Principles of Robot Autonomy I

CS 237A / AA 274A / EE 260A / ME 274A

Course overview, Logistics,

Environment, Robot, and Agent Representations

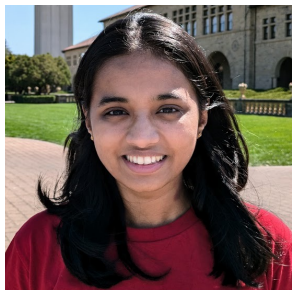
Team

Instructor:

Prof. Mac Schwager

Course Assistants:

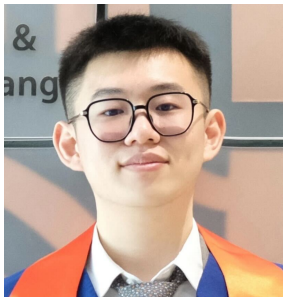
Rhea Malhorta



Aarya Sumuk

Purush Mani

Jingyun Yang



Naixiang Gao

Esteban Rincon

Polo Contreras

John Tucker

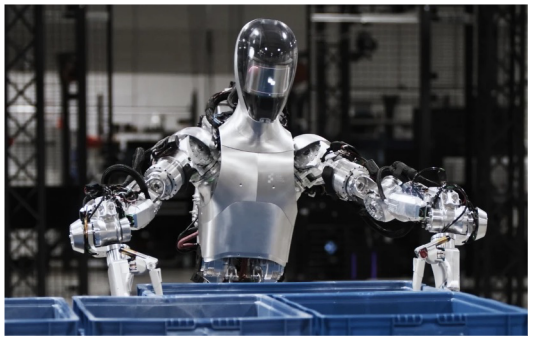
Email questions to: `cs237a-aut2526-staff@lists.stanford.edu`

Robot Autonomy

Waymo Self-Driving Car

Astrobee - NASA

Figure AI humanoid



Mobile Aloha

Course goals

To learn the *theoretical*, *algorithmic*, and *implementation* aspects of main techniques for robot autonomy:

1. Gain a fundamental knowledge of the “autonomy stack”
2. Be able to apply such knowledge in applications and research using ROS2
3. Devise novel methods and algorithms for robot autonomy

Main focus: *robot navigation,* not *robot manipulation*

Other robotics resources at Stanford

Manipulation Courses:

CS223A: Intro to robotics (Khatib)

CS225A: Experimental Robotics (Khatib)

CS327A: Advanced Manipulation (Khatib)

CS237B: Robot Autonomy II (Bohg, Sadigh, Pavone)

CS326: Advanced robotic manipulation (Bohg)

CS227: Robot perception (Song)

ME314: Robotic Dexterity (Kennedy)

Navigation Courses (including Localization, Mapping, SLAM):

CS237B: Robot Autonomy II (Bohg, Sadigh, Pavone)

AA203: Optimal and Learning-based Control (Pavone)

AA212: Advanced Feedback Control (Schwager)

AA275: Navigation for Autonomous Systems (Gao)

AA273: Filtering and Estimation for Robot Perception (Schwager)

CS 224R: Deep RL (Finn)

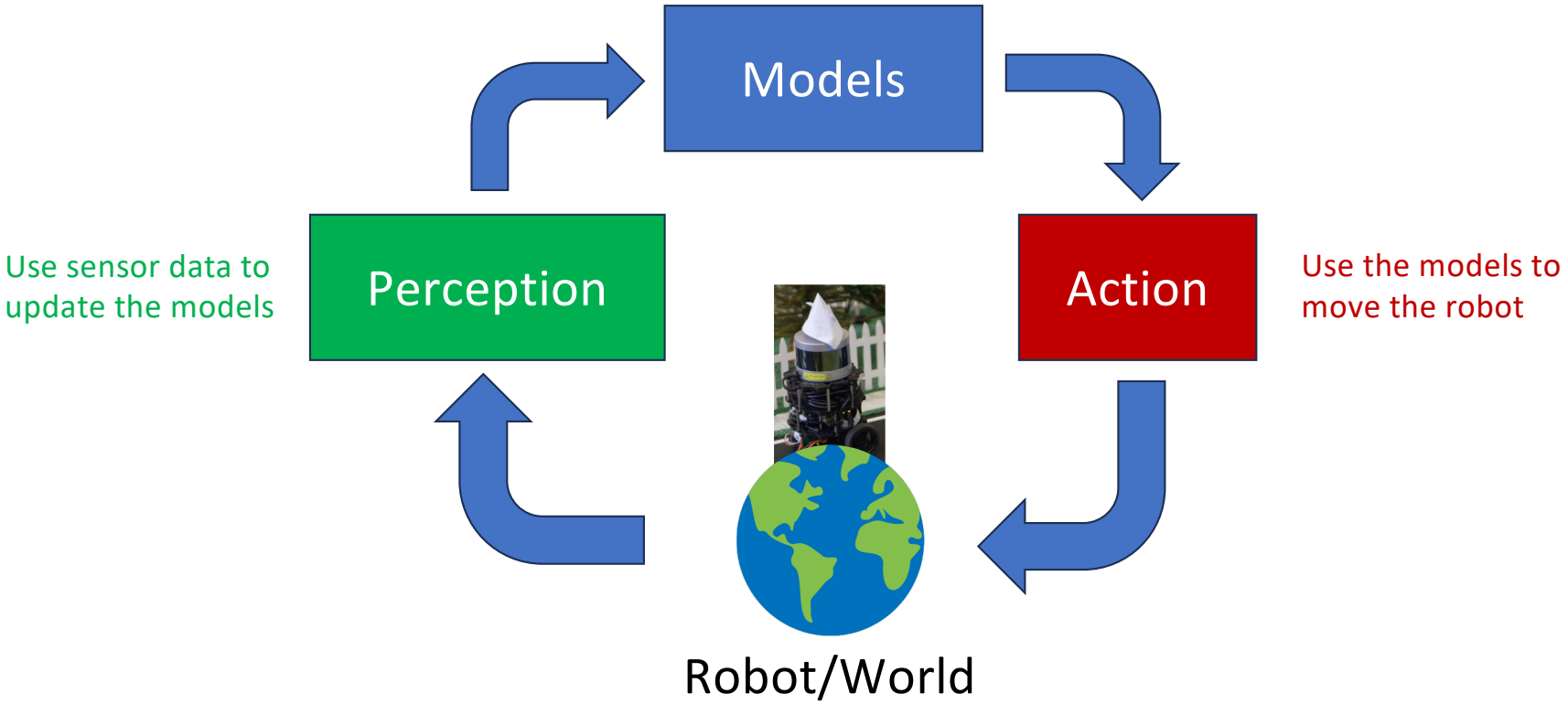
ME 326: Collab Robotics (Kennedy)

CS 345: Building AI-Enabled Robots (Liu)

AA 276: Principles of Safety-Critical Autonomy (Bansal)

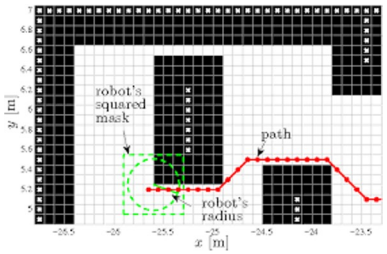
Stanford Robotics Center (SRC): https://src.stanford.edu/

Full Stack Autonomy: the Perception-Action Loop

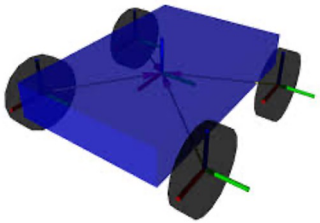


Models: Environment, Robot, Agent Representations

Maps:

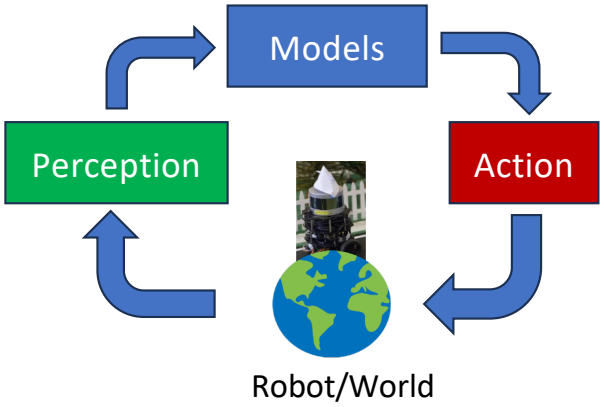


Robot models or simulators:



Agent models or simulators:

Maps:
- Location and geometry of obstacles
- Free vs occupied space
- Global coordinate frame
- Uncertainty quantification

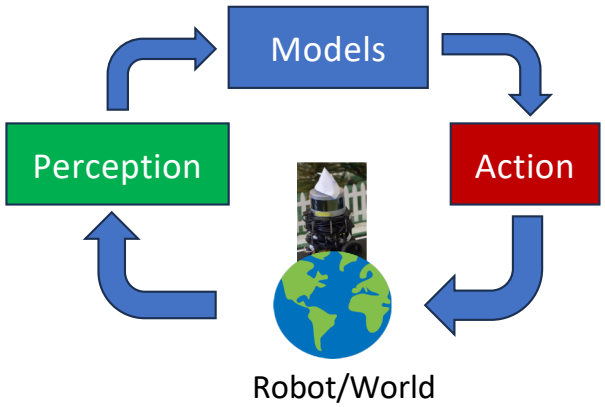


Robot models:
- Kinematics/dynamics
- Geometry
- Rigid body physics simulators

Object/Agent Models:
- Motion prediction models (how will it move)?
- Geometry models
- Uncertainty quantification

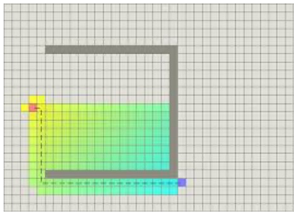
Action: Planning and Control Stack

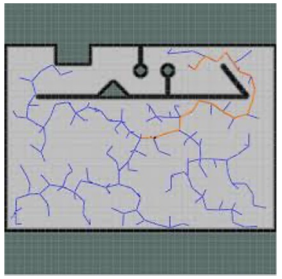
Use the models to move the robot



Path planning:

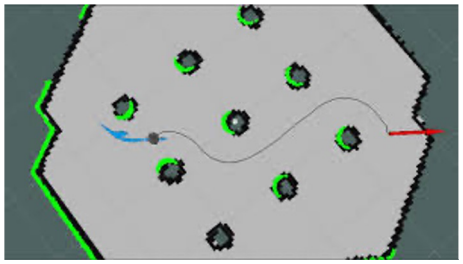
- A* planning on a graph
- RRT, RRT*





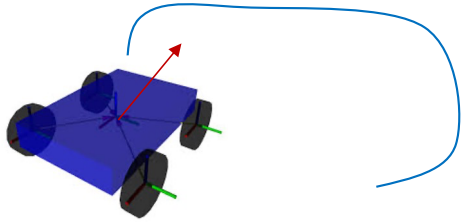
Trajectory optimization:

- Optimal control
- Numerical solvers
- **AA203: Opt and Learning Cont (Pavone)**



Trajectory following:

- Kinematic/dynamic models
- PID control
- LQR, gain scheduled LQR
- **AA212: Adv. Feedback Cont (Schwager)**

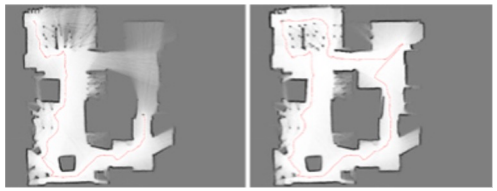


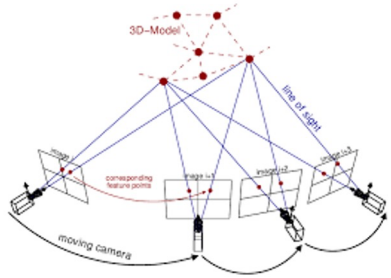
Perception Stack: Computer vision, filtering, SLAM

Use sensor data to update the models

Localization, Mapping, Tracking:

- EKF/Monte Carlo localization
- Occupancy grid mapping
- Pose graph optimization
- Tracking (EKF and Particle Filter)
- **AA273: Filtering (Schwager)**
- **AA275: Navigation (Gao)**



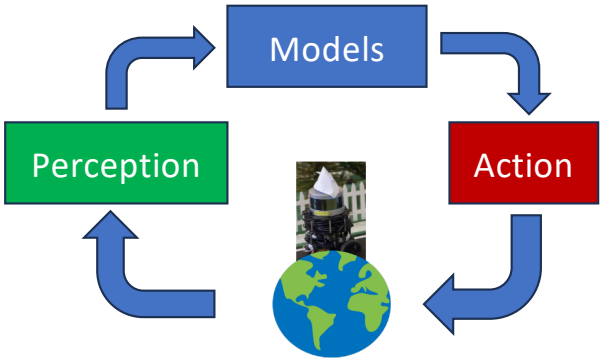


Information extraction

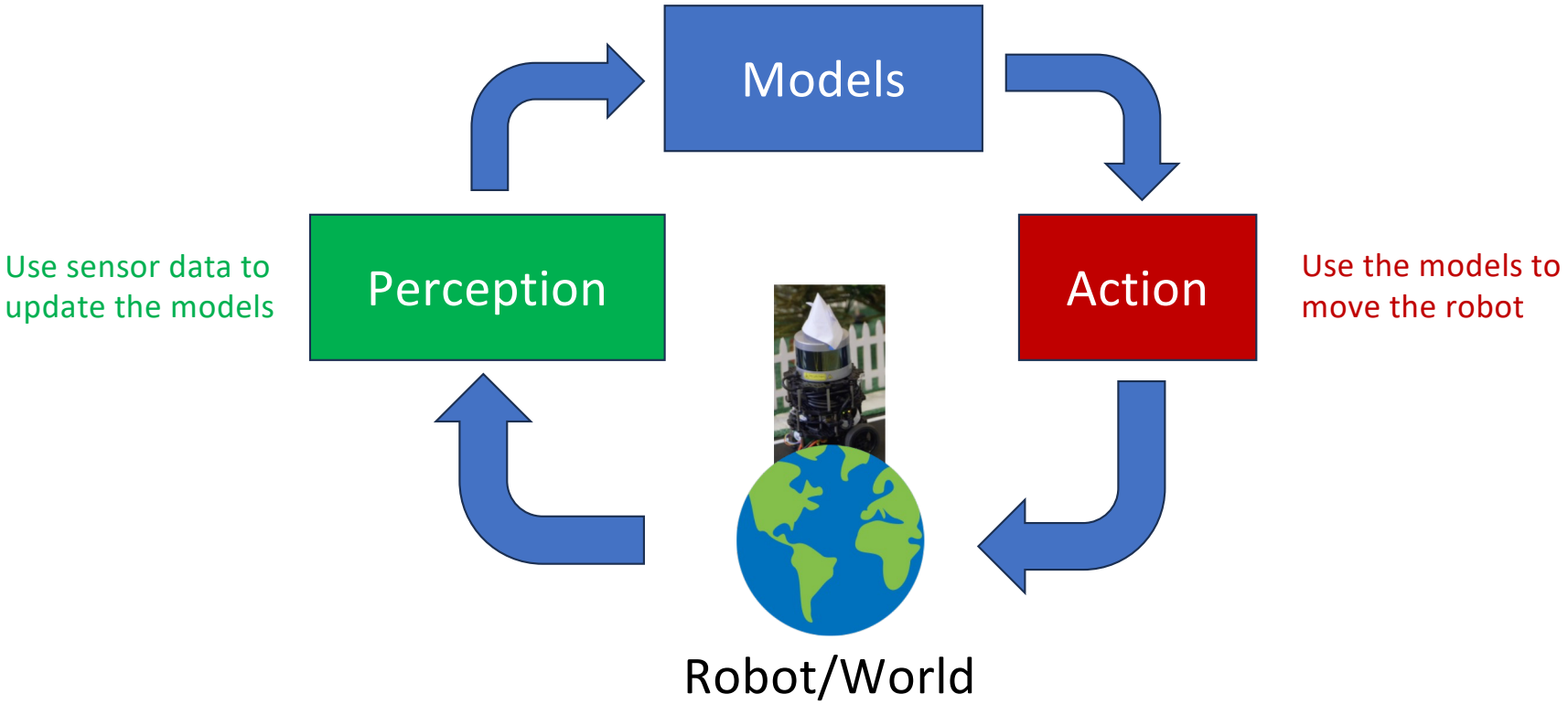
- Computer vision: features, correspondences, Structure from Motion (SfM), depth
- Lidar scan matching, ICP
- **CS231A: Comp Vision**

Sensors:

- RGB Cameras, RGB-D/stereo cameras, Lidar
- IMU, GPS, wheel encoders



Full Stack Autonomy: the Perception-Action Loop



But don’t robots use ML/AI?!

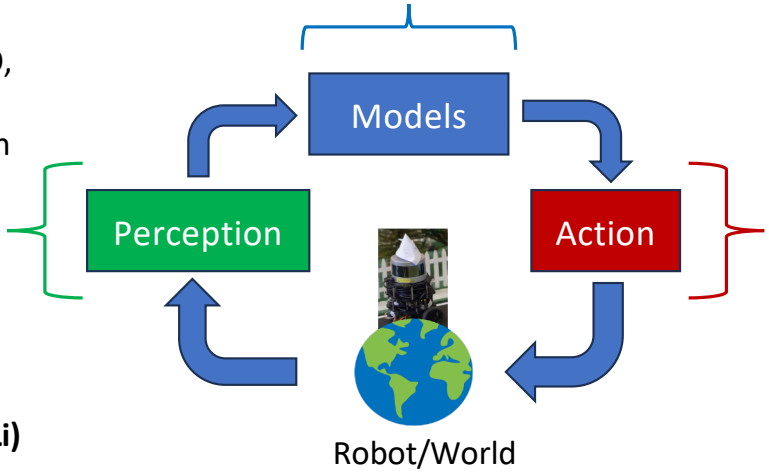
Yes! But only a little in this class. We treat classical, model-based autonomy.

Learning Based Perception Models:

- Extremely common, widely used in real robots, especially in semantic tasks
- Object detectors (YOLO, OWL-ViT)
- Object segmentation (SAM, SAM2)
- General vision-semantic features (DINO, CLIP)
- Geometric tasks SfM, SLAM, localization (VGGT, DUSt3R, SpatialTracker, Foundation Pose), we still often use classical techniques
- **CS227: Robot Perception (Song)**
- **CS231A: 3D Computer Vision (Bohg, Savarese)**
- **CS231N: Deep Learning for CV (Adeli, Li)**

Learning for Models

- Learned Maps: Neural Radiance Fields (NeRFs), Gaussian Splatting (GSplat)
- “World Models”: Learn function from action-image history to next image. Often learns from an existing simulator (not real world).
- Model Based Reinforcement Learning (MBRL)
- **AA273: State Estimation and Filtering (Schwager)**
- **AA 275: Navigation for Autonomous Systems (Gao)**

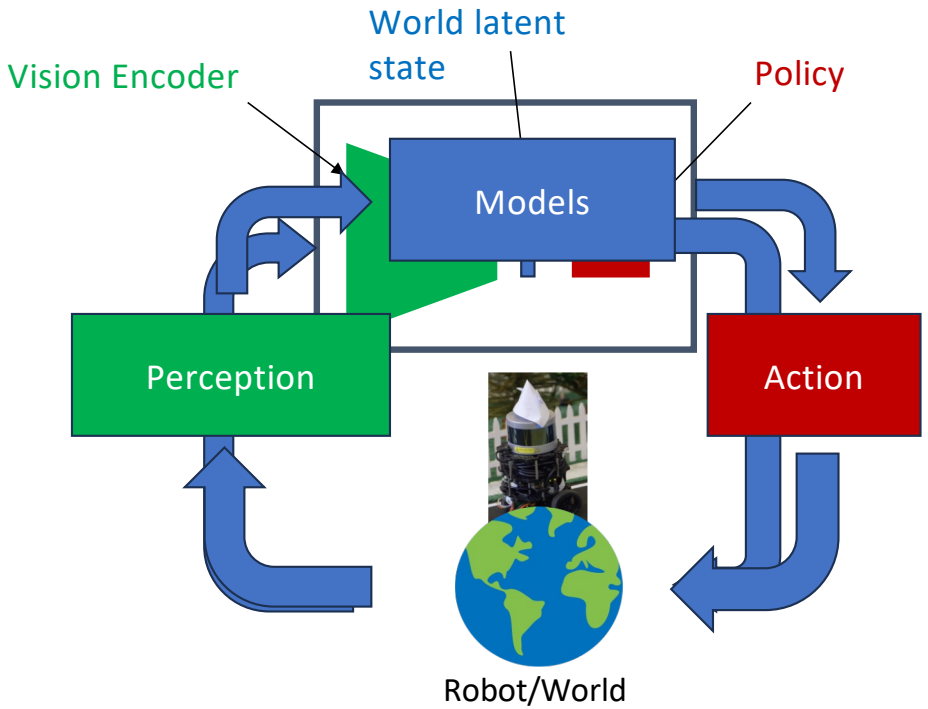


Reinforcement Learning (RL):

- Learn policy from state to action through incremental improvement
- Often heavily dependent on a simulator
- Sim-to-real a major challenge
- Often still relies on classical lower-level control layers
- **CS 224R Deep RL (Finn)**
- **CS 234 RL (Brunskill)**
- **AA203: Opt and Learning Cont (Pavone)**

What about End-to-End Learning?

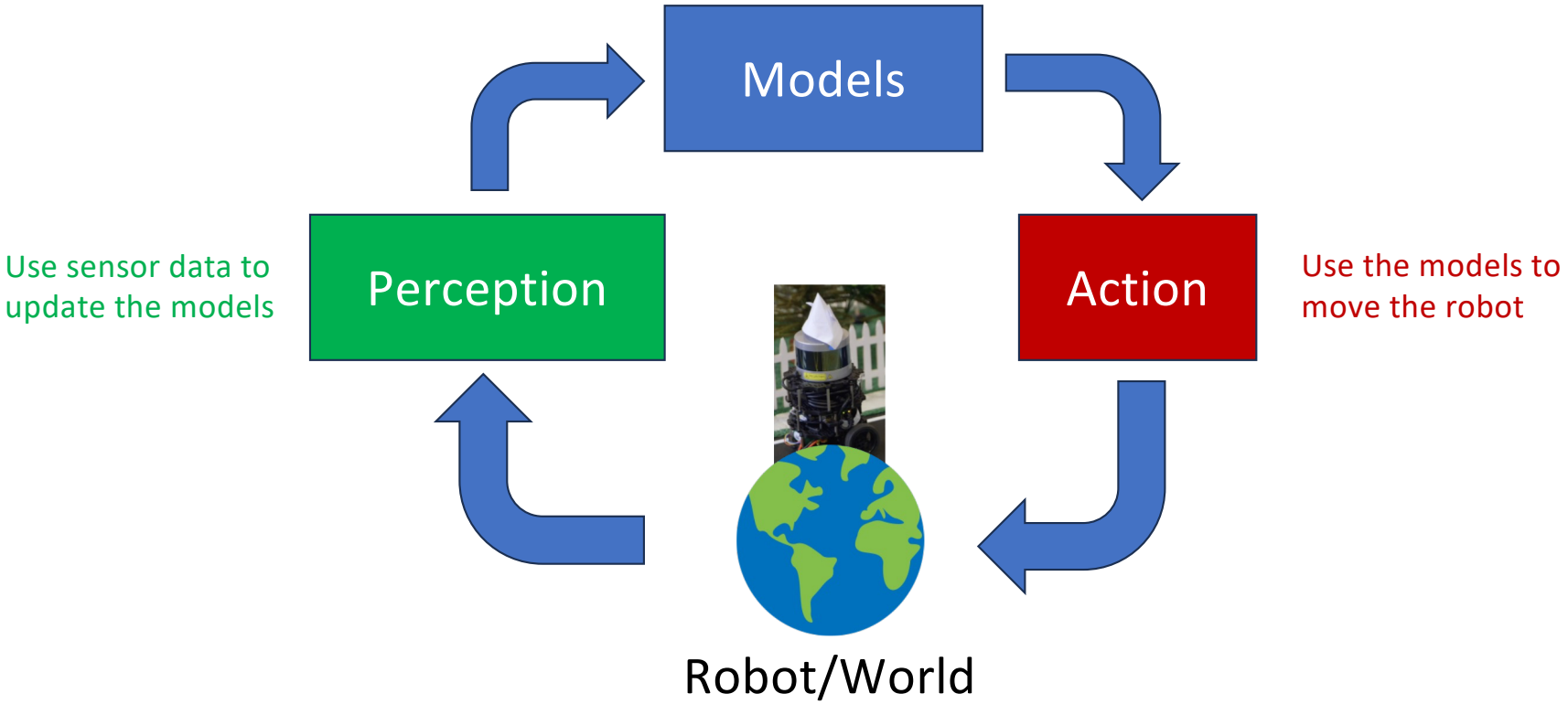
Yes, very cool! But not in this class.



End-to-End Visuo-Motor Policy

- Still have the three basic components internally in learning architecture
- Imitation learning (IL)/Behavior Cloning (BC)
  - Learn visuomotor policy from human demonstrations
  - Very successful in recent research!
  - **CS 237B Robot Autonomy II (Bohg, Sadigh, Pavone)**
- Reinforcement learning (RL)
  - Usually learned in simulators
  - Quite data-hungry
  - Sim-to-real is often a major challenge
  - **CS 224R: Deep RL (Finn)**
  - **CS 234: RL (Brunskill)**
  - **CS123: AI Enabled Robots (Liu)**

Full Stack Autonomy: the Perception-Action Loop



Logistics

Course structure

- Four modules
  1. Models: maps, kinematic and dynamic robot models
  2. Action: planning and control
  3. Perception: robot vision, point clouds, Kalman filtering, SLAM
  4. Advanced topics: NeRFs/3DGS, Imitation Learning/RL, VLAs
- Extensive use of the Robot Operating System (ROS2)
- Requirements
  - CS 106A or equivalent
  - CME 100 or equivalent (for calculus, linear algebra)
  - CME 106 or equivalent (for probability theory)
  - A pre-knowledge self-assessment will be sent out on Canvas today

Schedule

Models

Action

Perception

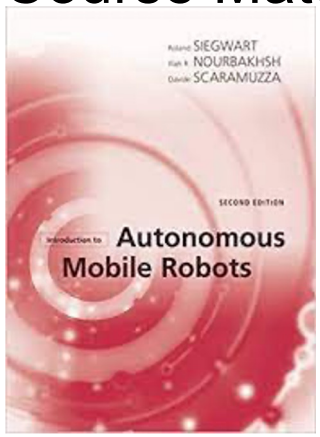
Date	Topic	Homework	Lab
09/23	Course overview, perception-action loop, maps		Lab 0: Install ROS
09/25	Maps, Robot geometry, Coordinate frames and SE(2)/SE(3) transforms	HW1 out	
09/30	Collision, C-space, motion models. Path planning I: A*		Lab 1: Command line, Git, Python
10/02	Path planning II: RRT, RRT*		
10/07	Trajectory optimization	HW1 due, HW2 out	Lab 2: ROS basics
10/09	Trajectory following: PID, LQR, gain scheduled LQR		
10/14	Robotic sensors: IMU, lidar, cameras, RGB-D. Point clouds & ICP		Lab 3: RViz, Turtlebot
10/16	Pinhole camera models, camera calibration	HW2 due, HW3 Out	
10/21	Structure from Motion (SfM), features, RANSAC		Lab 4: Heading controller
10/23	Learning based perception, semantic perception		
10/28	SLAM intro, factor graphs, PGO	HW3 due	Lab 5: Nav to goal
10/29	Midterm: 48 hour window	Midterm out	
10/30	Pose graph opt, bundle adjustment		
10/31	Midterm: 48 hour window window	Midterm due	
11/04	*No Lecture (Democracy Day)*	HW4 out	
11/06	Bayes Rule, RVs, Occ. mapping (Recorded Lecture, Mac traveling)		
11/11	Occ, mapping, frontier exploration		Lab 6: Object detection
11/13	Gaussian RVs, Kalman Filtering, EKF, UKF	HW4 due, HW5 out	
11/18	Particle Filtering, Monte Carlo localization		Lab 7: Frontier exploration
11/20	Guest Lecture		
11/25	*No lecture (Thanksgiving)*		
11/27	*No lecture (Thanksgiving)*		
12/02	EKF Localization, obj tracking	HW5 due	Lab 8: Makeup
12/04	Advanced Topics: Imitation learning, VLAs, 3DGS for sim2real, world models		
12/07	48 hour window take home final start	Final Exam out, 6:30pm	
12/09	48 hour window take home final start	Final Exam due, 6:30pm	

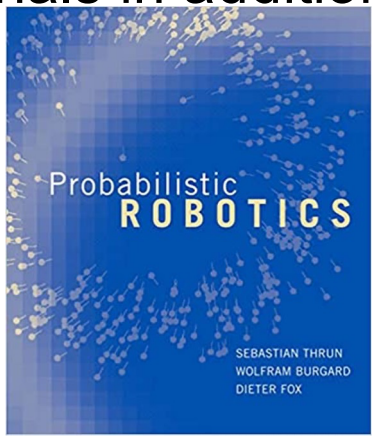
Advanced topics

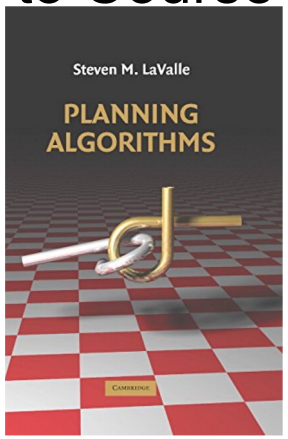
Logistics - Lectures

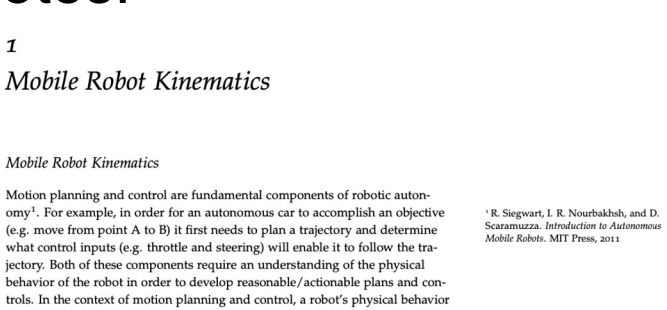
- Tuesdays and Thursdays, 1:30am – 2:50 (Skilling Aud)
- Recordings will be made available to all students on Canvas.

- Course Materials in addition to Course Notes:









1

Mobile Robot Kinematics

Mobile Robot Kinematics

Motion planning and control are fundamental components of robotic autonomy[1]. For example, in order for an autonomous car to accomplish an objective (e.g. move from point A to B) it first needs to plan a trajectory and determine what control inputs (e.g. throttle and steering) will enable it to follow the trajectory. Both of these components require an understanding of the physical behavior of the robot in order to develop reasonable/actionable plans and controls. In the context of motion planning and control, a robot's physical behavior

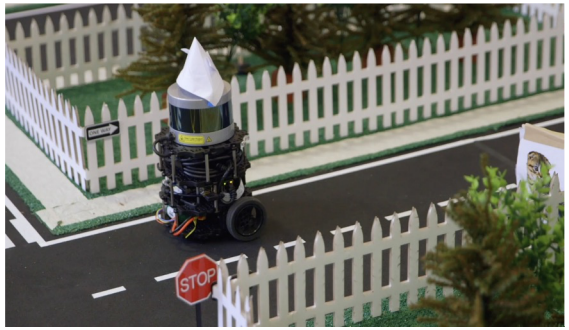
[1] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza. *Introduction to Autonomous Mobile Robots*. MIT Press, 2011

Logistics – Homework Assignments

- 5 assignments
- First homework out on Thursday
- ~2 weeks to submit on Gradescope
- Homeworks are crucial for sections, code development
- Plan ahead, and contact CAs if you need late days
- Cooperation and discussion is encouraged, but solutions must be prepared individually. Add names of classmates who you collaborated with. Copying from other students or other sources is considered a case of academic dishonesty.
- Need to be typeset in Latex!

Logistics – Sections

- 2-hour, once-a-week sessions starting Week 2
- Hands-on exercises that complement the lecture material, build familiarity with ROS,
- Progressively implement whole autonomy stack on turtlebot hardware
- Section sign-up sheet by webform is out now! (see Canvas announcement)
- Section timing details on Syllabus (see Canvas and course website)



Logistics – Exams

- Midterm exam: take home, 5 hour limit within 48 hours window.
- Final exam: take home, 5 hour limit within 48 hour window.

AI Policy

As specified by the Stanford Honor Code on Generative AI Policy:

Absent a clear statement from a course instructor, use of or consultation with generative AI shall be treated analogously to assistance from another person. In particular, using generative AI tools to substantially complete an assignment or exam (e.g. by entering exam or assignment questions) is not permitted. Students should acknowledge the use of generative AI (other than incidental use) and default to disclosing such assistance when in doubt.

1. AI Coding Assistants: These are modern productivity tools integral to most software engineering projects. Use them but **understand your code! It is your responsibility to learn what the code means and how it works.**

2. Chat GPT, Gemini, Etc. on Homeworks or Exams (except coding): Don't do it. If you do, cite it and justify the reason. Points may be deducted. If you use it and do not cite it, we will escalate the issue.

Logistics - Grades

- 20% Homeworks (5 x 4% each)
- 40% Sections (8 x 5% each)
- 15% Midterm
- 25% Final

Logistics

- Office hours: See Syllabus on Canvas
- Course websites:
  - **Cavas:** For syllabus, announcements, lecture videos, slides, and supplementary material
  - **Edstem:** For course-related questions and discussion
  - **Gradescope:** For homework submissions and grading
  - **email:** cs237a-aut2526-staff@lists.stanford.edu to contact the CS237A staff
  - **Legacy Website:** http://asl.stanford.edu/aa274a/ will be updated soon. Same info as Syllabus
- Syllabus has all the info!
- Class capacity, enrollment cap, waitlist

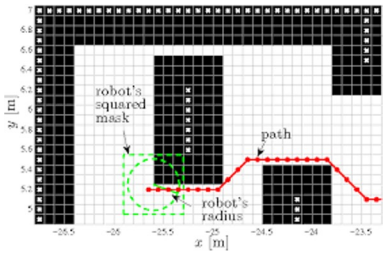
Announcements

- Sign up for section **by Thursday**. Ranked Choice – NOT first come-first-serve.
- Sign up if you are on waitlist and still want to take course.
- Homework 1 will be released Thursday (canvas)
  - due Tues 10/07 (by gradescope)
- Watch AA174A Section 0 video and Notion Instructions- set up your Virtual Machine or Docker Container and install ROS 2.0
- Pre-knowledge self-assessment out today (canvas)

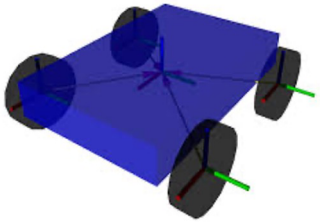
Models:
Environment models, Robot models, Agent models

Models: Environment, Robot, Agent Representations

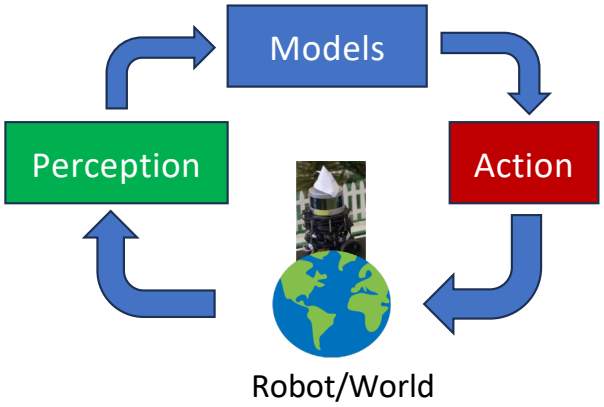
Maps:



Robot models
or simulators:



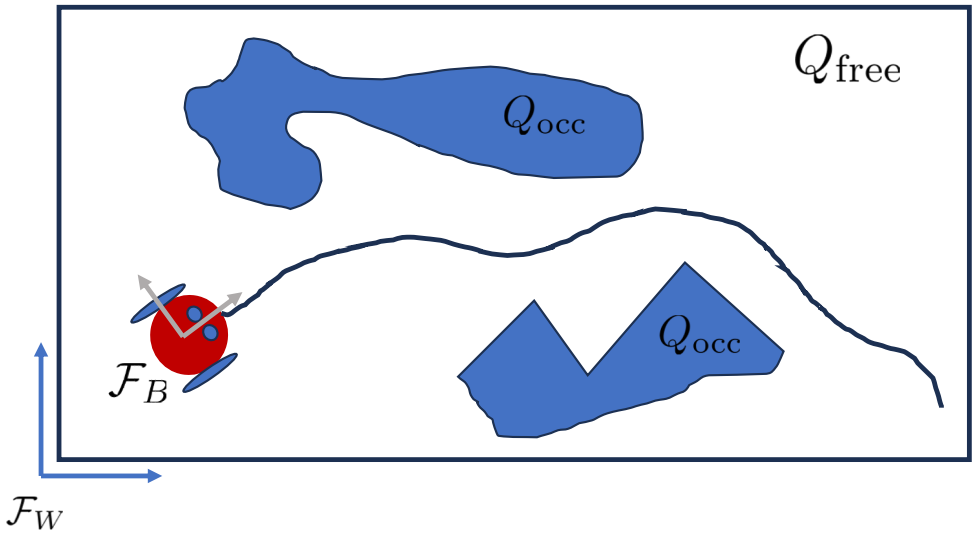
Agent models or
simulators:



Maps

Purpose: To represent free and occupied space in a *fixed world frame*. Used for collision avoidance and path planning in robot navigation.

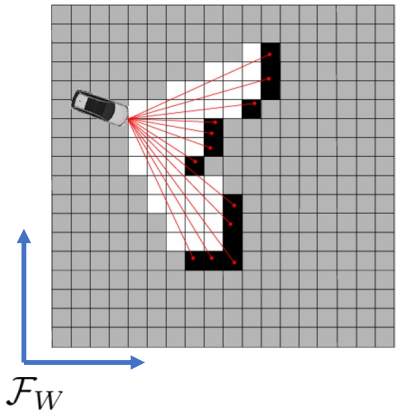
$$Q = Q_{\text{free}} \cup Q_{\text{occ}} \quad \text{so} \quad Q_{\text{free}} = Q_{\text{occ}}^{C}$$



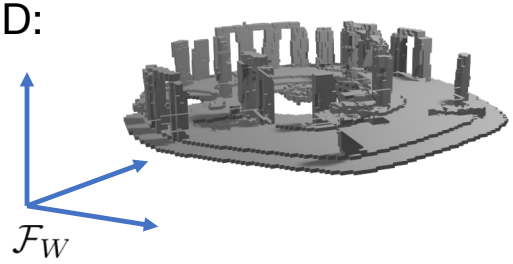
How to mathematically represent these sets?
How to update them from robot sensor measurements as the robot moves?

Occupancy Grid Map

2D:



3D:



Grid cell i:

Location $q_i = (q_i^x, q_i^y)$

State $x_i \in \{0, 1\}$

$$Q = Q_{\text{free}} \cup Q_{\text{occ}}$$

$$Q_{\text{free}} = \{q_i \mid x_i = 0\}$$

$$Q_{\text{occ}} = \{q_i \mid x_i = 1\}$$

Probabilistic occupancy grid:

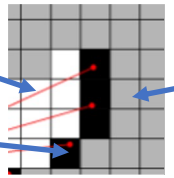
$$p_i = p(x_i = 1 \mid \text{sensor measurements})$$

$$p_i \in [0, 1]$$

Probability threshold:

$p_i \leq 0.05$

$p_i \geq 0.95$



$0.05 \leq p_i \leq 0.95$

Log-Odds:

$$l_i = \log(\frac{p_i}{1 - p_i}) \in [-\infty, \infty]$$

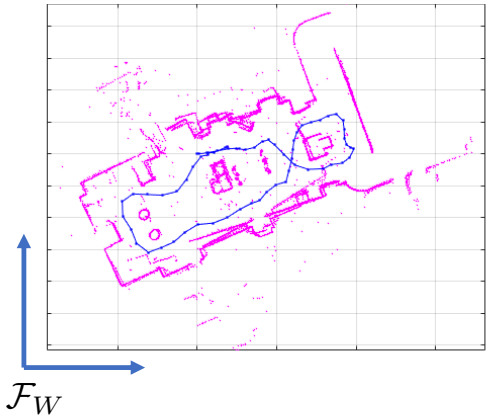
$$l_i = 0 \iff p_i = 0.5$$

Makes updating with Bayes' rule very simple (we'll see later)

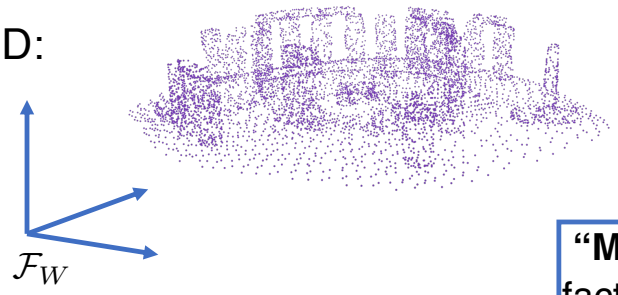
"Mapping": Update all p_i with Bayes' rule in real time using robot sensor measurements.

Point Cloud Mapping

2D:



3D:



Points represent occupied space:

$$q_i \in \mathbb{R}^2 \text{ or } \mathbb{R}^3$$

Native output of a lidar sensor.

$$Q_{\text{occ}} = \{q_i\}_{i=1}^{N}$$

Still need to transform from robot body to world frame.

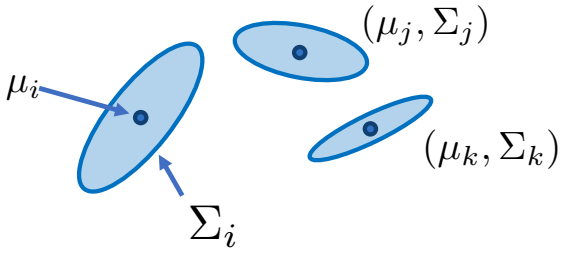
Probabilistic point cloud map:

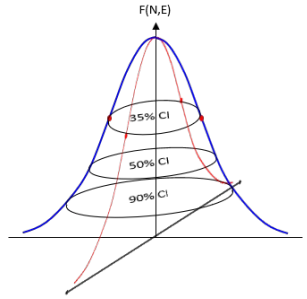
Multivariate Gaussian

$$p(q_i \mid \text{robot measurements}) = \mathcal{N}(\mu_i, \Sigma_i)$$

Mean vector

Covariance matrix





"Mapping": Update (μ_i, Σ_i) with a Kalman filter or a factor graph optimization solver.