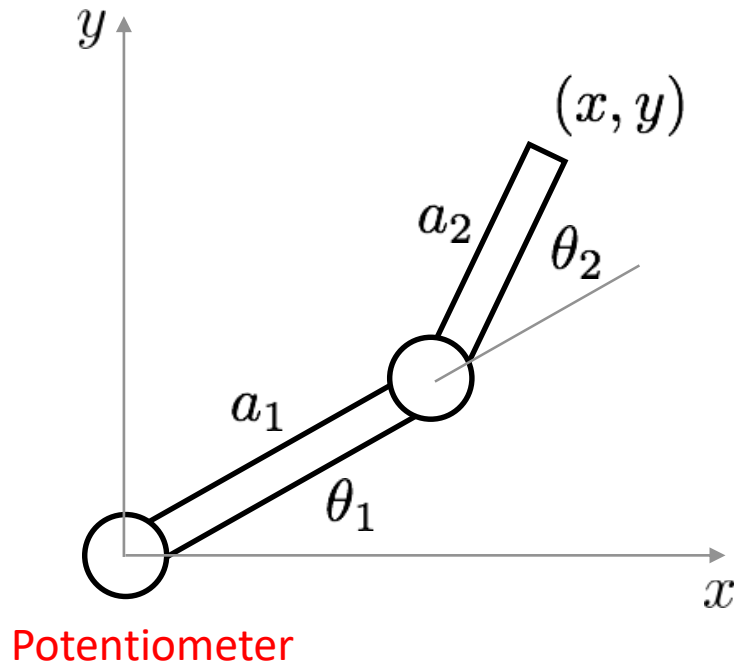
Repeated Measurements

Let's say we have no idea of the starting config:

$$p(\theta_1) \sim U[\theta_{min}, \theta_{max}]$$

Take one sensor reading:

$$p(\theta_1) = \eta p(S = s_1 | \theta_1) p(\theta_1)$$



Take another sensor reading:

$$\eta = \left(\sum_{\theta_1} p(S = s_2 | \theta_1) p(\theta_1) \right)^{-1}$$

$$p(\theta_1) = \eta p(S = s_2 | \theta_1) p(\theta_1)$$

Often, the robot is moving at the same time

Let's say we have no idea of the starting config:

$$p(\theta_1^0) \sim U[\theta_{min}, \theta_{max}]$$

Take one sensor reading:

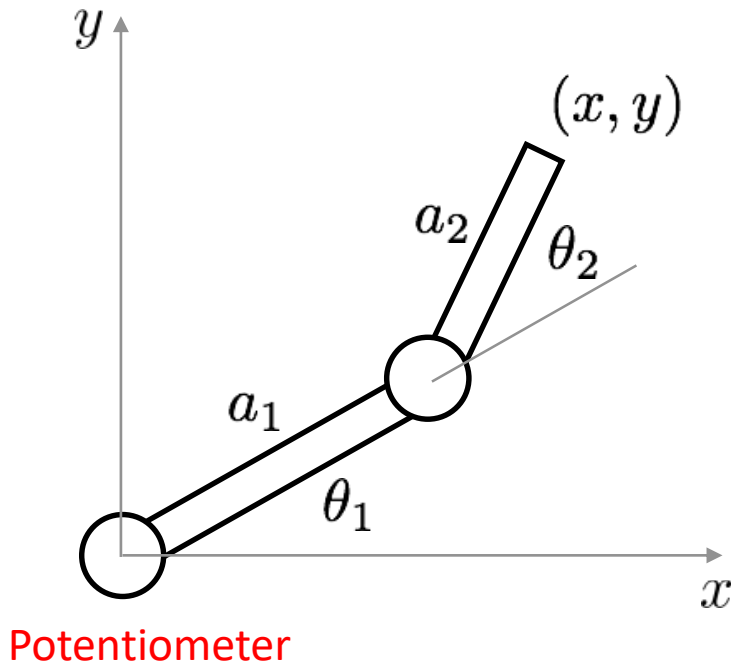
$$p(\theta_1^1) = \eta p(S = s_1 | \theta_1^0) p(\theta_1^0)$$

Move the robot:

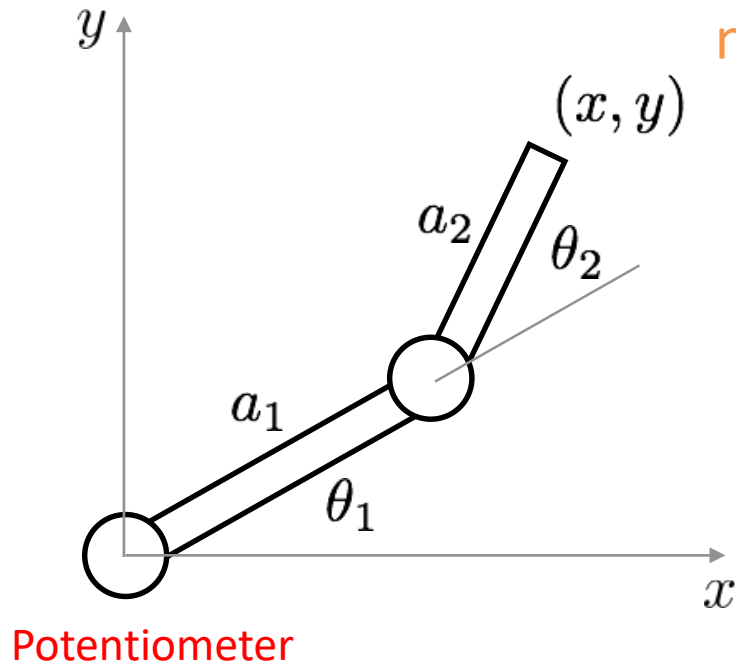
$$p(\theta_1^2) = \sum_{\theta_1^1} p(\theta_1^2 | u_1, \theta_1^1) p(\theta_1^1)$$

Take another sensor reading:

$$\eta = \left(\sum_{\theta_1} p(S = s_2 | \theta_1^2) p(\theta_1^2) \right)^{-1}$$
$$p(\theta_1^2) = \eta p(S = s_2 | \theta_1^2) p(\theta_1^2)$$



Bayes Filter



Pseudocode:

For all q_t configuration at time t

new belief w/o sensing control input belief at time t-1

$$\overline{bel}(q_t) = \int p(q_t | u_t, q_{t-1}) bel(q_{t-1}) dt$$

configuration at time t-1

$$bel(q_t) = \eta p(z_t | q_t) \overline{bel}(q_t)$$

normalize sensing info

end

return $bel(q_t)$
"belief"

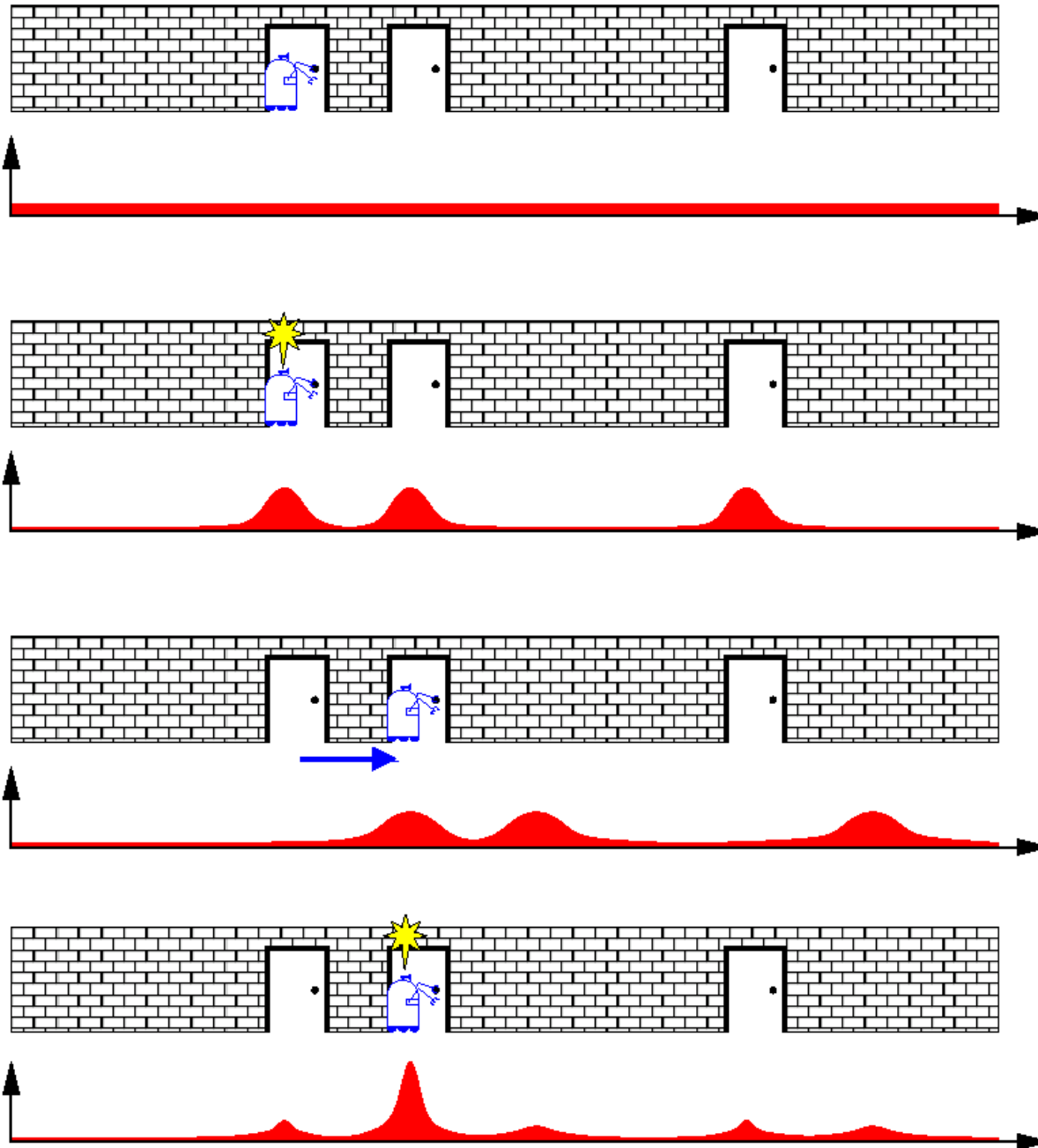
Why do we need to do this when we have encoders?

modeling errors

unknown relationship with environment

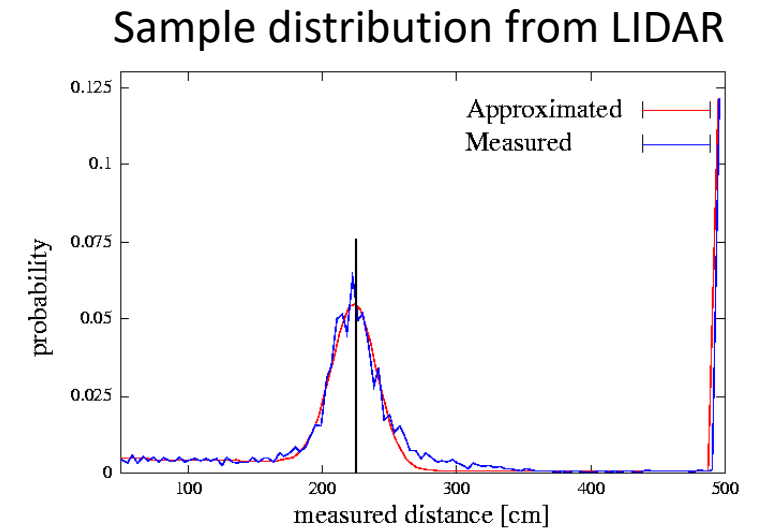
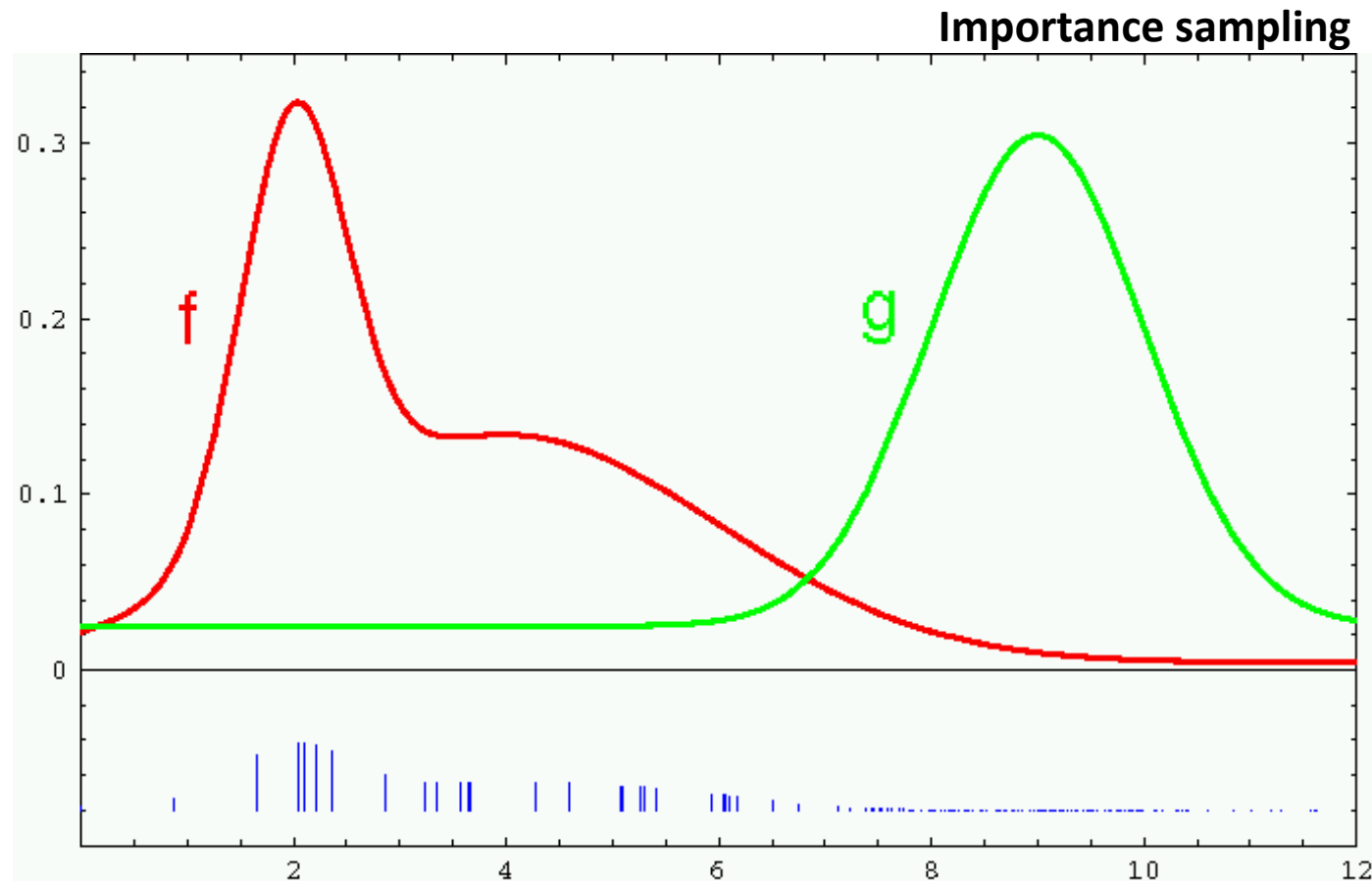
slip





**These integrals are difficult to compute,
especially if the distributions are non-linear.**

Let's try sampling instead.

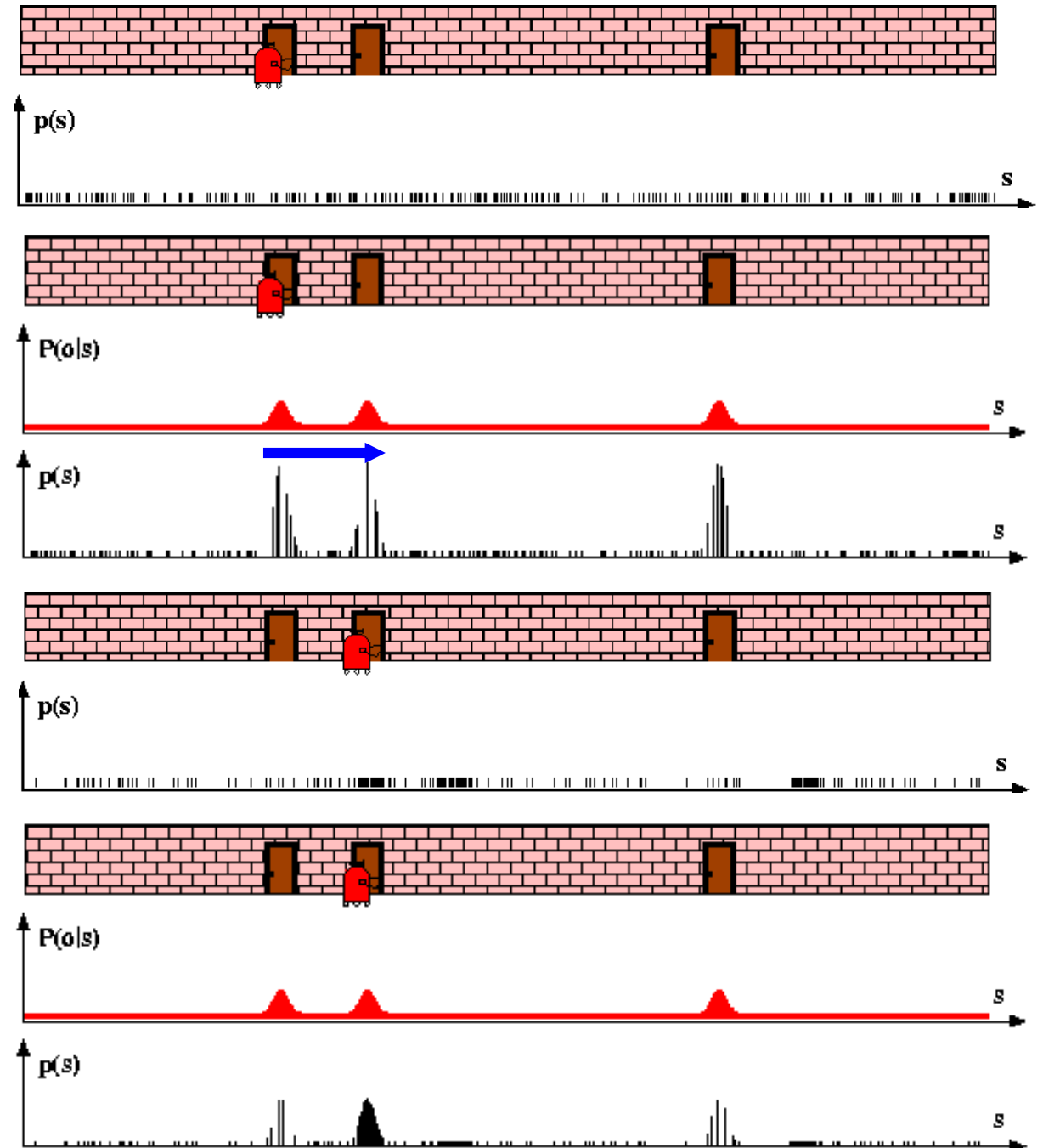


Particle Filters

$$\begin{aligned} Bel(x) &\leftarrow \alpha p(z|x) Bel^-(x) \\ w &\leftarrow \frac{\alpha p(z|x) Bel^-(x)}{Bel^-(x)} = \alpha p(z|x) \end{aligned}$$

$$Bel^-(x) \leftarrow \int p(x|u, x') Bel(x') dx'$$

$$\begin{aligned} Bel(x) &\leftarrow \alpha p(z|x) Bel^-(x) \\ w &\leftarrow \frac{\alpha p(z|x) Bel^-(x)}{Bel^-(x)} = \alpha p(z|x) \end{aligned}$$



Particle Filter

Pseudocode:

While true:

sampled configuration control input

Sample n configurations p_t^i from $p(q_t | q_{t-1}, u_{t-1})$

importance weight

$$w_t^i = p(z_t | q_{t-1}^i)$$

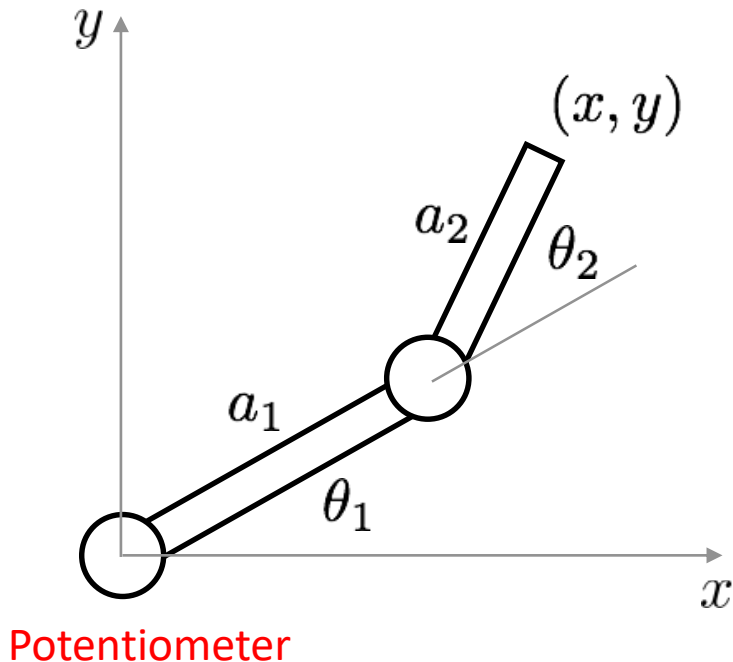
sensing info

$$w_t^i = (w_t^i) / \eta$$

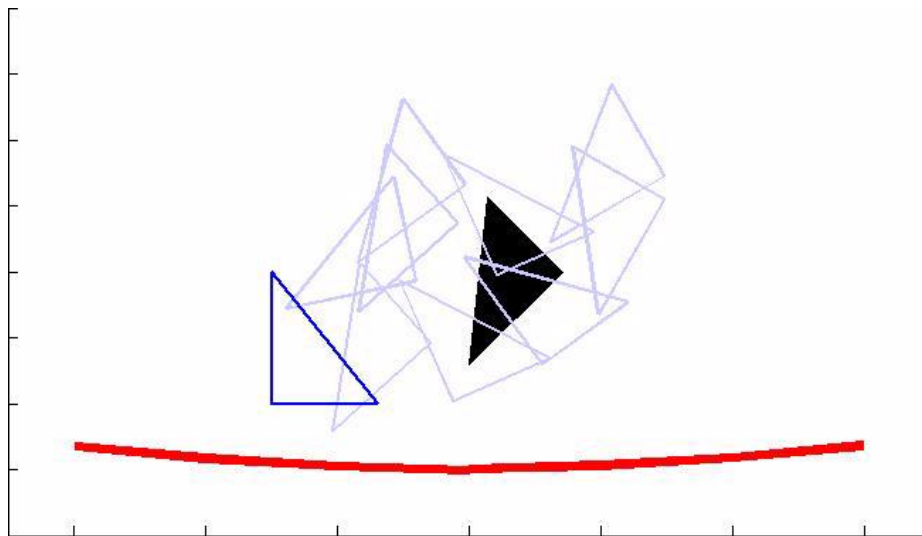
normalize

configuration at time t-1

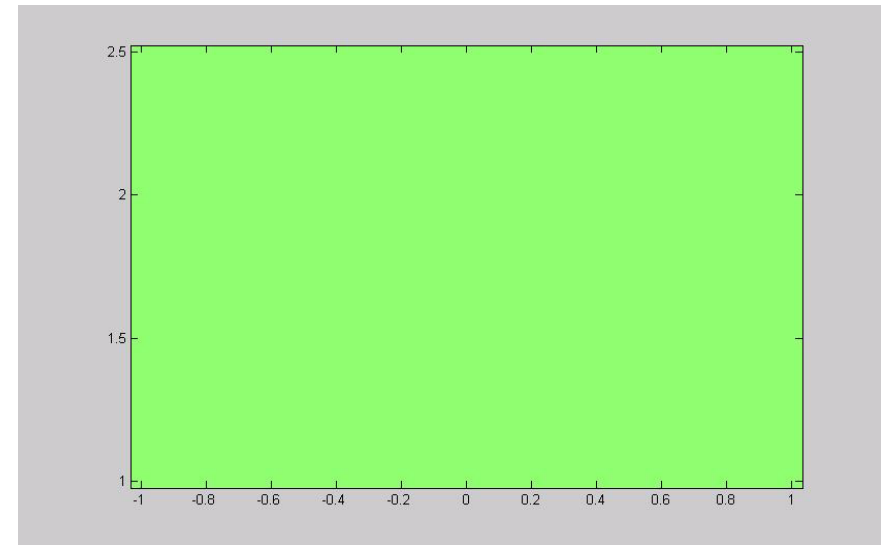
end



Object Localization



$$\theta = \frac{\pi}{4}$$



Particle Filter



D. Fox. Adapting the sample size in particle filters through KLD-sampling. *International Journal of Robotics Research*, 2003.

Next time:

Coordinated multi-arm motion planning: Reaching for moving objects in the face of uncertainty

Seyed Sina Mirzazavi Salehian, Nadia Figueroa and Aude Billard
LASA Laboratory, Ecole Polytechnique Fédérale de Lausanne (EPFL), Switzerland
Email: {sina.mirzazavi, nadia.figueroafernandez, aude.billard@epfl.ch}

Abstract—Coordinated control strategies for multi-robot systems are necessary for tasks that cannot be executed by a single robot. This encompasses tasks where the workspace of the robot is too small or where the load is too heavy for one robot to handle. Using multiple robots makes the task feasible by extending the workspace and/or increase the payload of the overall robotic system. In this paper, we consider two instances of such tasks: a co-worker scenario in which a human hands over a large object to a robot; intercepting a large flying object. The problem is made difficult as the pick-up/intercept motions must take place while the object is in motion and because the object's motion is not deterministic. The challenge is then to adapt the motion of the robotic arms in coordination with one another and with the object. Determining the pick-up/intercept point is done by taking into account the workspace of the multi-arm system. The point is continuously recomputed to adapt to change in the object's trajectory. We propose a virtual object based dynamical systems (DS) control law to generate autonomous and synchronized motions for a multi-arm robot system. We show theoretically that the multi-arm + virtual object system converges asymptotically to the moving object. We validate our approach on a dual-arm robotic system and demonstrate that it can re-synchronize and adapt the motion of each arm in a fraction of a second, even when the motion of the object is fast and not accurately predictable.

I. INTRODUCTION

Many daily activities involve tasks such as lifting, carrying and reaching for large and heavy objects. To accomplish these tasks, humans rely heavily on bi-manual reaching and, more generally, on coordination between the hands [6]. Performing these tasks with one arm is often infeasible, mainly because a single arm has a limited workspace. Moreover, the dexterity and flexibility required for such task is beyond a single arm's capabilities. This holds for robotic systems as well. A single robot arm is simply incapable of meeting the requirements of such complex tasks. A dual or multi-arm robotic system, on the other hand, extends the workspace of a single robot arm. It allows for highly complex manipulation of heavy or large objects that would otherwise be infeasible for single-arm systems. Most effort in the field of multi-arm control has focused primarily on devising strategies for coordinated and stable manipulation of *static* objects that are already partially or fully grasped by the multi-arm system [1, 23, 5, 21, 33, 7]. Seldom work has focused on developing *reaching strategies* that a multi-arm system can use to reach and grab *moving* objects.

One can envision a plethora of applications in factories, airports, or storage facilities that would benefit from such

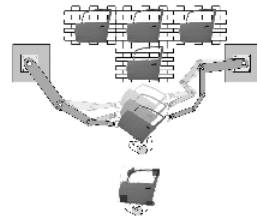


Fig. 1: Illustration of a possible application where a multi-arm system has to reach for large moving objects carried to them by humans. The arms move in synchrony towards the coupled feasible reaching points of the object (e.g. squares or ellipses). As the object approaches, the arms' end-effectors align their trajectories with that of the object.

strategies. Examples include, grabbing, catching, carrying, lifting or re-orienting packages or large-sized parts traveling on a cart or a running conveyor or belt (Fig. 1), carried by humans or even flying towards the multi-arm robot system. We posit that, until now, these applications have not yet been explored, due to the unsuitability of the state-of-the-art to tackle the challenges of coordinating the motion of multiple arms while reaching and adapting to the motion of a dynamic object in a computationally efficient way.

According to studies in motor and cognitive development, reaching for an object in a smooth and efficient manner requires ones to deal simultaneously with different issues [27]. Each hand has to adjust to the orientation, shape and size of the object while reaching for it. Moreover, the action of grasping the object (i.e. closing the fingers of the hands) must be timed prior to rather than as a reaction to intercepting the object. Hence, bi-manual, and by extension, multi-manual reaching require us to solve simultaneously several spatial and temporal coordination constraints to move toward the object in coordination and to intercept the object [27].

We propose an approach that generates coordinated trajectories for a multi-arm robot system that ensures that the arms will reach the moving object simultaneously. Most importantly, the approach updates the arms' motion continuously and in synchrony to adapt to changes in the target object's trajectory. To validate the approach, we consider a scenario in which

Final Project

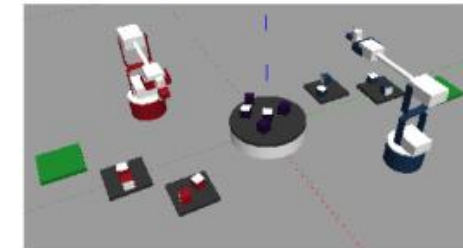
MEAM 520, University of Pennsylvania

November 13, 2020

Teams will use the concepts learned during the semester to control their simulated Lynx robot in a head-to-head competition with their opponents' robot. The robots will manipulate objects in the simulated environment to score points, culminating in a class-wide tournament!

Instructions: Just as in lab, this final project is an opportunity for you to explore the concepts we learned in class in a more complicated environment. Expand on previous labs, pull techniques from the literature, or try some experimentation of your own. You should document your approach through a report similar to the reports you have written throughout the semester.

The final project is worth 70 pts. Bonus points will be awarded to teams who perform particularly well during the tournament: 5 pts to 1st place, 3 pts to 2nd place, 1 pt to 3rd place.



1 Competition Rules

1.1 Ground Rules

- Students are required to work in teams of four. If you would like, you may also be randomly matched with other students by the teaching staff. Regardless, you must fill out the form on Piazza to either register your team or ask to be matched by November 20 @ 12 noon. Any student who does not register their team will be automatically assigned to a team. A few students will likely end up in teams of 3 but this will be sorted out by the teaching staff.
- Teams will submit their code through Gradescope before the competition. During the competition, TA's will run the game on a physical Ubuntu machine (not the provided Virtual Machine) while streaming the simulation live on Zoom.

(RSS 2016)

Final projects