
# Initialization

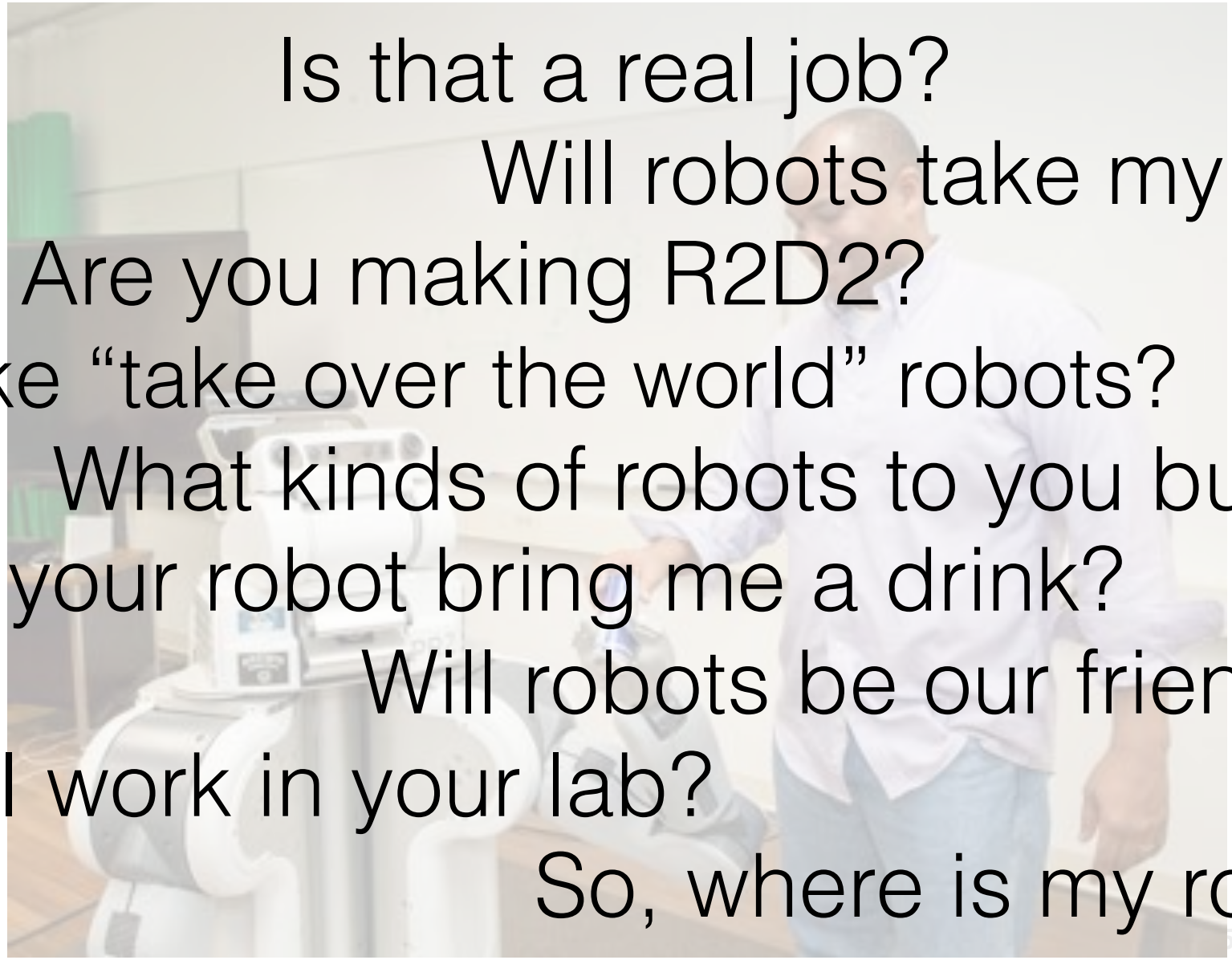
EECS 367
Intro. to Autonomous Robotics

ME/EECS 567 ROB 510
Robot Modeling and Control

Fall 2019

# Agenda

- So, where is my robot?
- Roadmap for autonomous robotics
- Course administrative overview
- Assignment 1 (Path Planning) assigned today, due Sep 18 11:59pm
  - JavaScript/HTML5 and git covered this and next lecture
- Action items: what I need from you now



Is that a real job?

Will robots take my job?

Are you making R2D2?

Like “take over the world” robots?

What kinds of robots to you build?

Can your robot bring me a drink?

Will robots be our friends?

Can I work in your lab?

So, where is my robot?

So, where is my robot?



So, where is my robot?

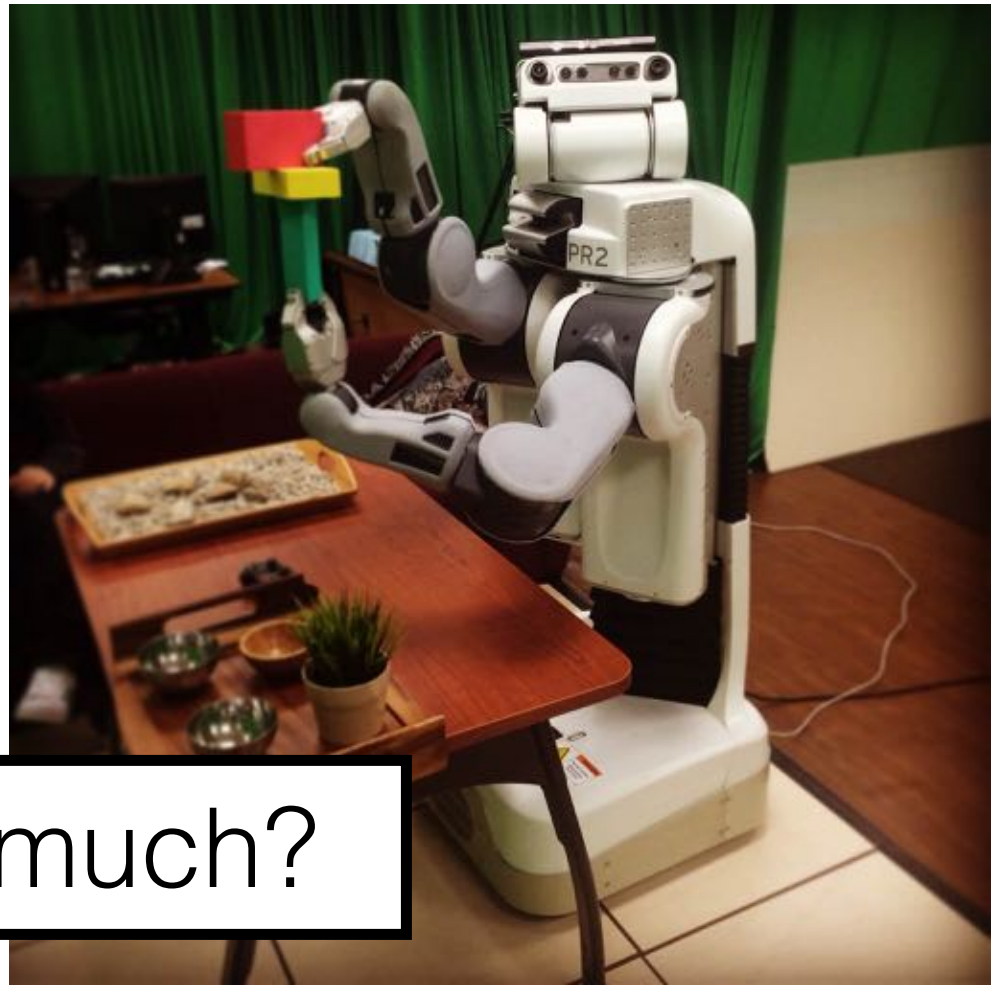


# Mobile Manipulation Robots

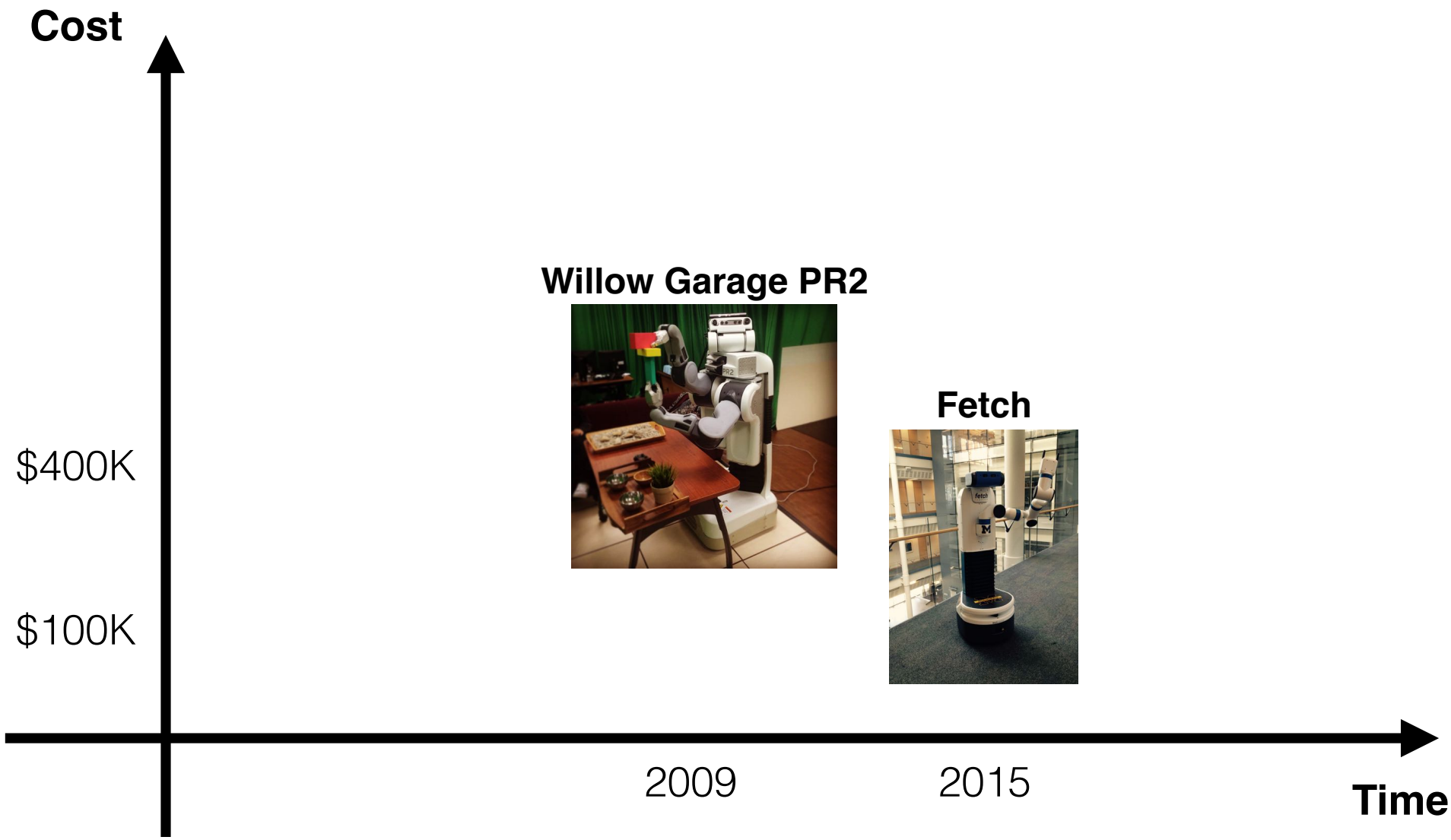


fetch


PR2


How much?


Cost
Willow Garage PR2
Fetch
$400K
$100K
2009
2015
Time

Cost

2002

Willow Garage PR2

Fetch

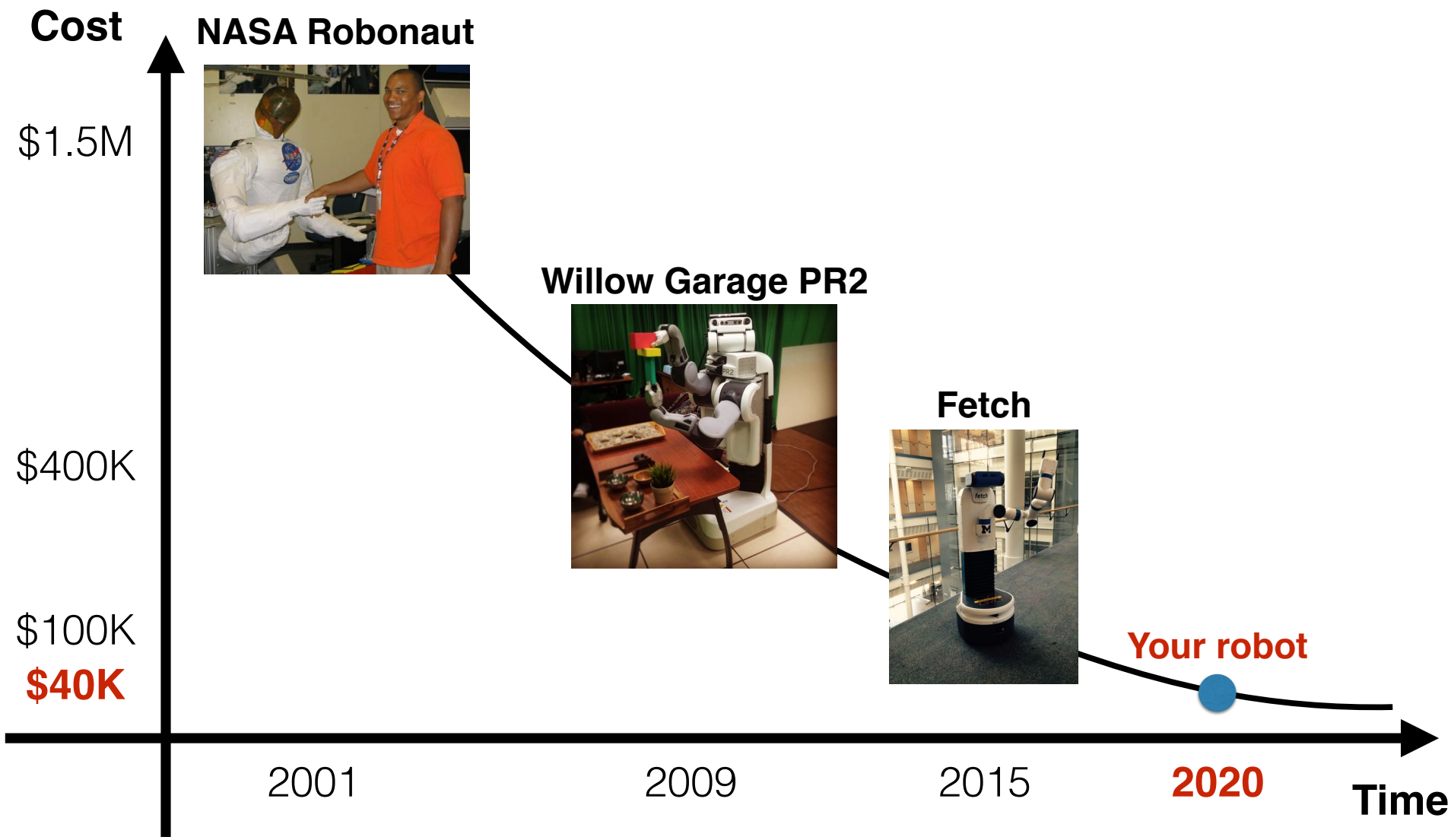
$400K

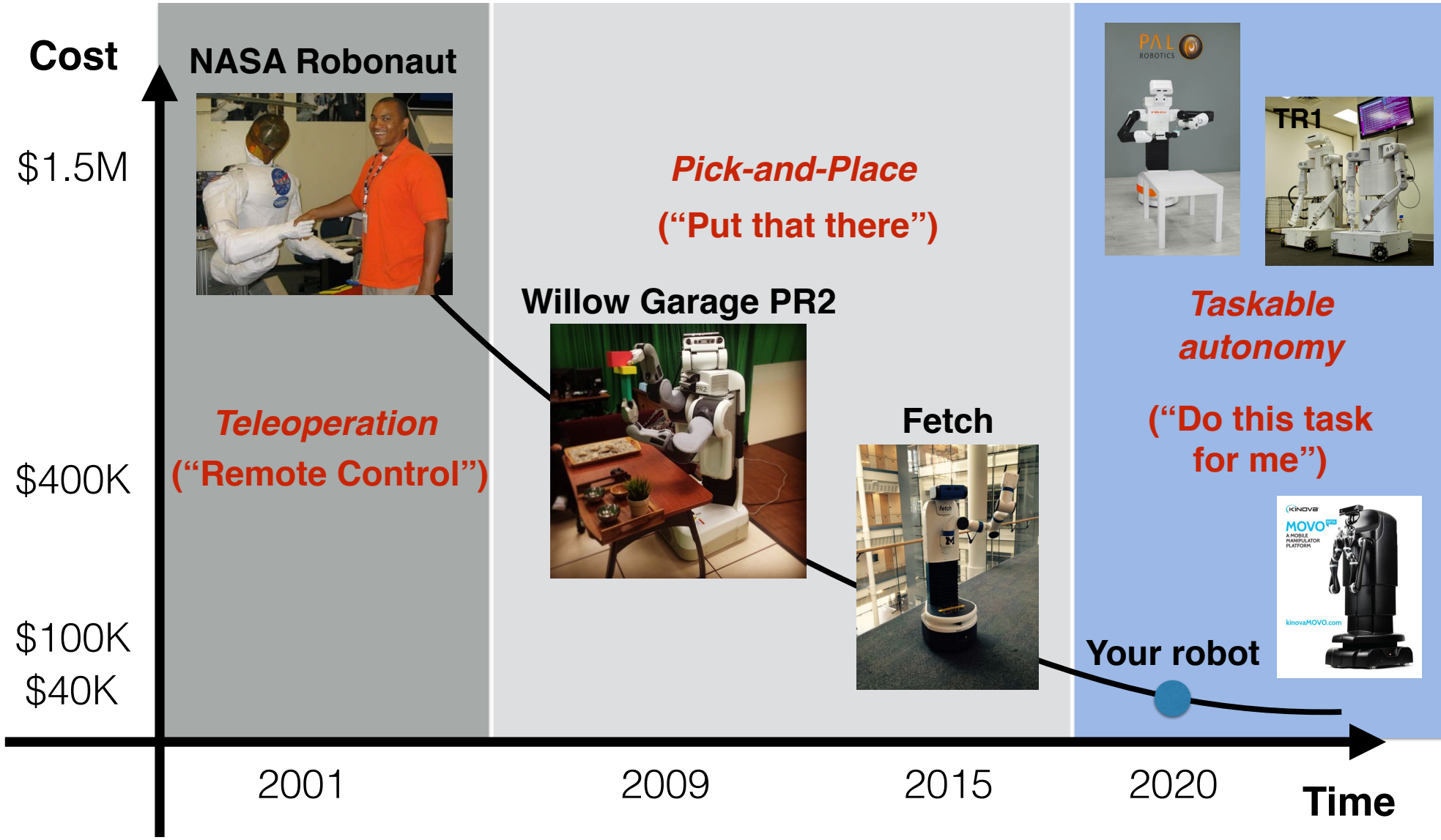
$100K

2009

2015

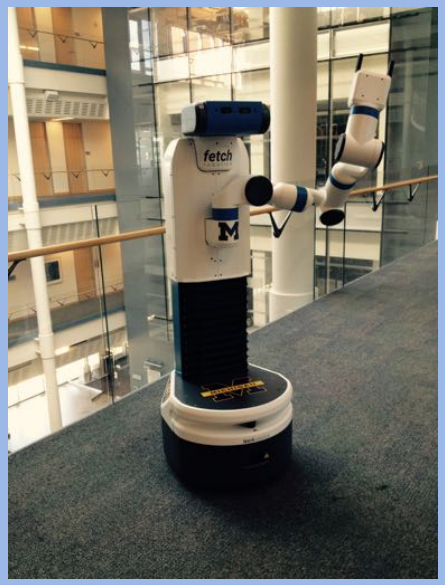
Time


Cost
NASA Robonaut
$1.5M
Willow Garage PR2
Fetch
$400K
$100K
$40K
Your robot
2001
2009
2015
2020
Time


Cost
$1.5M
$400K
$100K
$40K
NASA Robonaut
Teleoperation
("Remote Control")
Pick-and-Place
("Put that there")
Willow Garage PR2
Fetch
Taskable autonomy
("Do this task for me")
TR1
Your robot
2001
2009
2015
2020
Time

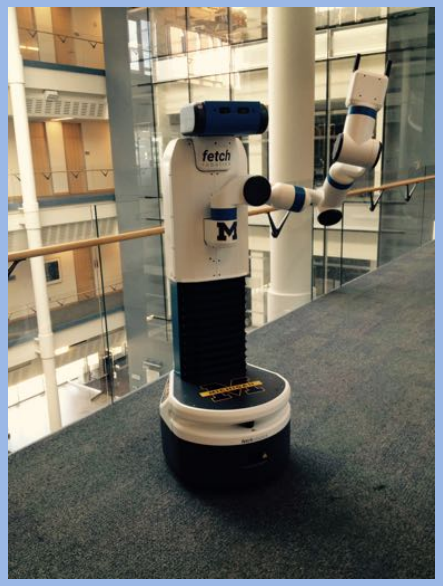
“Do this task for me”

Can we make your
**world programmable** ?


fetch

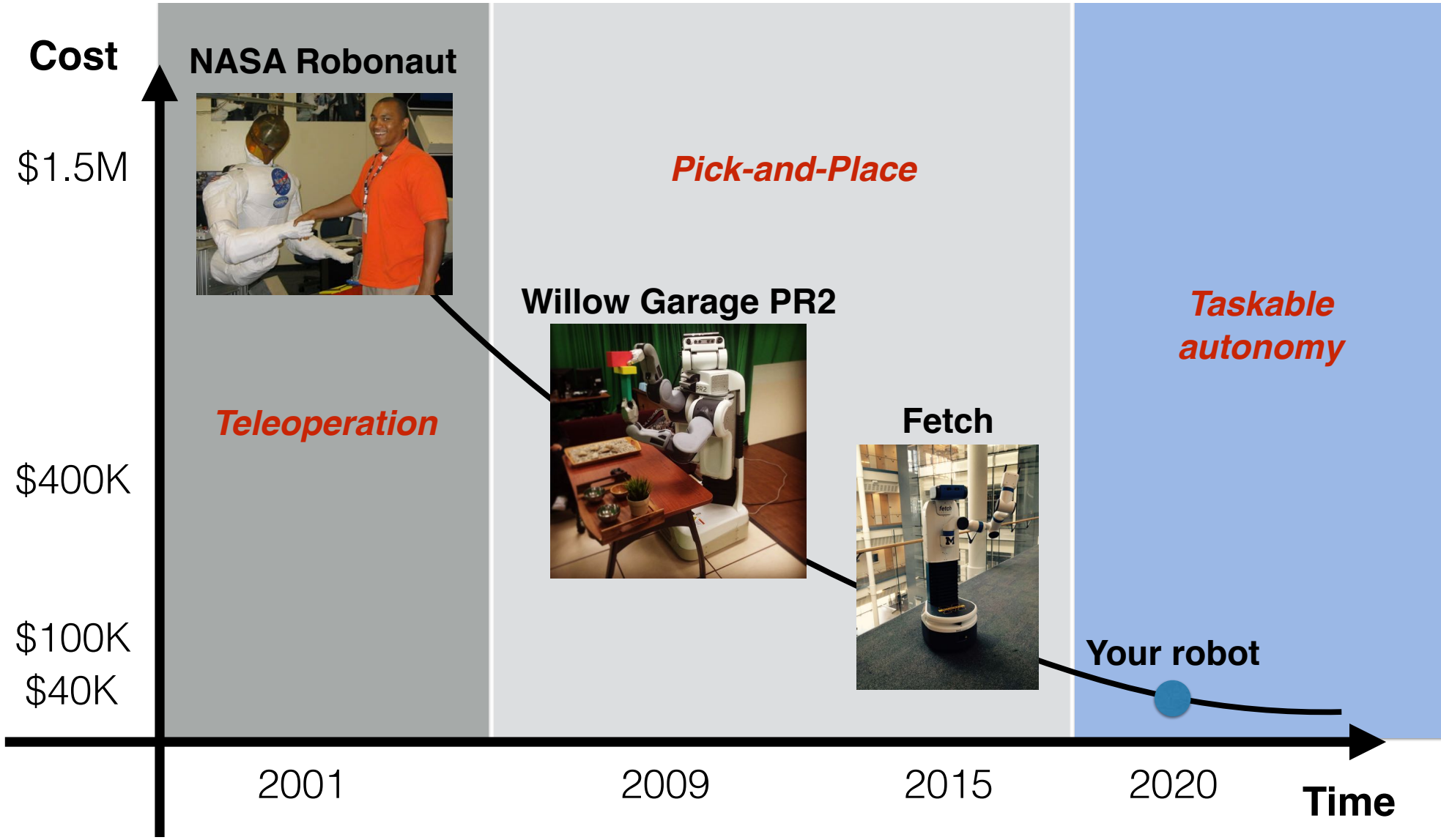

PR2

Use $x$ robot




to perform task $y$

in $z$ environment


Cost
NASA Robonaut
$1.5M
Pick-and-Place
Willow Garage PR2
Taskable autonomy
Teleoperation
Fetch
$400K
$100K
$40K
Your robot
2001
2009
2015
2020
Time

## *Teleoperation*

**NASA Robonaut 2 on the International Space Station**

Wearable wireless motion capture for robot teleoperation

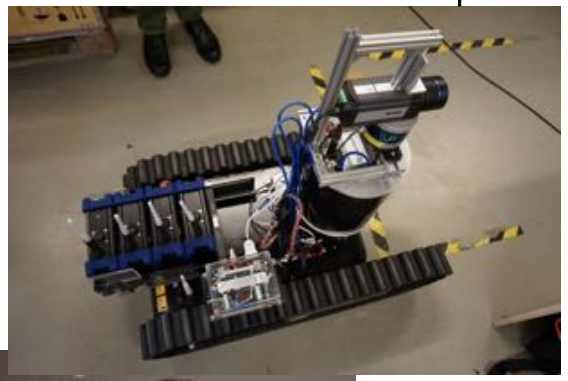

**[Miller, Jenkins, et al., 2004, "Motion Capture from Inertial Sensing for Untethered Humanoid Teleoperation"]**

# The 3Ds: Dirty, Dull, and Dangerous

"Autonomous" Driving

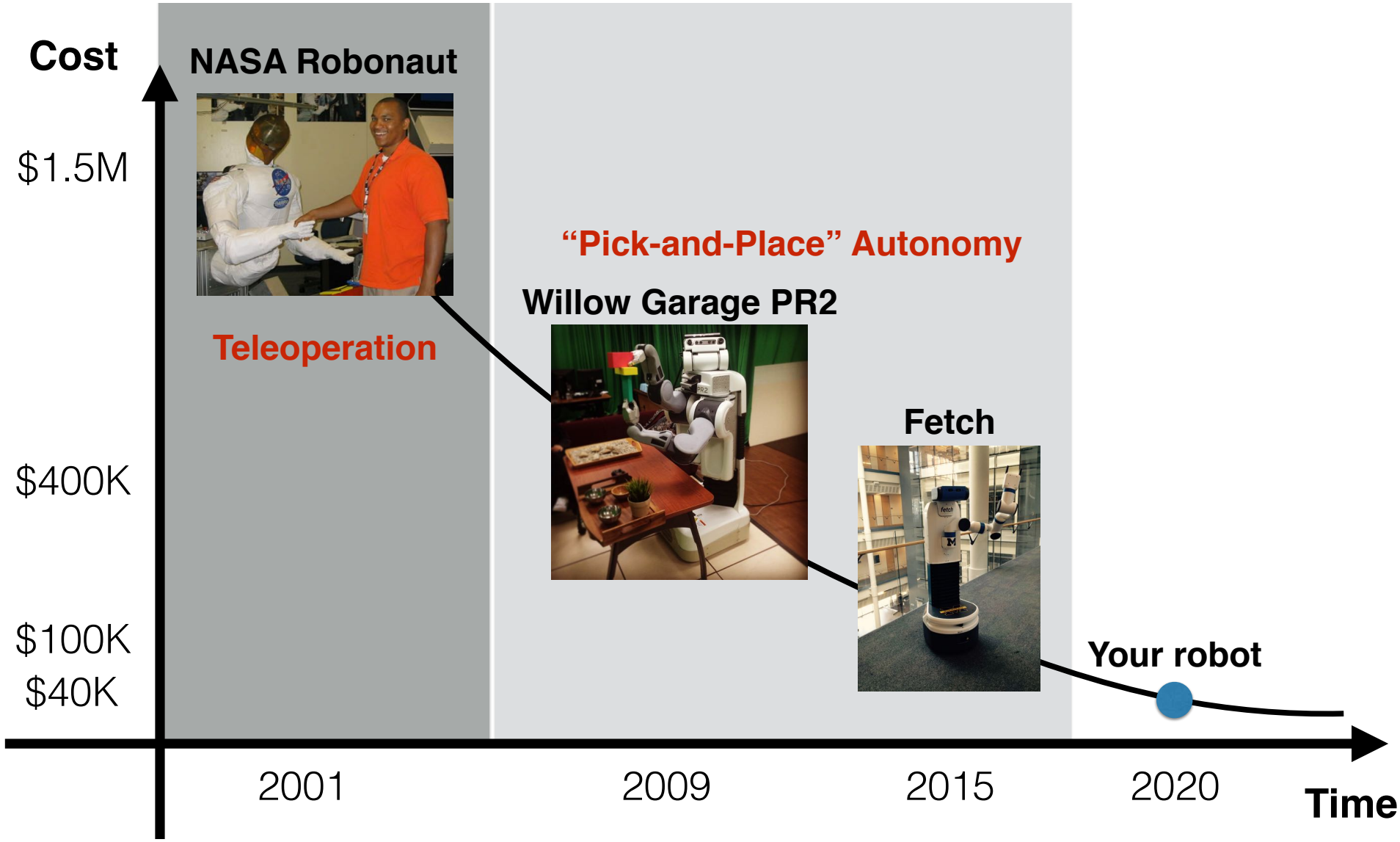

Infrastructure inspection

Nuclear cleanup



https://www.shadowrobot.com/blog/robots-saving-humans-from-dangerous-jobs/

https://techcrunch.com/2018/06/05/remote-control-driverless-car-startup-partners-with-vehicle-manufacturers/


Cost
NASA Robonaut
$1.5M
Teleoperation
"Pick-and-Place" Autonomy
Willow Garage PR2
Fetch
$400K
$100K
$40K
Your robot
2001
2009
2015
2020
Time

# Autonomous Robotics in 3 words

Sense.
Plan.
Act.

# Autonomous Robotics in 3 words

Sense. **Perceive** a model of the current world state.

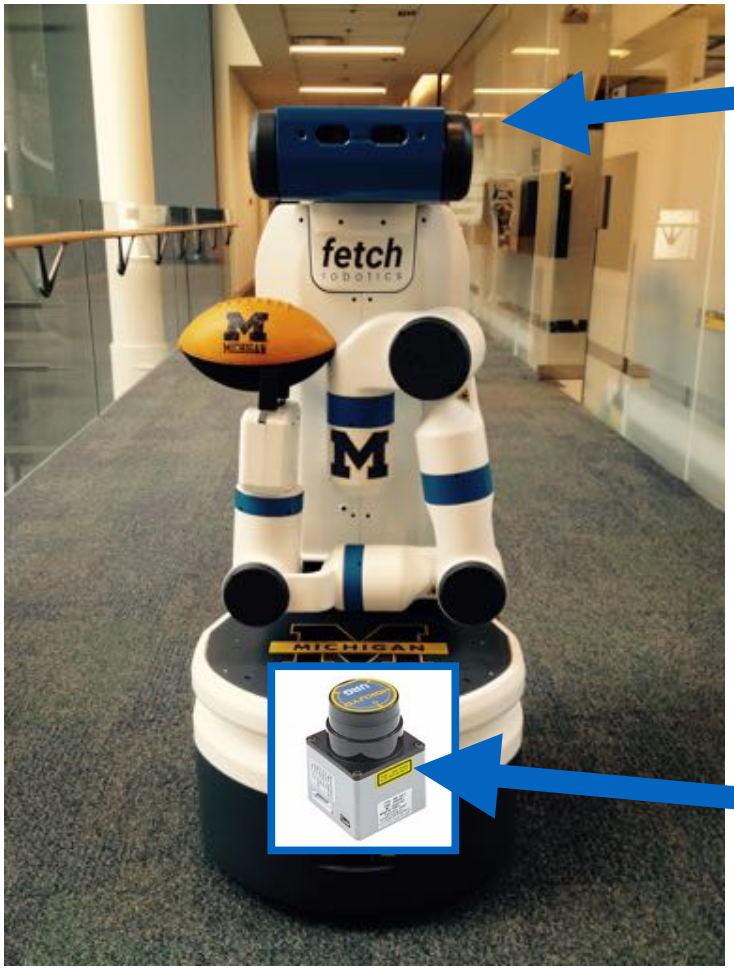
Plan. **Search** over actions towards a goal state.

Act. **Execute** actions through forces at robot's motors



Color+Depth Camera

Laser Rangefinder



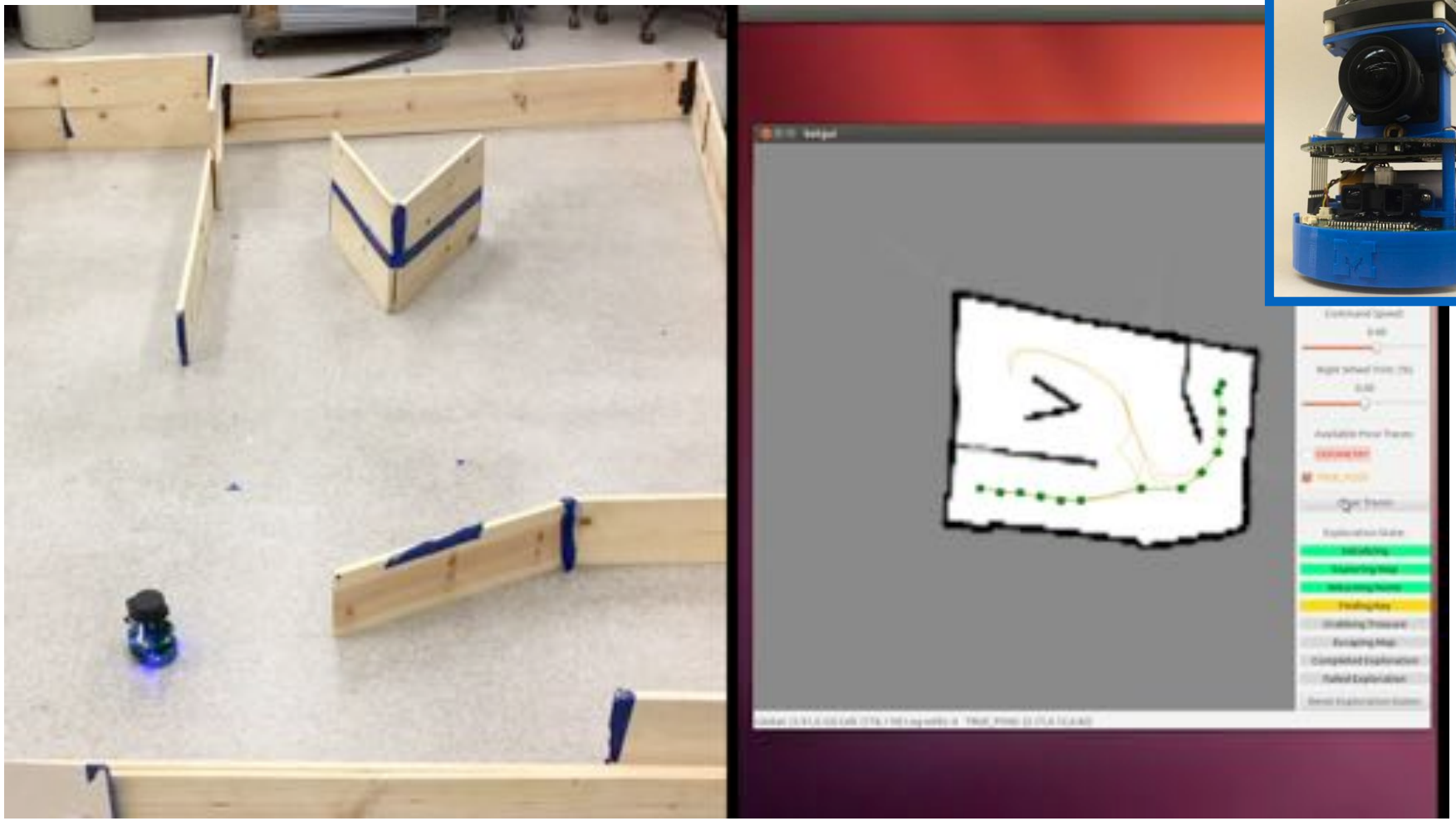


Color+Depth Camera

Laser Rangefinder

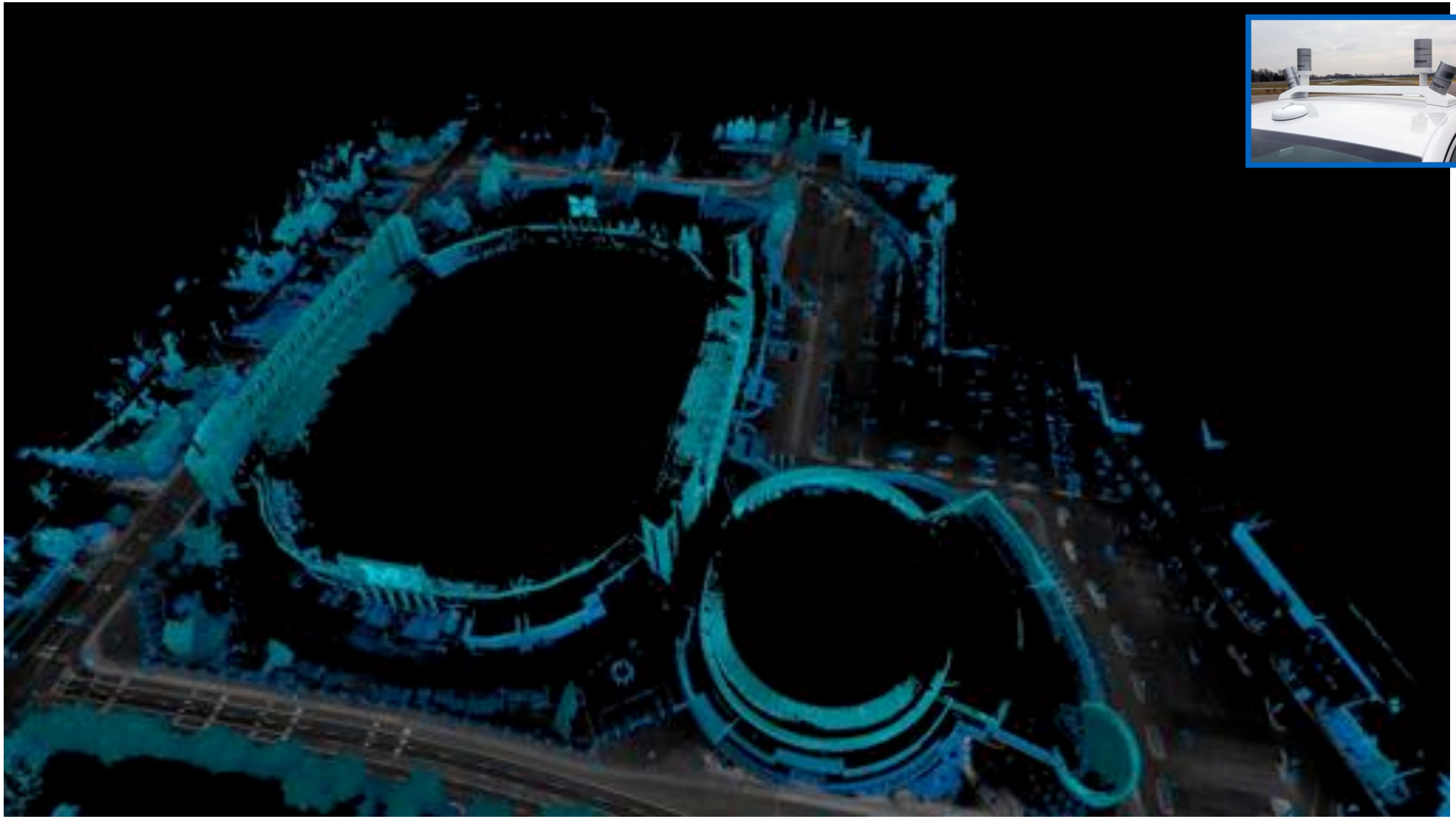

fetch
ROBOTICS
MICHIGAN
fetchrobotics

ROB 550 BotLab / EECS 467 Escape Challenge 2016-17



Jasmine Liu et al.

Michigan Next Generation Vehicle (Olson, Eustice, et al.)

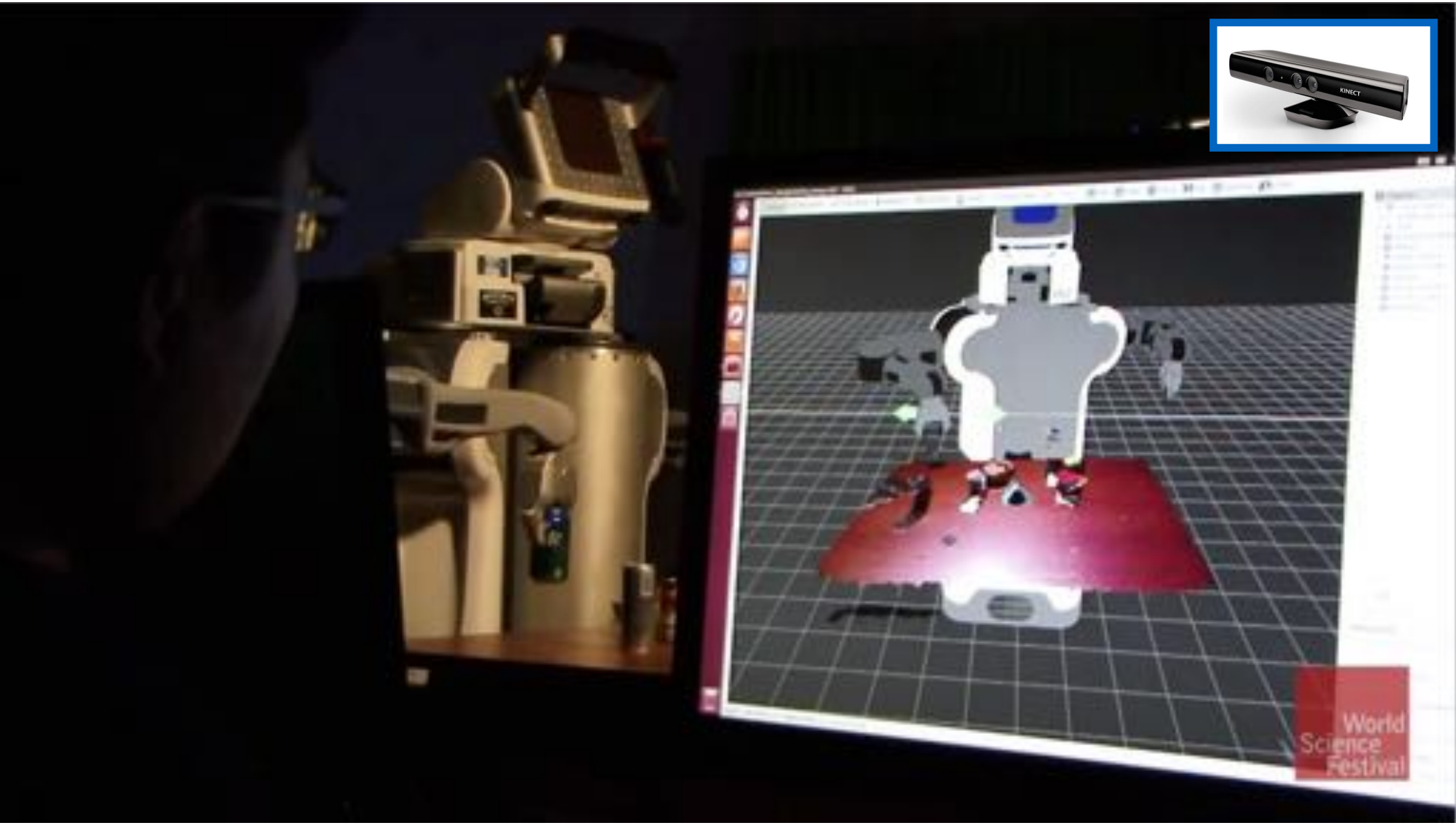


Michigan Next Generation Vehicle (Olson et al.)



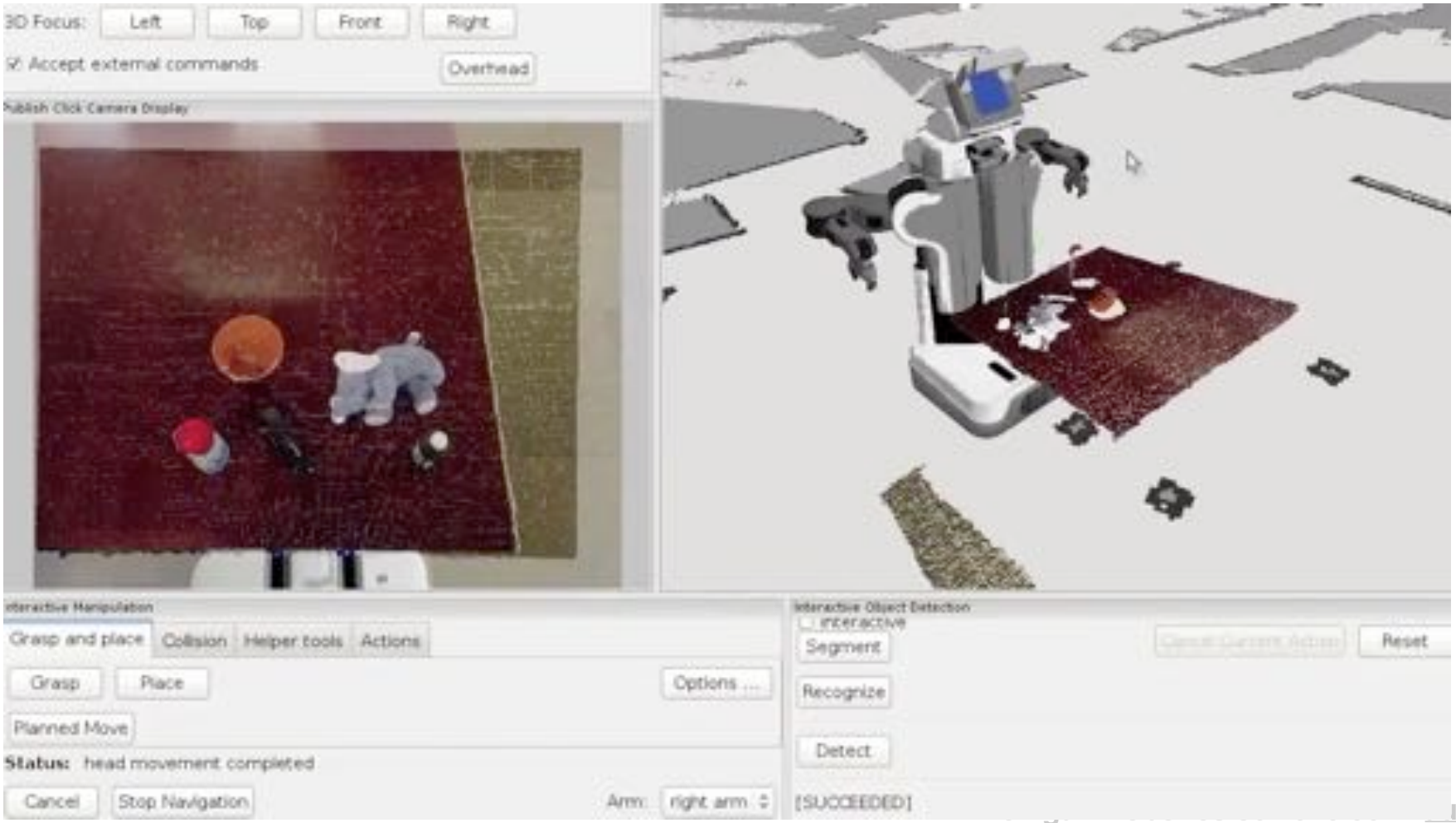
Color+Depth Camera

Laser Rangefinder

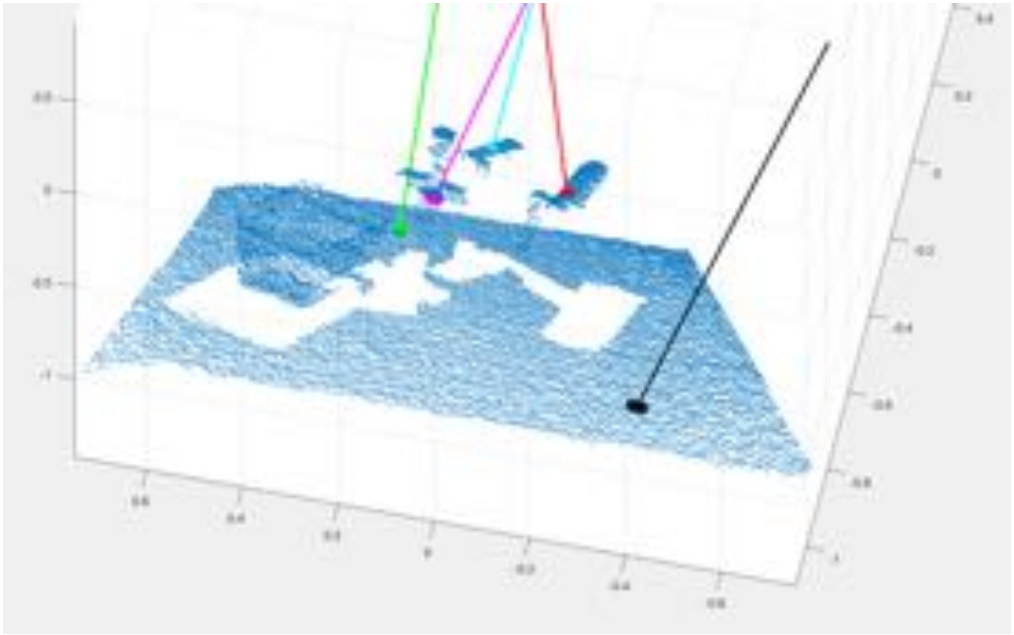

KINECT
World
Science
Festival

# Object Manipulation

Willow Garage



# Point Cloud Processing (briefly)



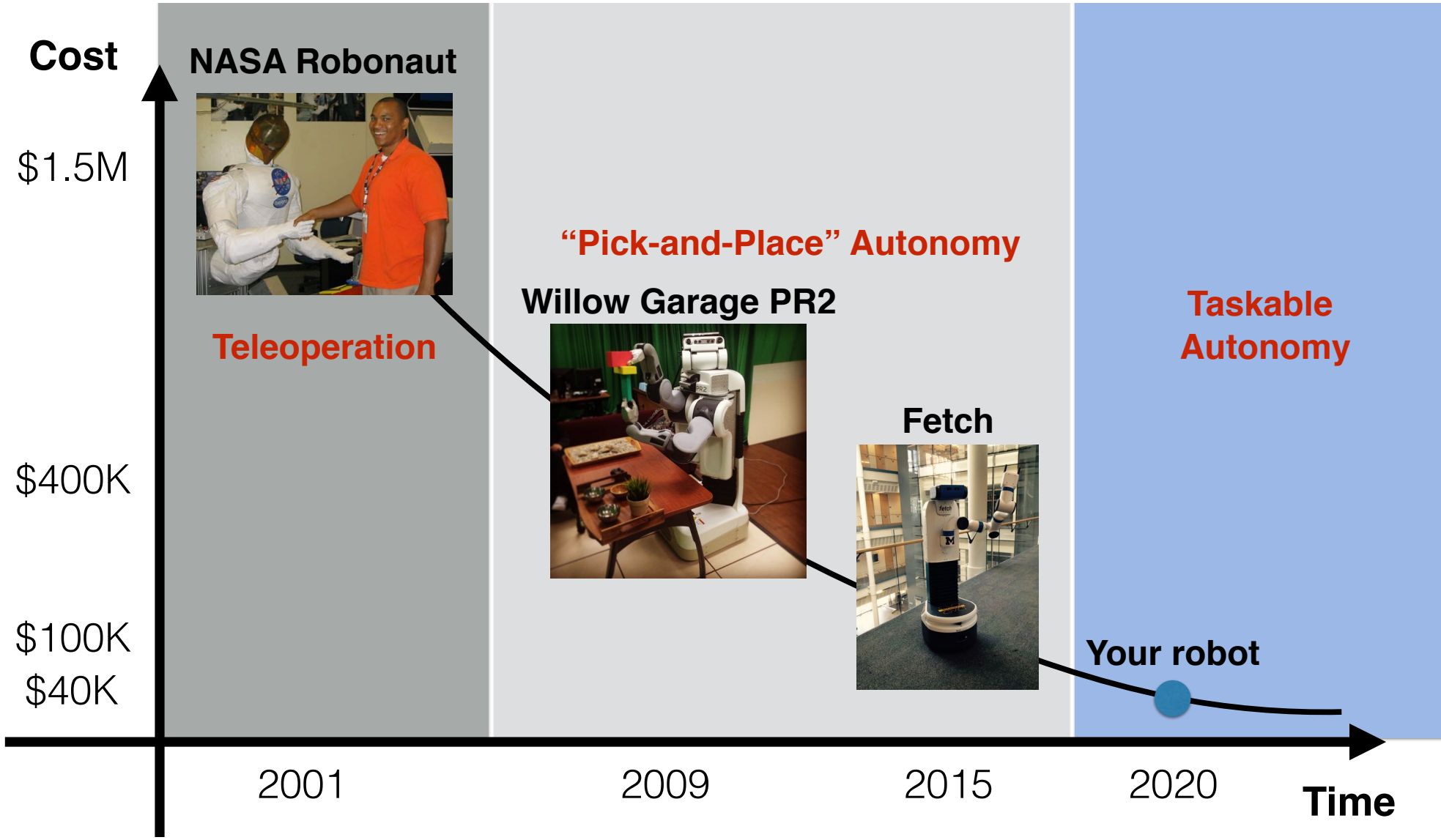
- For every point:
  - compute nearest neighbors
  - compute principal components in neighborhood `eig(cov(nbhd(:,1:3))`
  - surface normal is eigenvector for smallest eigenvalue
- Cluster points based on direction similarity of normal

# Object Manipulation

Willow Garage


itorob.org


[Sui, Jenkins, et al. IROS 2017]
GO BLUE


Cost
NASA Robonaut
$1.5M
Teleoperation
"Pick-and-Place" Autonomy
Willow Garage PR2
Taskable Autonomy
Fetch
$400K
$100K
$40K
Your robot
2001
2009
2015
2020
Time

# Taskable Manipulation


Initial Scene → Goal Scene

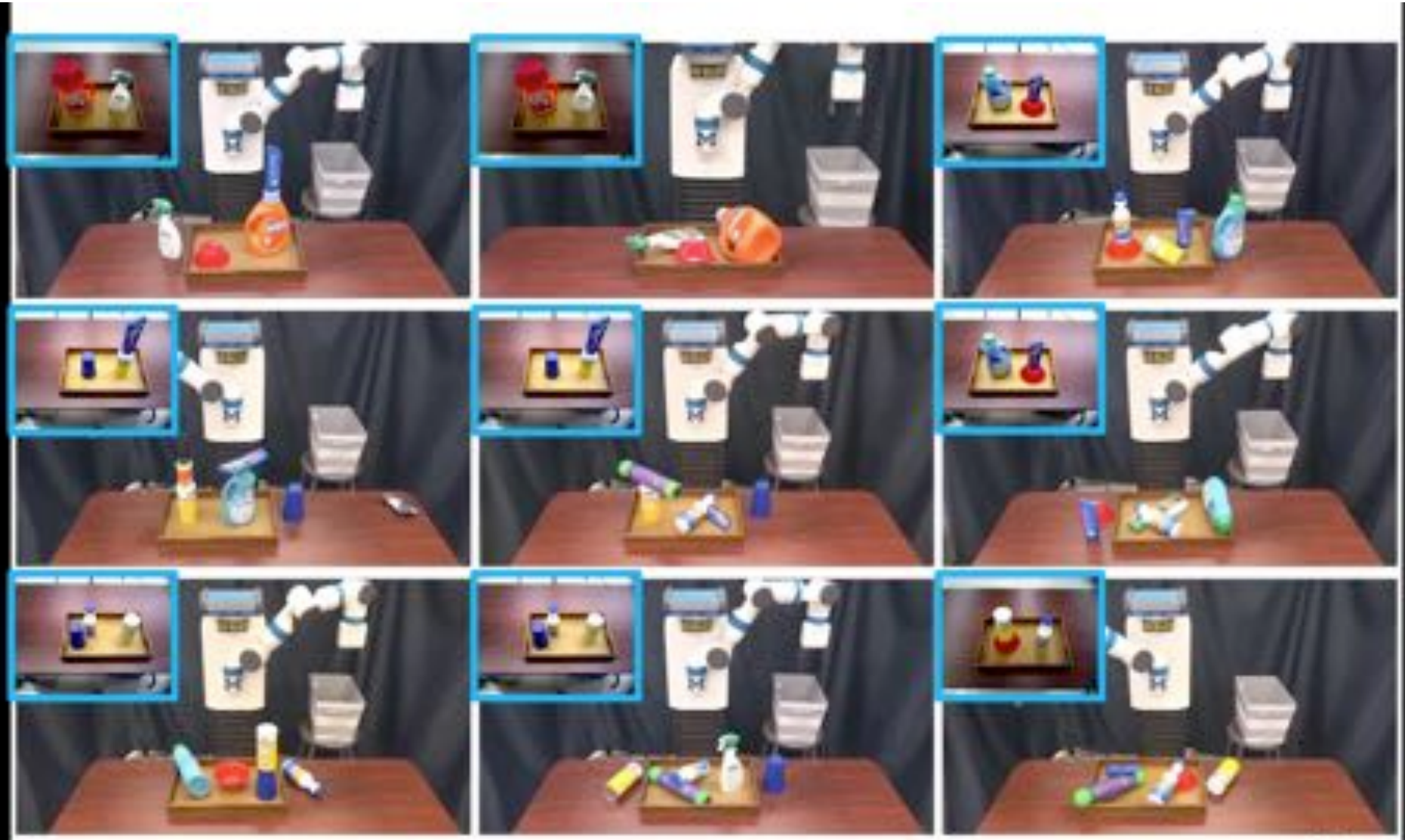
"put small objects in bin"


“put small objects in bin”
[Sui, Xiang, Jenkins, Desingh, IJRR 2017]

# Semantic Robot Programming

**[Zeng, Jenkins, et al. ICRA 2018]**


Enable natural user programming of robots
by demonstration of intended goal scenes



*Goals shown in upper left corners*

8x

# Agenda

- So, where is my robot?
- Roadmap for autonomous robotics
- **Course administrative overview**
- Assignment 1 (Path Planning) assigned today
  - JavaScript/HTML5 and git covered this and next lecture

# Course Staff

- Instructor: Chad Jenkins (ocj)
  - Office hours (Beyster 3644)
    - Monday 3-5pm, Tuesday 1-3pm
- GSI: Anthony Opipari (topipari)
  - Office hours (Beyster 1637 A)
    - Tuesday 1-3pm, Wednesday 3-5pm

# Administrivia

- Lecture time/place:  MW 1:30-2:40pm MMT, EECS 1500
  - 367 Lab Section: F 2:30-3:20 MMT, EECS 1500
- Website: http://autorob.org
- Discussion channel: https://autorob.slack.com/
  - This might switch to discord, mattermost, or IRC
- Office hours queue: https://oh.eecs.umich.edu/courses/eecs567

https://autorob.org

AUTOROB schedule kineval git assignments: 1 2

# AutoRob

## Introduction to Autonomous Robotics

Michigan EECS 367

## Robot Kinematics and Dynamics

Michigan ME 567 EECS 567 ROB 510

Fall 2019

https://autorob.org/#schedule

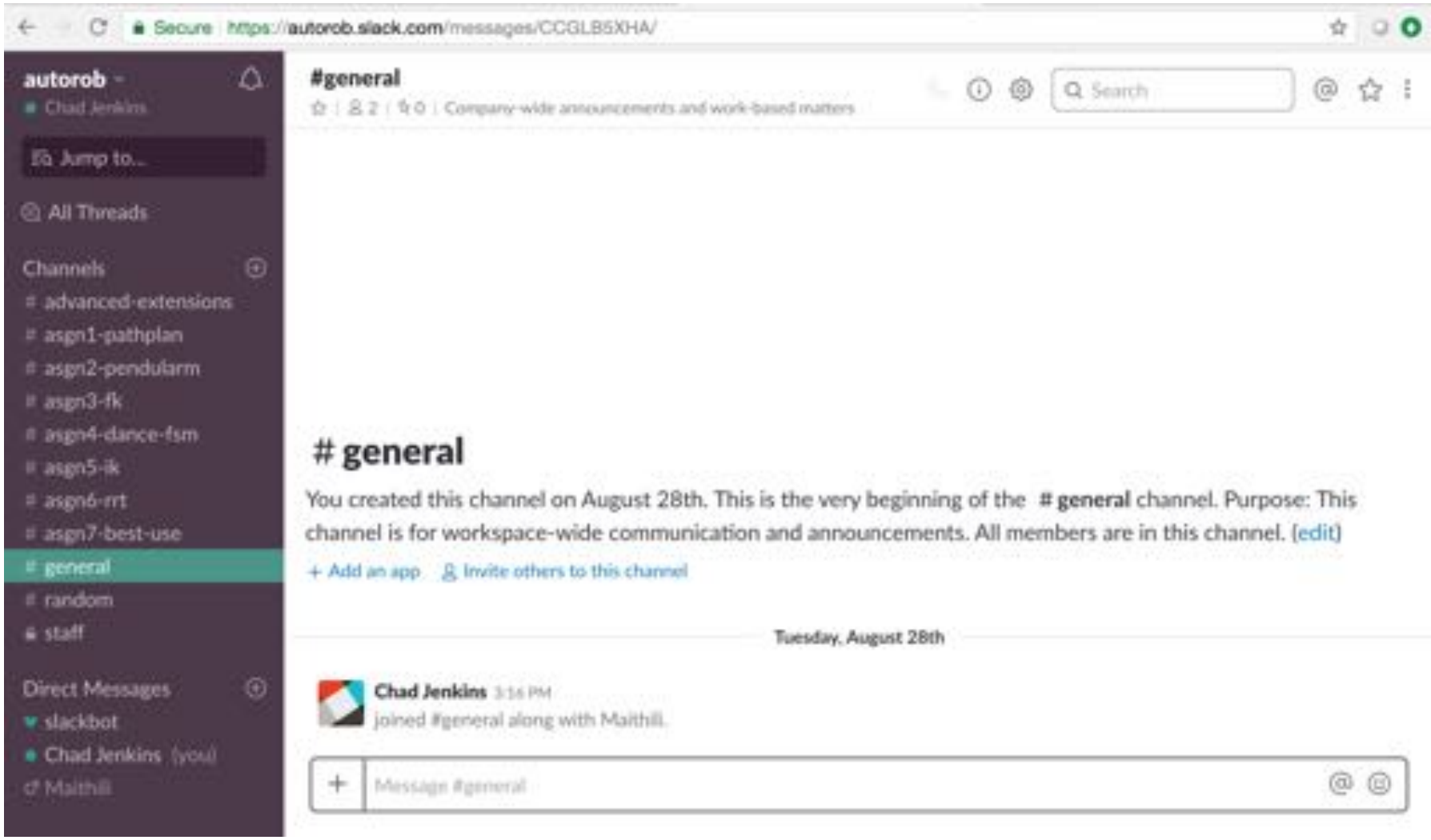
AUTOROB | schedule | kineval | git | assignments: 1 2

# Course Schedule (tentative and subject to change)

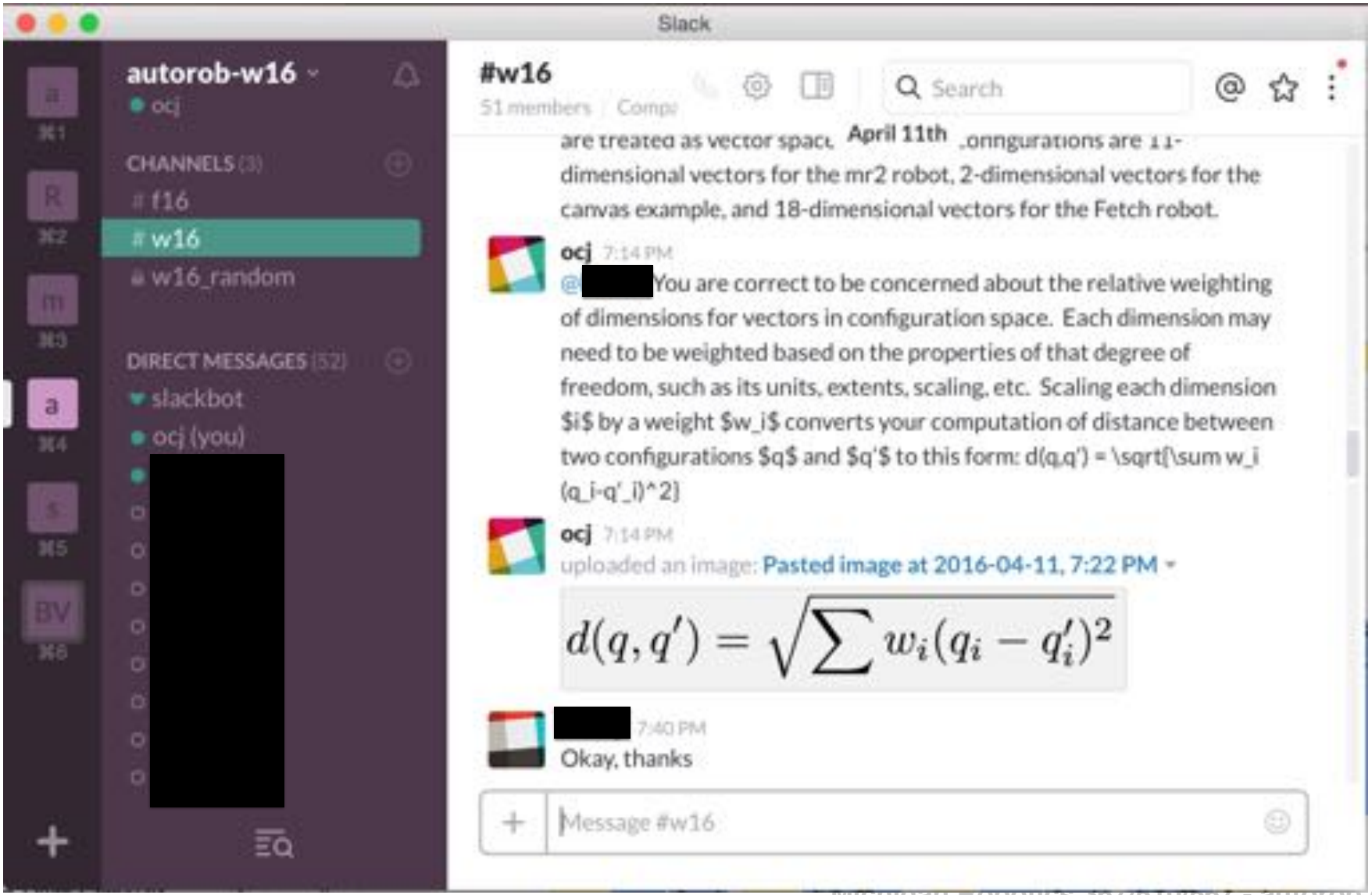
Note: Assignment descriptions will have updated assignment due dates. Assignment due dates listed in the schedule are merely a guide.

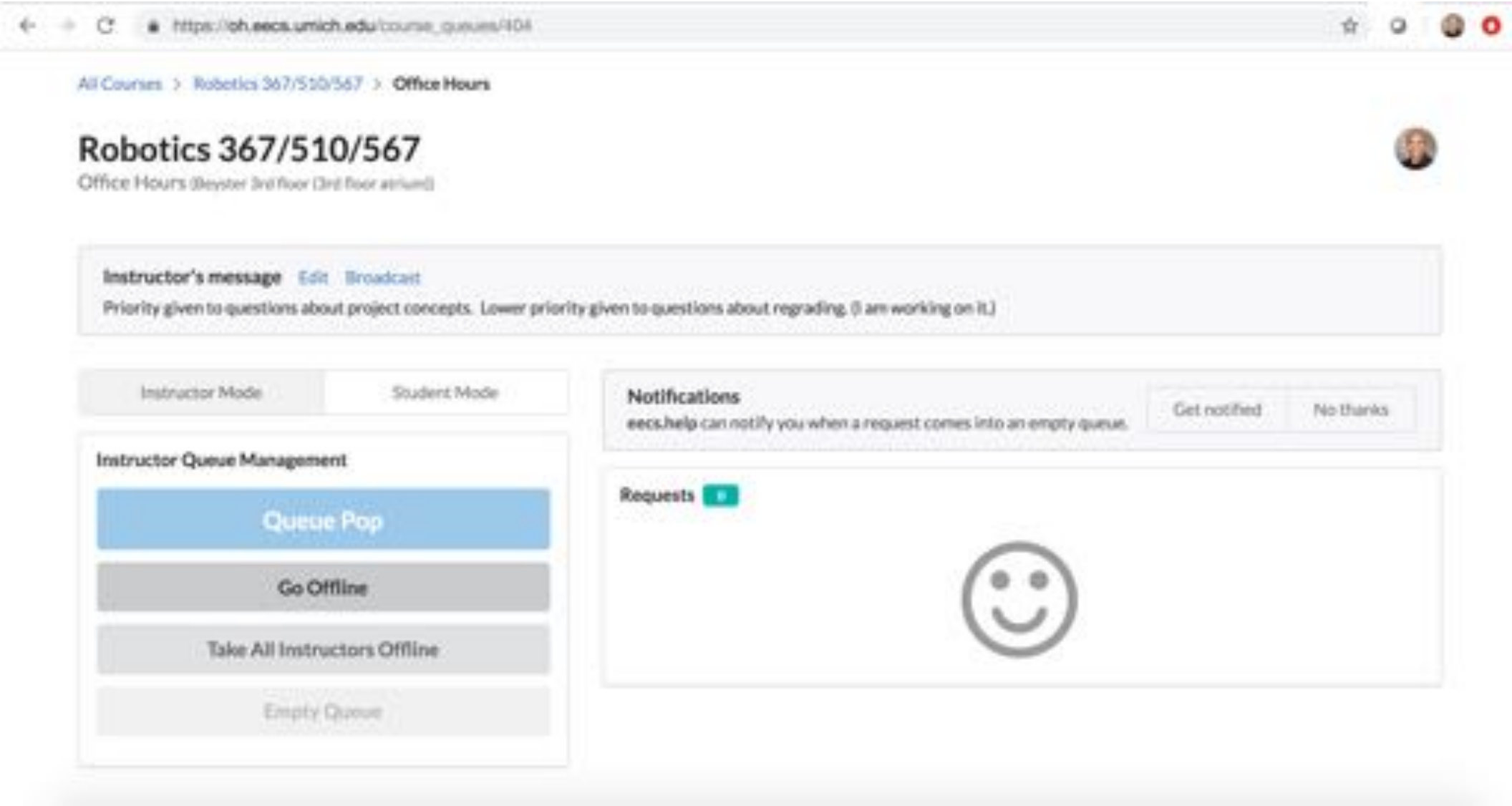
| Date | Topic | Reading | Project |
|---|---|---|---|
| Sep 4 | **Initialization**: Course overview, Robotics roadmap, Path planning quick start | Spong Ch.1<br>Corke Ch.1 | Setup git repository |
| | **What is a robot?**: Brief history and definitions for robotics | | Out: **Path Planning** |
| Sep 6 | 367 Lab: Git-ing started with git, JavaScript, and KinEval | | |
| | Week 2 | | |
| Sep 9 | **Path Planning**: DFS, BFS, A-star, Greedy best first<br>**Addendum: DFS example** | **Wikipedia** | |
| Sep 11 | **JavaScript and git tutorial**: Heap sort example<br>**Addendum: coding workflow** | Crockford,<br>**HTML Sandbox**,<br>**hello.html** (source),<br>**JavaScript by Example** (source),<br>**hello_anim** (source),<br>**hello_anim_text** (source) | |
| Sep 13 | 367 Lab: search_canvas.html code overview | | |
| | Week 3 | | |
| Sep 16 | **Course meeting cancelled** - off-site meeting | | |
| Sep | | | Due: Path Planning |

Slack
via web



Slack OS X app



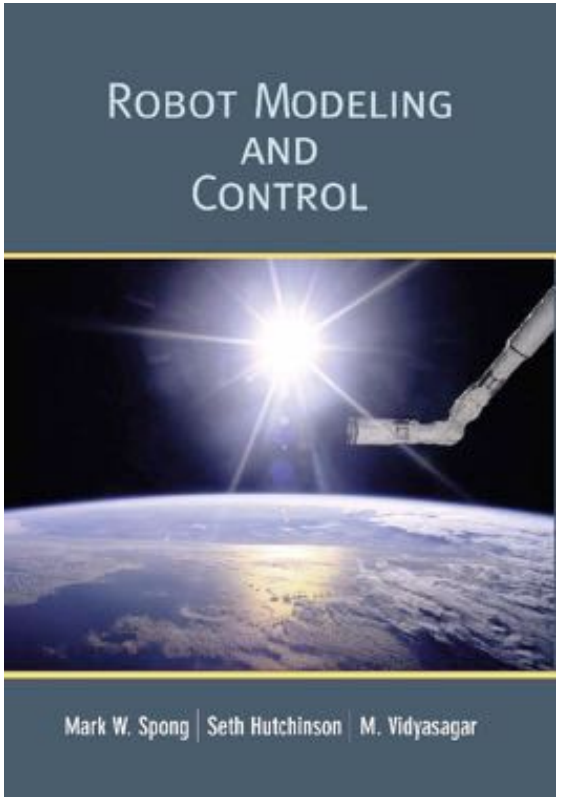


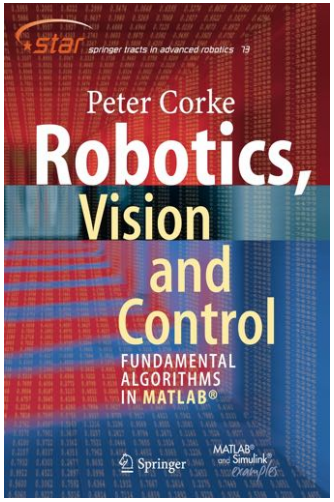
# Course Structure

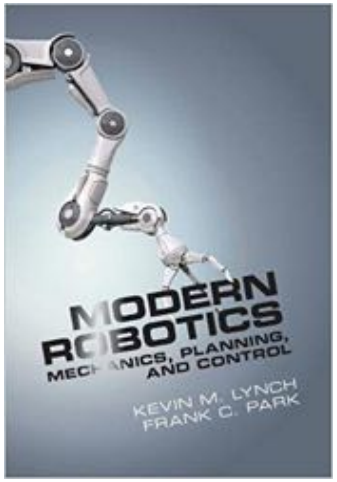
- Autonomous robot modeling and control
  - Objective: Give you the computational skills (and code) to model and control any mobile manipulator
- Project-focused class
  - 7 individual projects: from single joint control up to articulated motion planning
- Computing-friendly introduction to robotics: projects in JavaScript

# Course Textbook

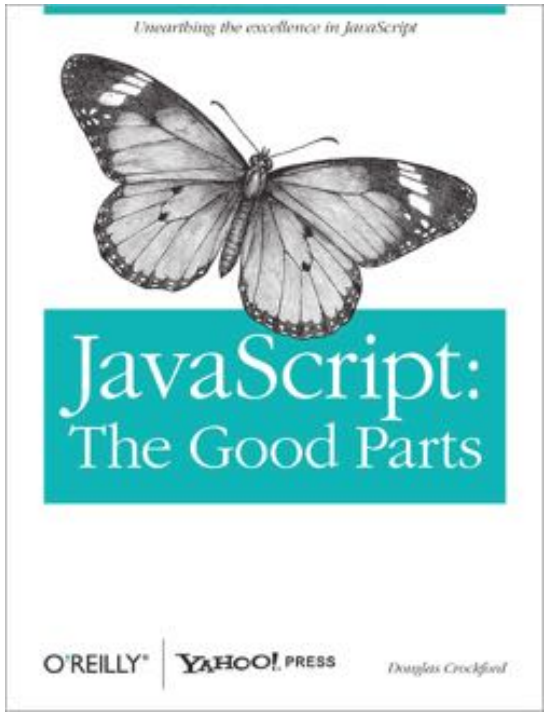
- Robot Modeling and Control (Spong, Hutchinson, Vidyasagar)
- Alternative: Robotics, Vision, and Control (Corke)
- Suggested but unsupported: Modern Robotics (Lynch and Park)
- In-depth coverage of concepts and math contained in textbooks
- Additional handouts and links will appear on the course website

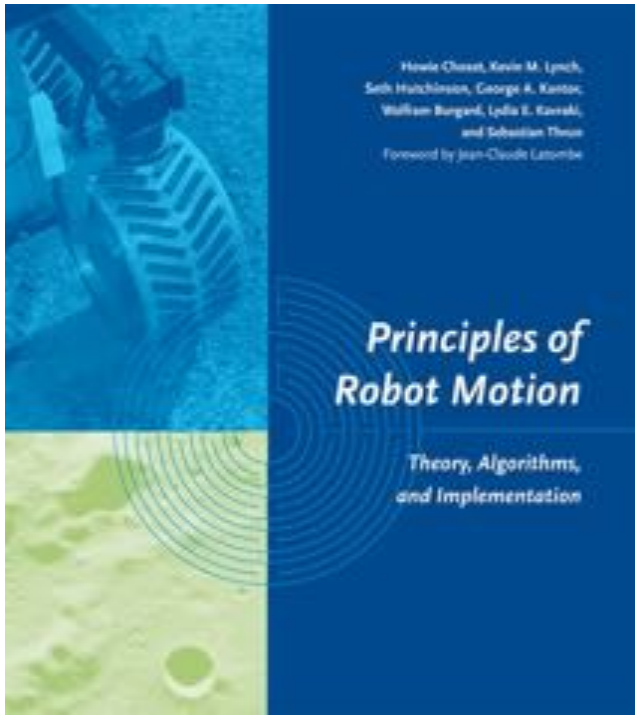


# Optional reading



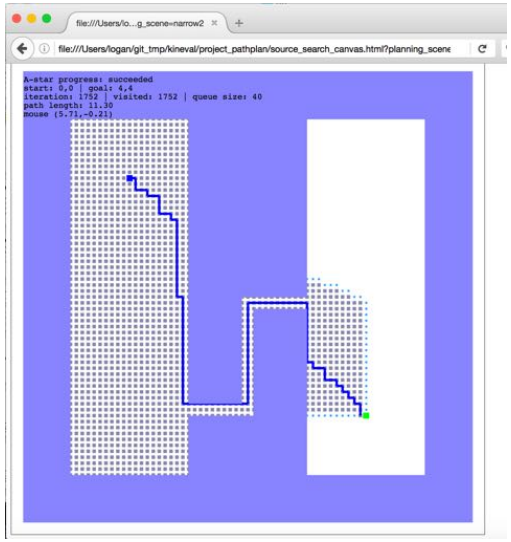

# Projects

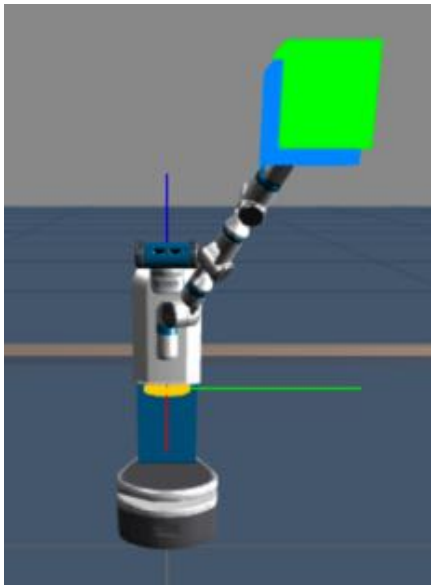
- Projects implemented in JavaScript/HTML5 using KinEval stencil
  - Projects submitted and tracked through git (gitlab|github|bitbucket)
  - Instructor (ohseejay|ocj) needs admin access
- 7 projects
  - 6 Programming, 1 Written/Oral
- Grading: projects are broken down into features that are “checked”
  - points are earned through successful implementation of features

# Projects

- **Path Planning** A-star search in 2D world
- **Pendularm** physical simulation and PID control of 1 DoF robot
- **Forward Kinematics** convert robot configuration to 3D space
- **Dance Contest** control of robot joints to do a dance
- **Inverse Kinematics** control gripper of a robot to reach a point in 3D
- **Motion Planning** collision-free planning over robot configurations
- **Best Use of Robotics** what will you do with all of this power?

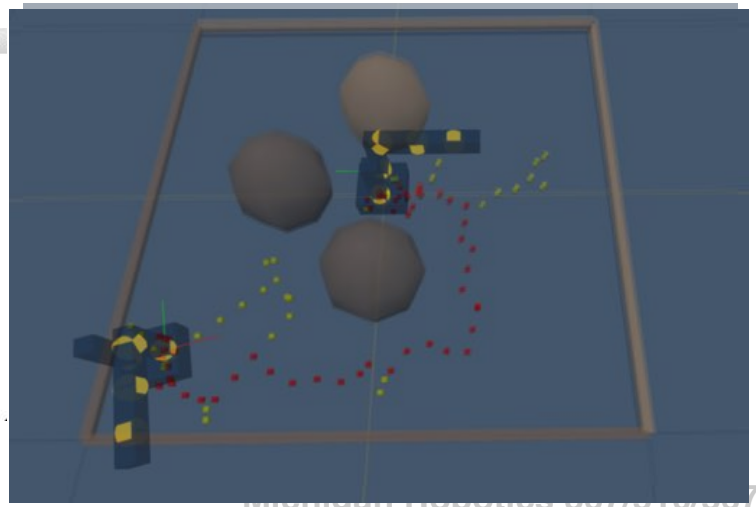
- **Path Planning**



- **FK/IK**

- **Pendularm**



- **RRT**

Will you work with a real robot?

Will you work with a real robot?

Yes, at least once using rosbridge/ROS

# Grading Summary

EECS 367: Introduction to Autonomous Robotics

- 7 projects (12 points each)
- 5 quizzes (4 points each)

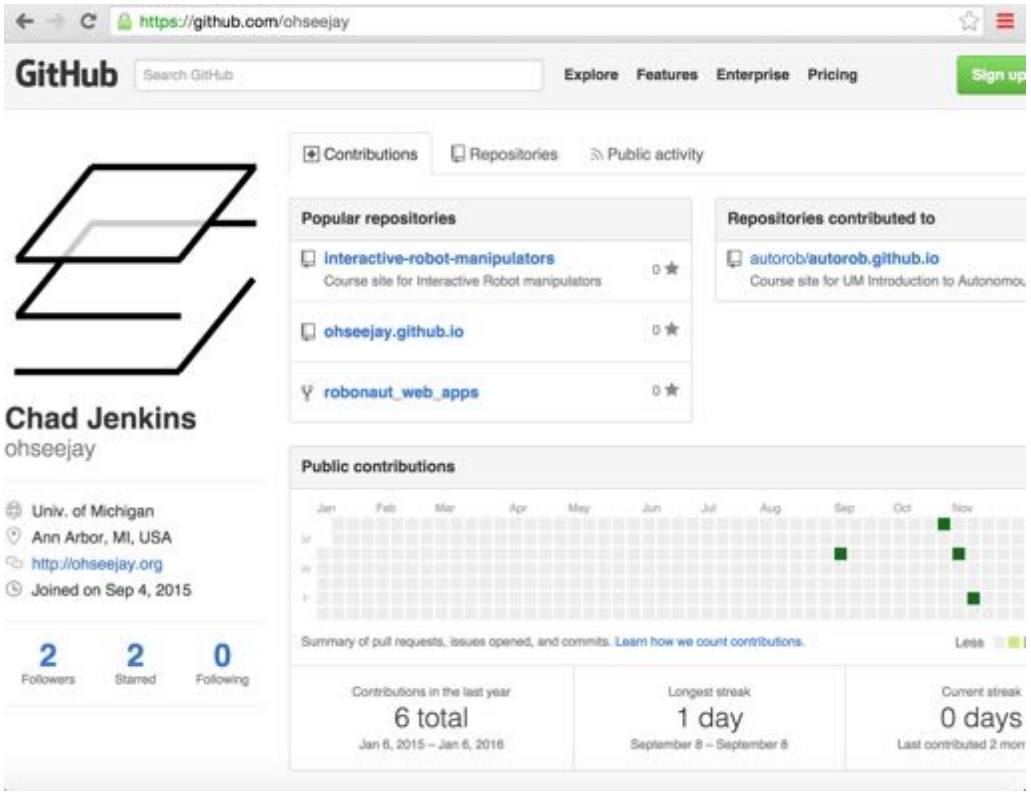
**A**: 93+ points
**B**: 83+ points
**C**: 73+ points

ME/EECS 567 ROB 510: Robot Kinematics and Dynamics

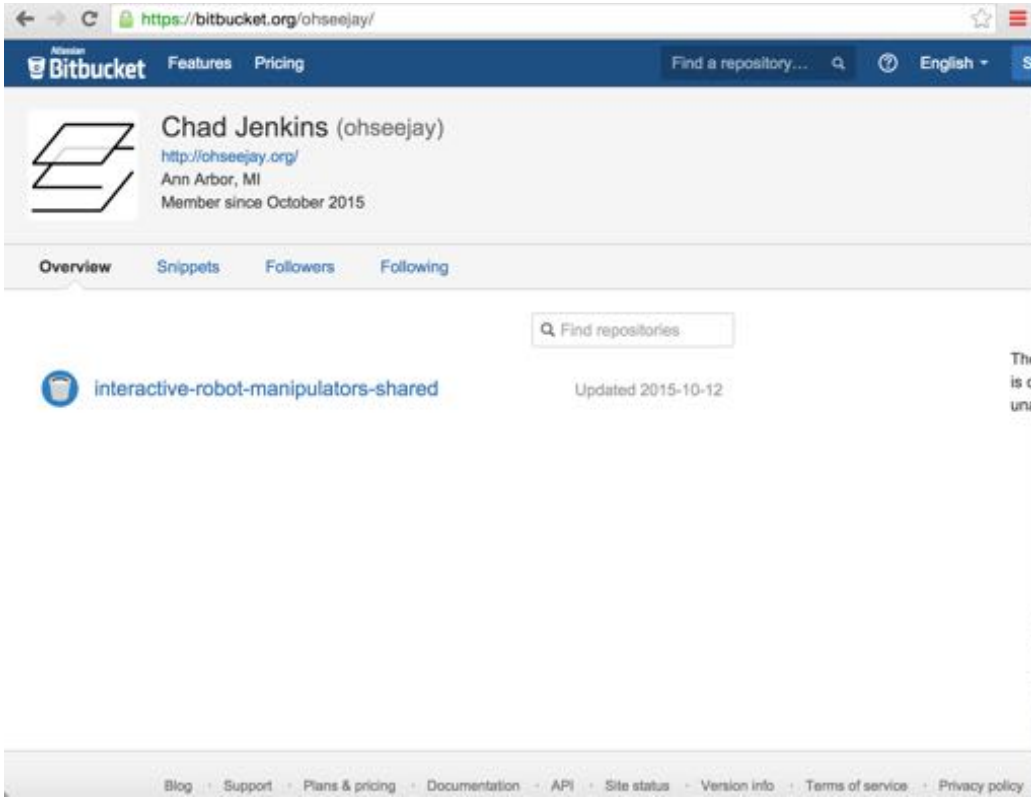
- 7 projects (18 points each)
- 5 quizzes (4 points each)
- Advanced features (4 points)

**A**: 135+ points
**B**: 120+ points
**C**: 105+ points

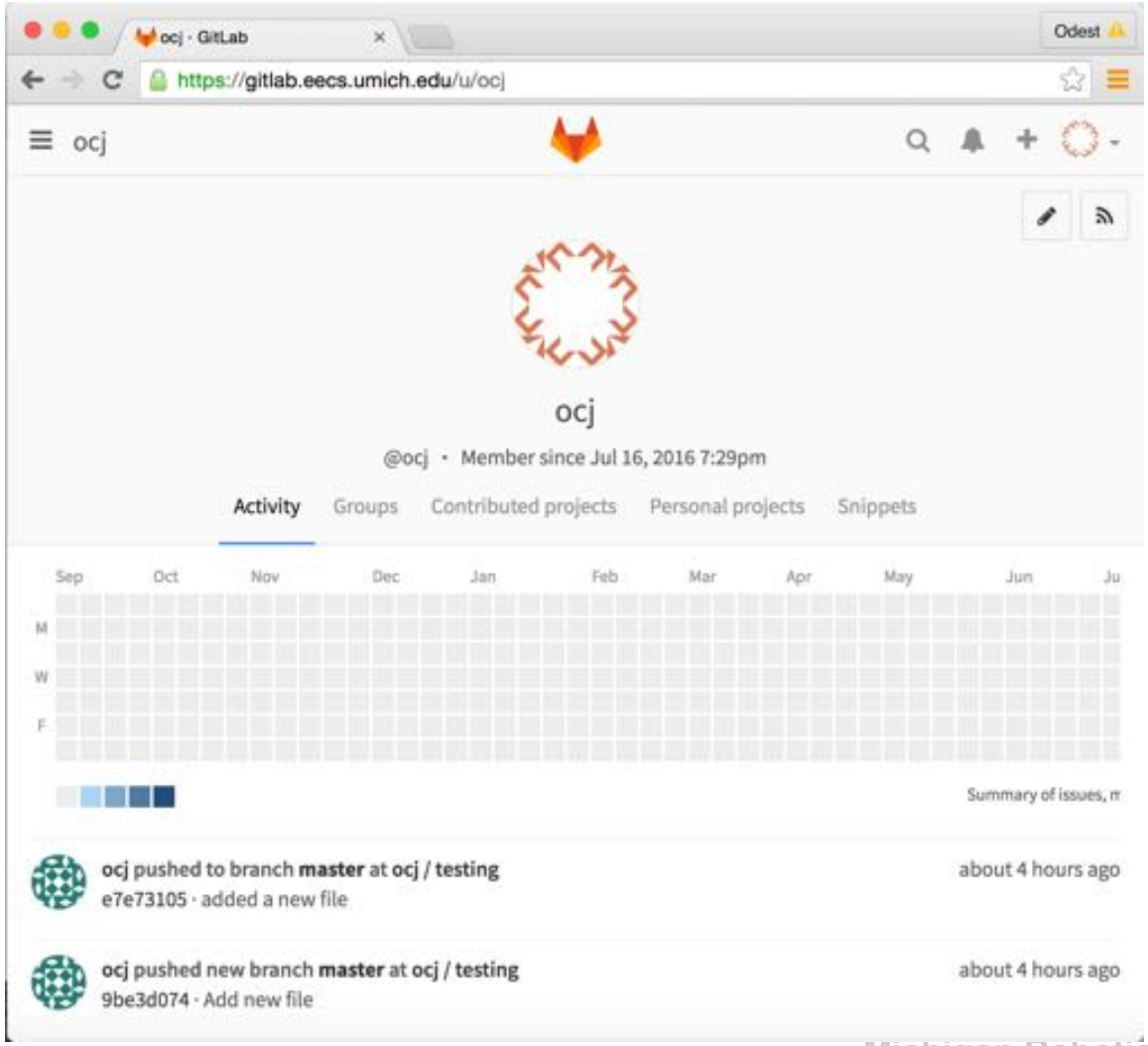
github.com/ohseejay




bitbucket.org/ohseejay



gitlab.eecs.umich.edu/ocj



# Collaboration Policy

- All work submitted must be your own
  - All code submitted must comply with Michigan Honor License
- No code can be communicated, including verbally
  - Explicit use of external sources must be clearly cited
  - Repositories must be **private** for proper compliance
- Free flow of discussion and ideas is encouraged

# Michigan Honor License

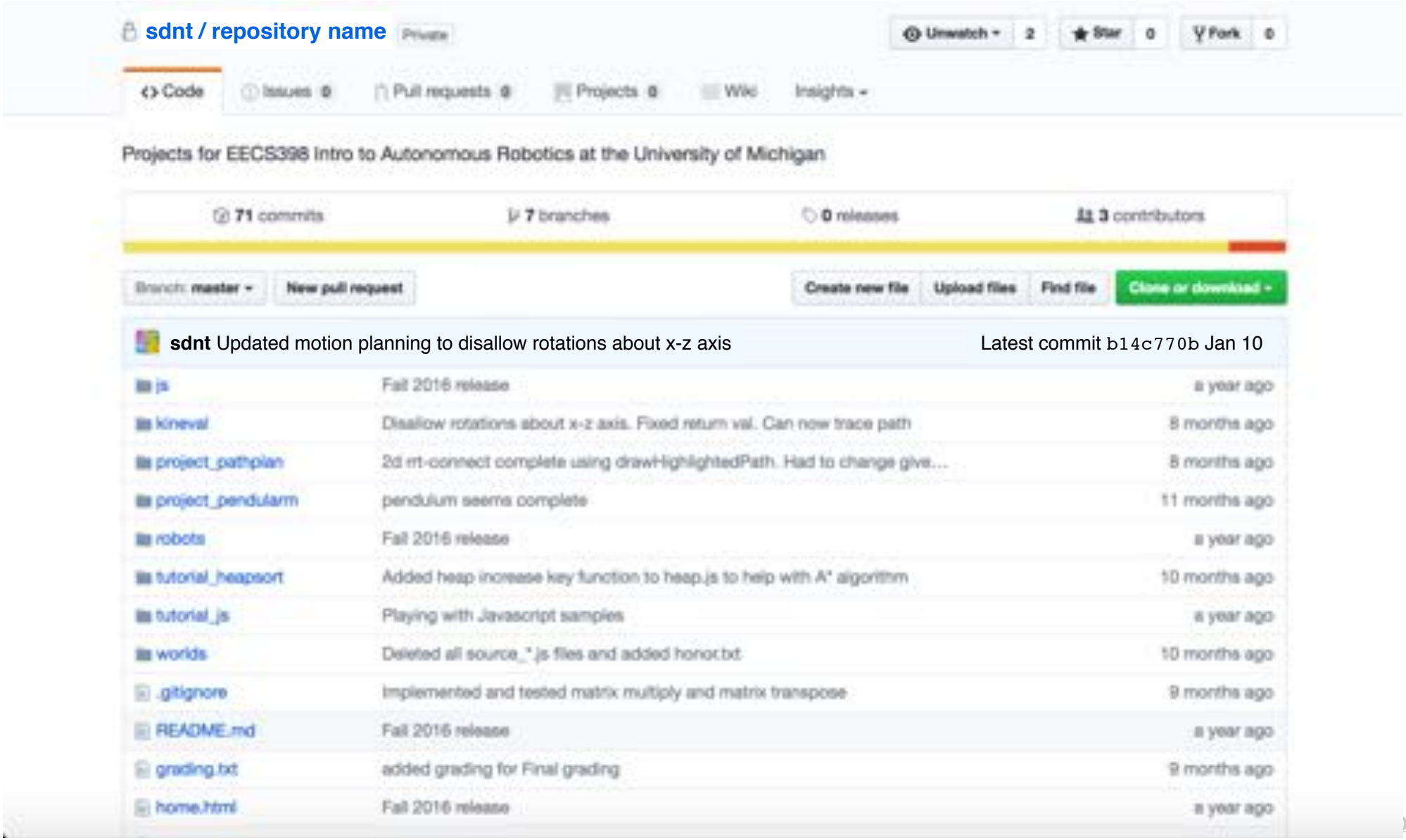
- 3-Clause BSD License + Michigan Honor Code + proper citation
- Assert the compliance of your code with the MHL
  - Append your name to the end of LICENSE in your repository
- Submitted code will not be graded without asserting LICENSE

# Late Policy

- Projects submitted after deadline may not be graded (zero credit)
- If a late submission is allowed, it can receive at most
  - 80% credit if pushed within 2 weeks of the deadline
  - 60% credit if pushed within 4 weeks of the deadline
  - 50% credit if pushed anytime before final grading

# Regrading policy

- Projects features are graded with:
  - “CHECK” (sufficiently completed)
  - “DUE” (insufficiently completed)
  - “PENDING” (not due yet)
- A project feature can be regraded for partial credit for at most
  - 80% credit if pushed within 2 weeks of the last returned grading
  - 60% credit if pushed anytime before final grading


sdnt / repository name
Private
Unwatch 2
Star 0
Fork 0
Code
Issues 0
Pull requests 0
Projects 0
Wiki
Insights
Projects for EECS398 Intro to Autonomous Robotics at the University of Michigan
71 commits
7 branches
0 releases
3 contributors
Branch: master
New pull request
Create new file
Upload files
Find file
Clone or download
sdnt Updated motion planning to disallow rotations about x-z axis
Latest commit b14c770b Jan 10
js Fall 2016 release a year ago
kineval Disallow rotations about x-z axis. Fixed return val. Can now trace path 8 months ago
project_pathplan 2d rrt-connect complete using drawHighlightedPath. Had to change give... 8 months ago
project_pendularm pendulum seems complete 11 months ago
robots Fall 2016 release a year ago
tutorial_heapsort Added heap increase key function to heap.js to help with A* algorithm 10 months ago
tutorial_js Playing with Javascript samples a year ago
worlds Deleted all source_*.js files and added honor.txt 10 months ago
.gitignore Implemented and tested matrix multiply and matrix transpose 9 months ago
README.md Fall 2016 release a year ago
grading.txt added grading for Final grading 9 months ago
home.html Fall 2016 release a year ago

Branch: master ▾ repository name / grading.txt

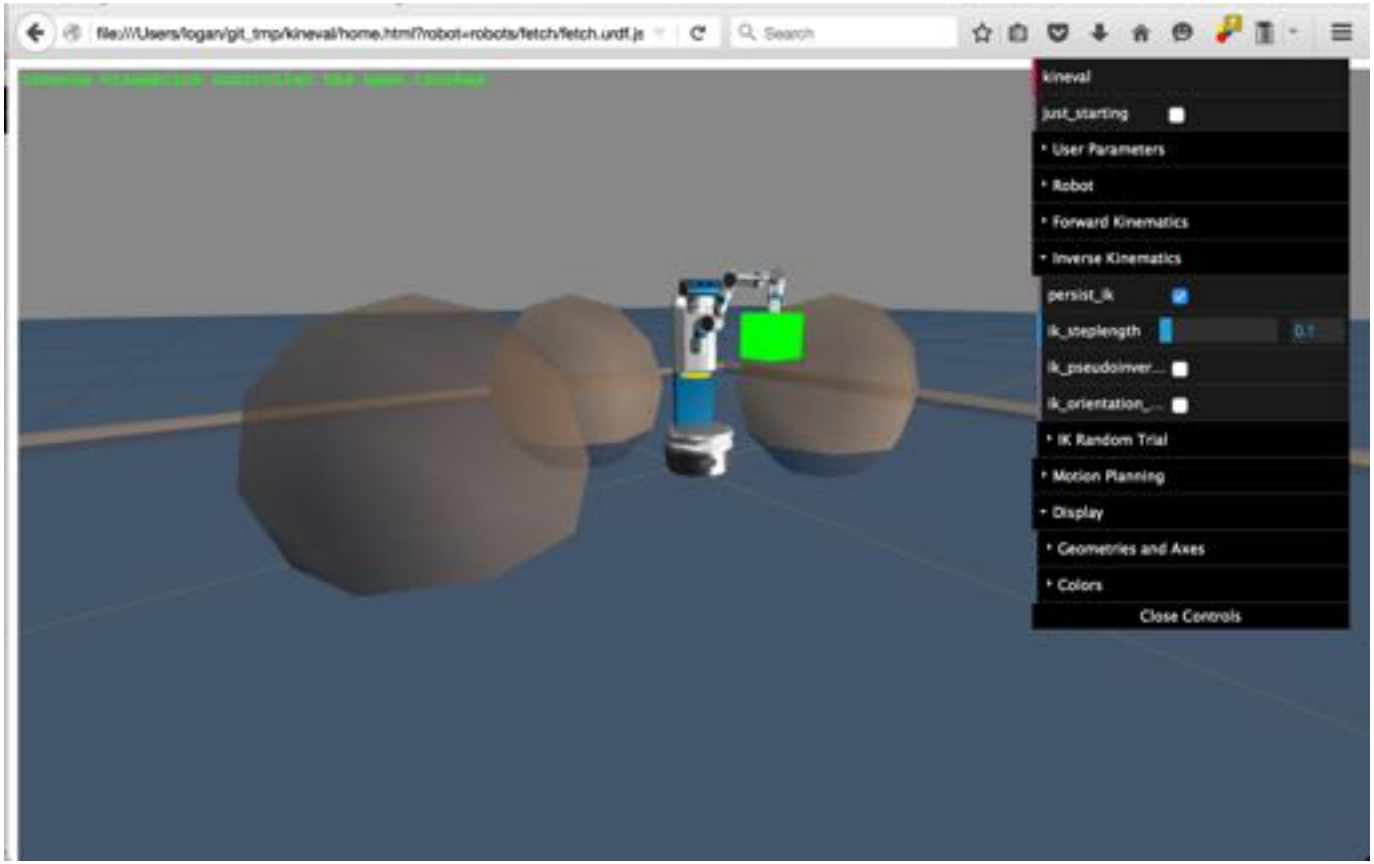
odestcj added grading for Final grading

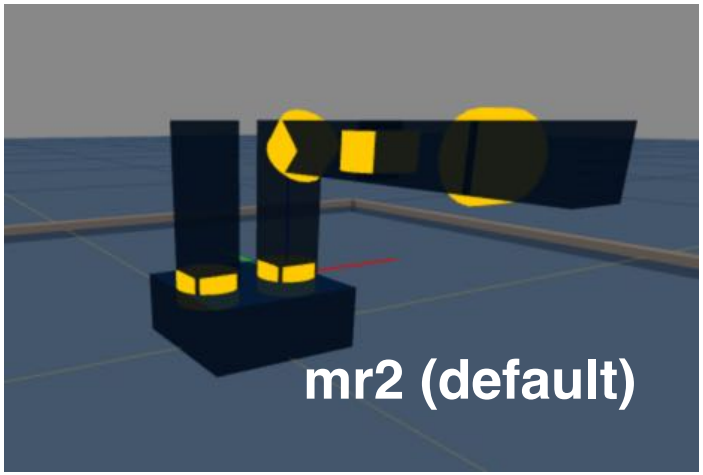
1 contributor

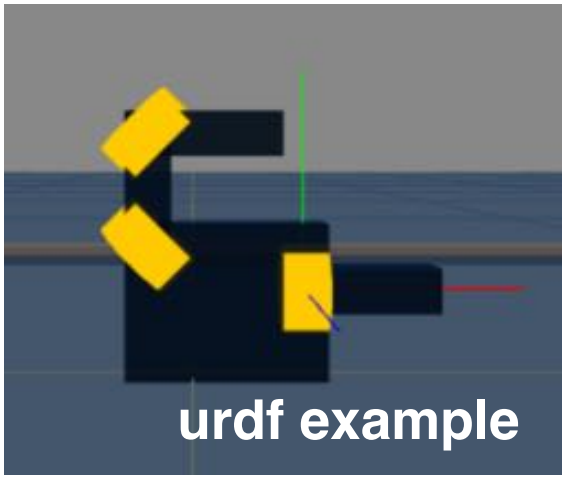
60 lines (58 sloc) 2.26 KB Raw Blame

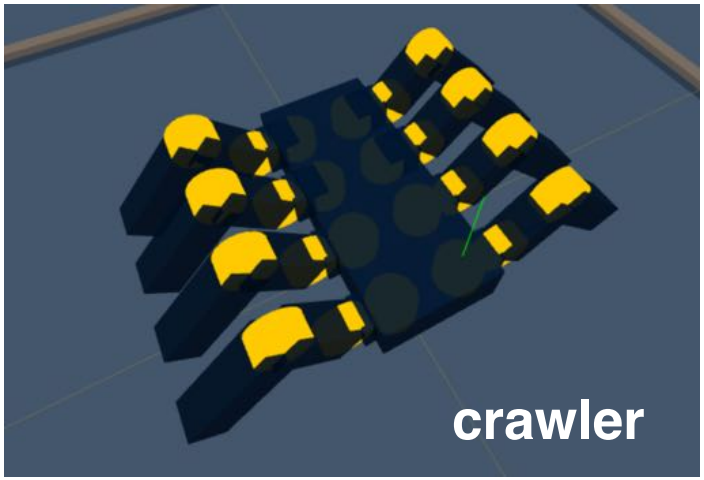
```
53: student name
EECS 398-004 F16
Assignment 1 feature 1 (4.00/4): PathPlan_Heap: CHECK
  comment: good work
Assignment 1 feature 2 (8.00/8): PathPlan_AStar: CHECK
  comment: good work
Assignment 2 feature 3 (4.00/4): Pendularm_Euler: CHECK
  error: Euler integrator is incorrect
  comment: why is this integrator stable?
  error: integration of dynamics is incorrect
  comment: double check computation of pendulum acceleration
  regrade: servo converges after several oscillations, borderline control performance
Assignment 2 feature 4 (4.00/4): Pendularm_VelocityVerlet: CHECK
  error: integration of dynamics is incorrect
  comment: double check computation of pendulum acceleration
  regrade: working
Assignment 2 feature 5 (4.00/4): Pendularm_PID: CHECK
  error: integration of dynamics is incorrect
  comment: modulo correction is not needed
Assignment 3 feature 6 (2.00/2): FK_MatrixRoutines: CHECK
```

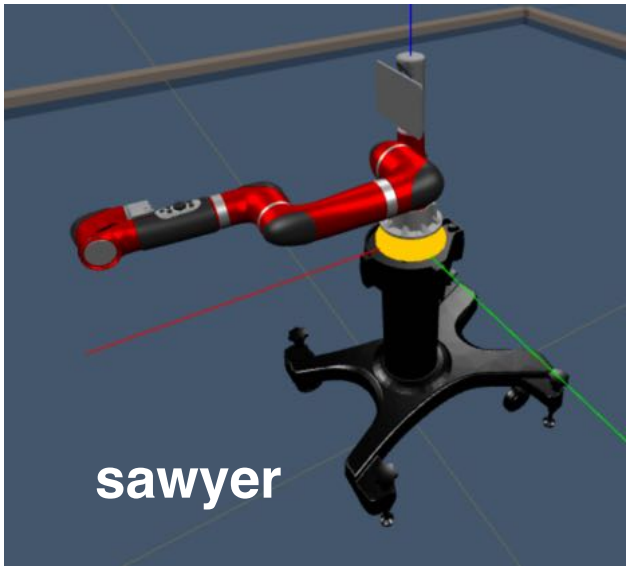
# KinEval code stencil

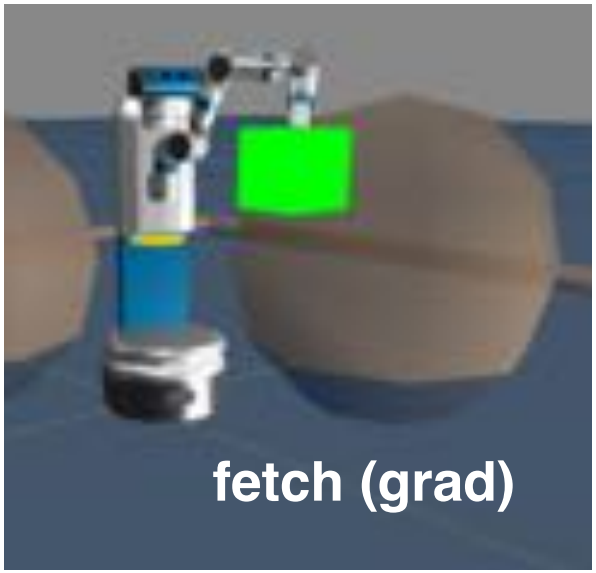


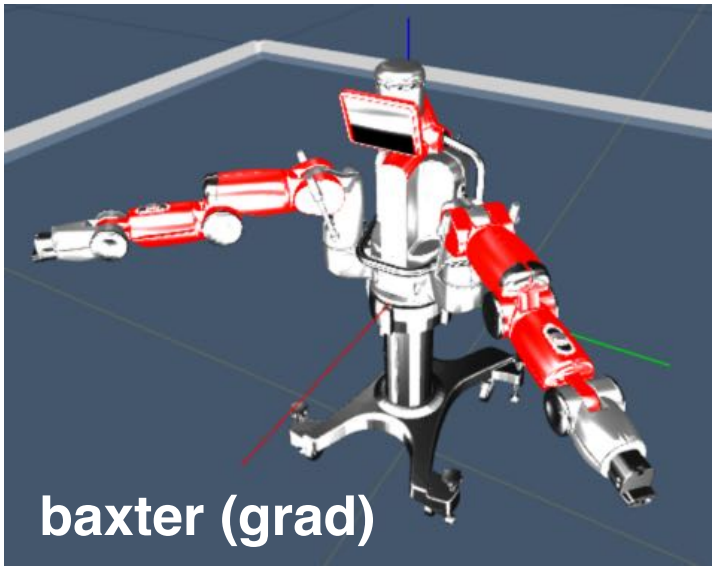

mr2 (default)


urdf example


crawler

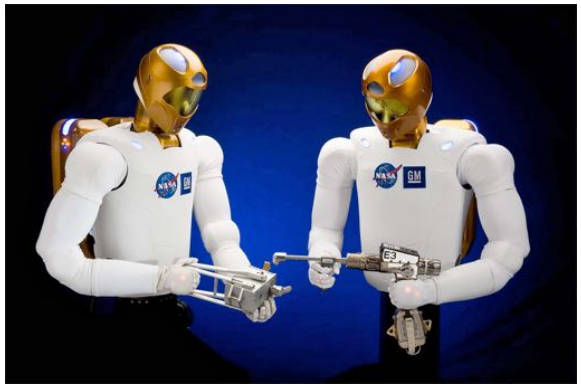

sawyer


fetch (grad)


baxter (grad)

# Maybe also?

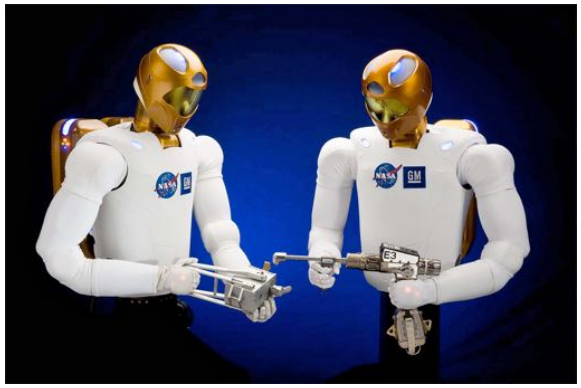

KINOVA
MOVO BETA
A MOBILE
MANIPULATOR
PLATFORM
kinovaMOVO.com

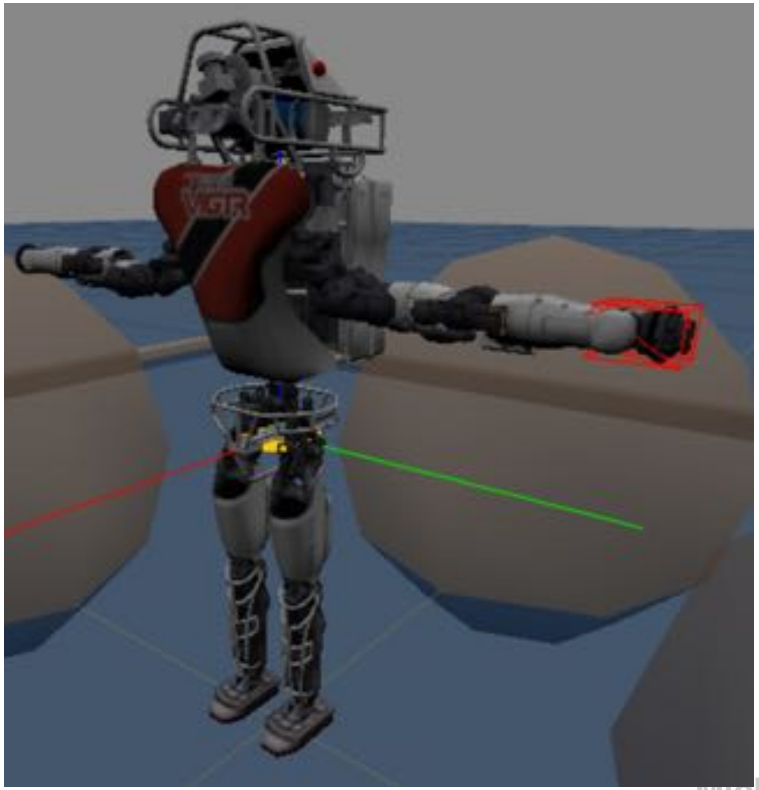



Boston Dyn


KINOVA
MOVO BETA
A MOBILE
MANIPULATOR
PLATFORM
kinovaMOVO.com

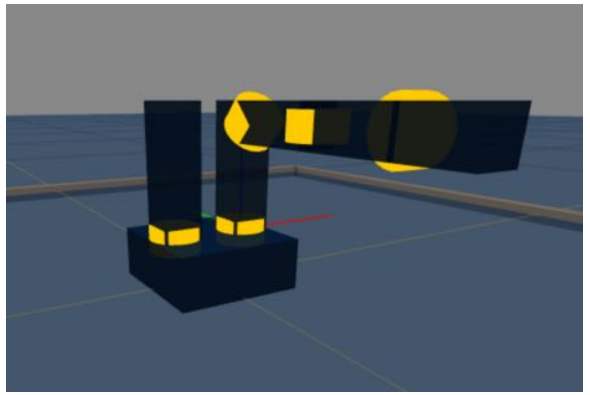
# Maybe also?





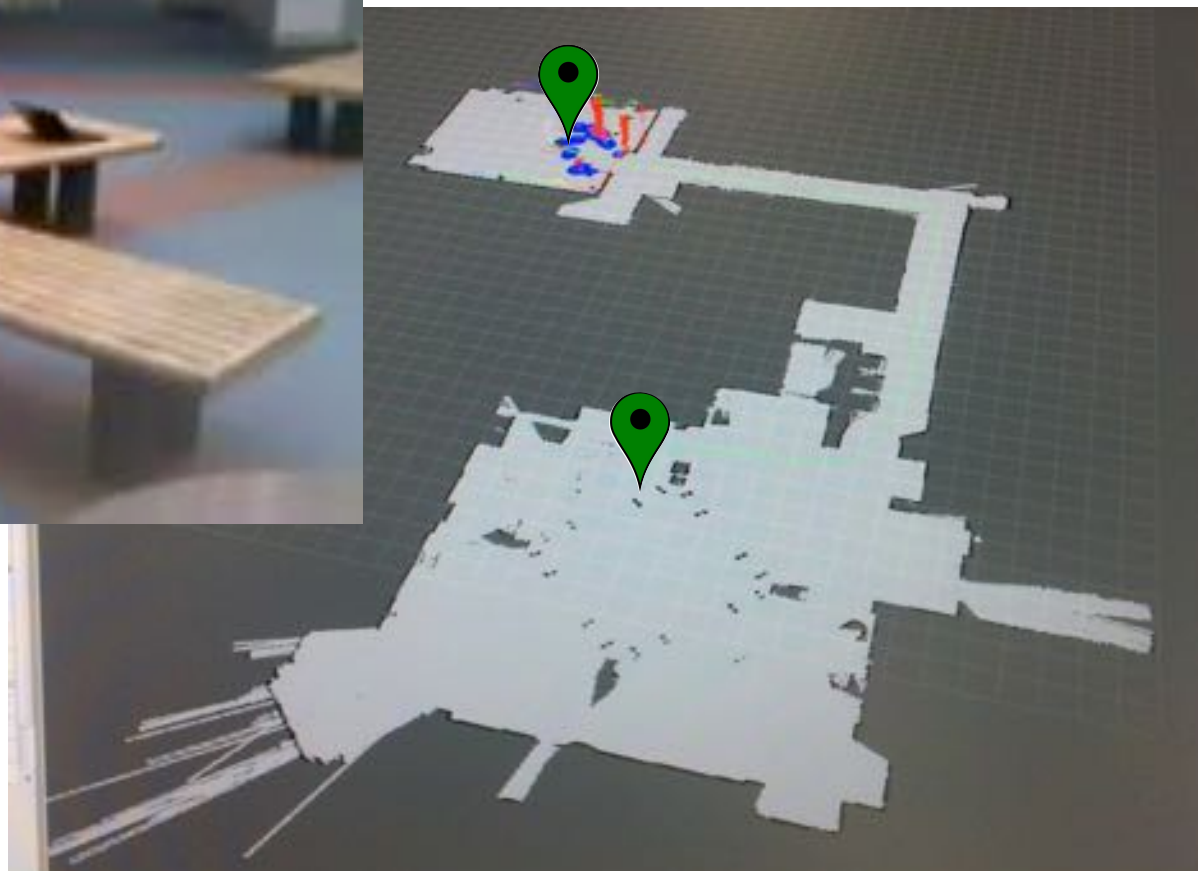
# KinEval code stencil

- Code stencil for AutoRob projects in 3D
- Uses threejs 3D rendering library and WebGL
- URDF-like robot description
- Usable, but not perfect, camera and UI controls
- AABB collision detection provided for planning
- Warning: professor-level coding



# Let’s start with base navigation

(Lecture 2 foreshadowing)

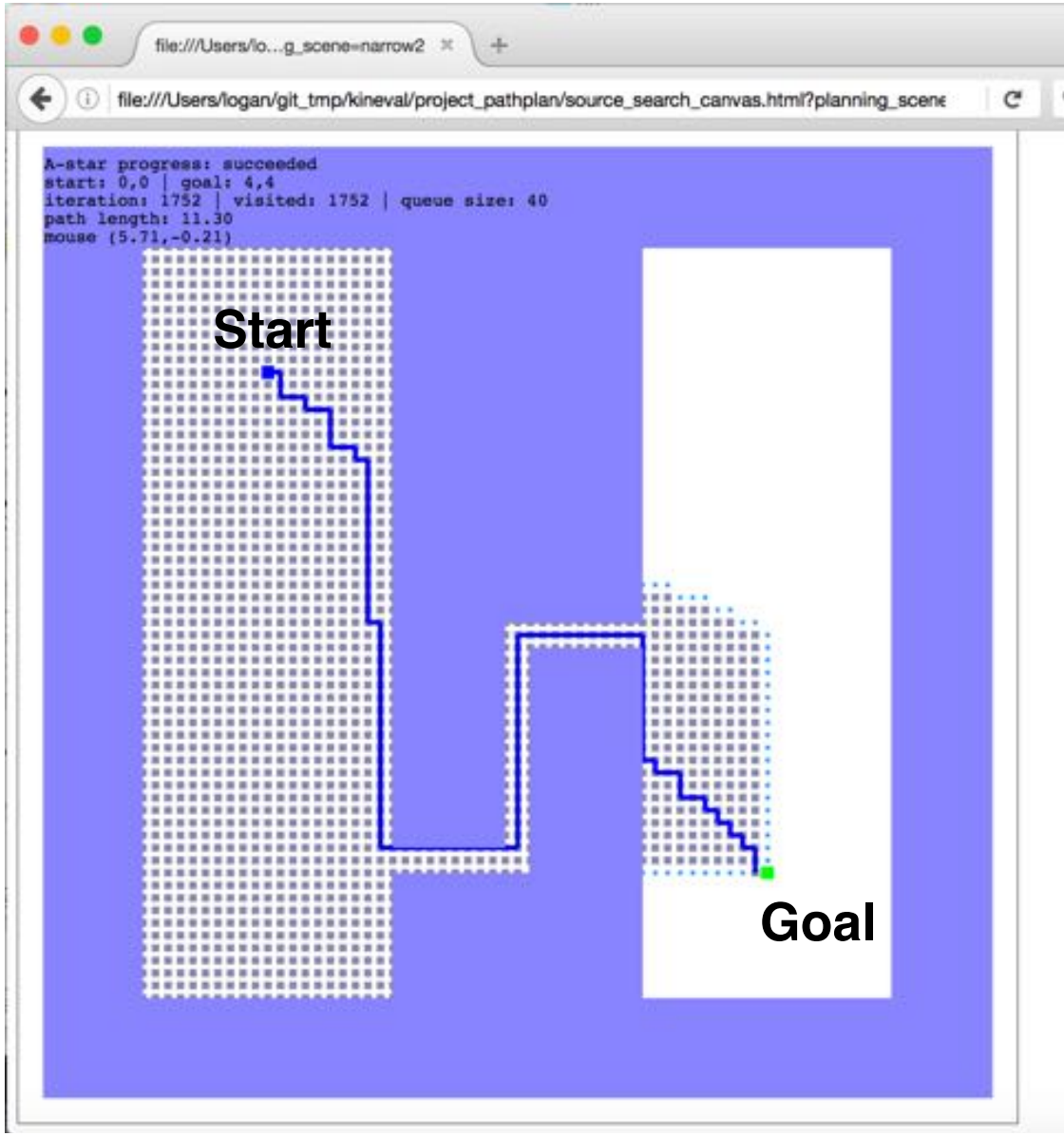


Let's start with base navigation

How to get from Location A to Location B?

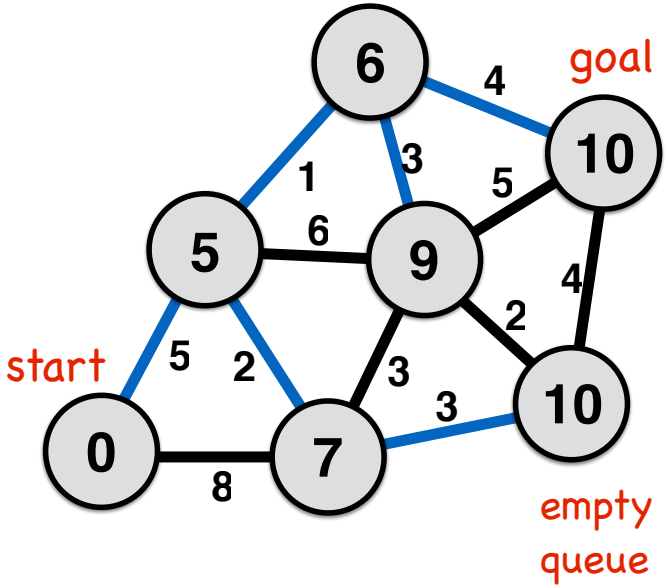
# Project 1:
# 2D Path Planning

- A-star algorithm for search in a given 2D world
- Implement in JavaScript/ HTML5
- Heap data structure for priority queue
- Grad: DFS, BFS, Greedy
- Submit through your git repository



## Dijkstra shortest path algorithm

all nodes ← {$dist_{start}$ ← infinity, $parent_{start}$ ← none, $visited_{start}$ ← false}
start_node ← {$dist_{start}$ ← 0, $parent_{start}$ ← none, $visited_{start}$ ← true}
visit_queue ← start_node
    **while** visit_queue != empty ~~&& current_node != goal~~
        cur_node ← min_distance(visit_queue)
        $visited_{cur\_node}$ ← true
        **for** each nbr in not_visited(adjacent(cur_node))
            enqueue(nbr to visit_queue)
            **if** $dist_{nbr}$ > $dist_{cur\_node}$ + distance(nbr,cur_node)
                $parent_{nbr}$ ← current_node
                $dist_{nbr}$ ← $dist_{cur\_node}$ + distance(nbr,cur_node)
            **end** if
        **end** for loop
    **end** while loop
output ← parent, distance



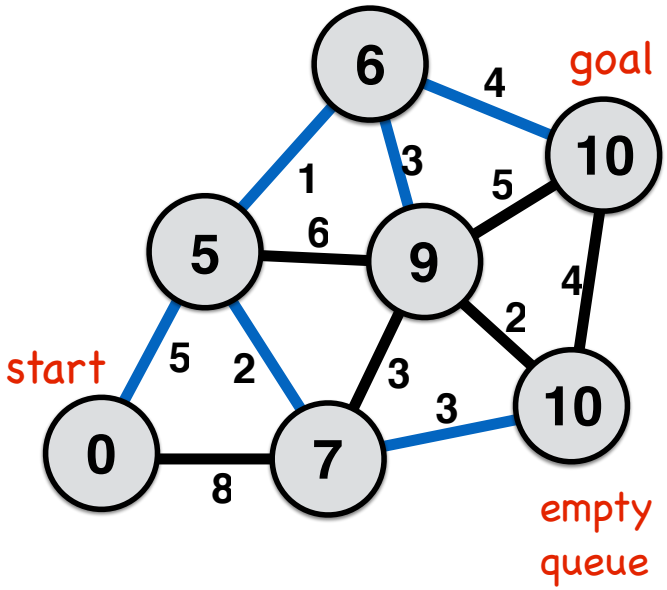
## A-star shortest path algorithm

all nodes ← {$dist_{start}$ ← infinity, $parent_{start}$ ← none, $visited_{start}$ ← false}
start_node ← {$dist_{start}$ ← 0, $parent_{start}$ ← none, $visited_{start}$ ← true}
visit_queue ← start_node
    **while** (visit_queue != empty) && current_node != goal
        dequeue: cur_node ← f_score(visit_queue)
        $visited_{cur\_node}$ ← true
        **for** each nbr in not_visited(adjacent(cur_node))
            enqueue: nbr to visit_queue
            **if** $dist_{nbr}$ > $dist_{cur\_node}$ + distance(nbr,cur_node)
                $parent_{nbr}$ ← current_node
                $dist_{nbr}$ ← $dist_{cur\_node}$ + distance(nbr,cur_node)
                f_score ← $distance_{nbr}$ + $line\_distance_{nbr,goal}$
            **end** if
        **end** for loop
    **end** while loop
output ← parent, distance

implement min binary heap for priority queue

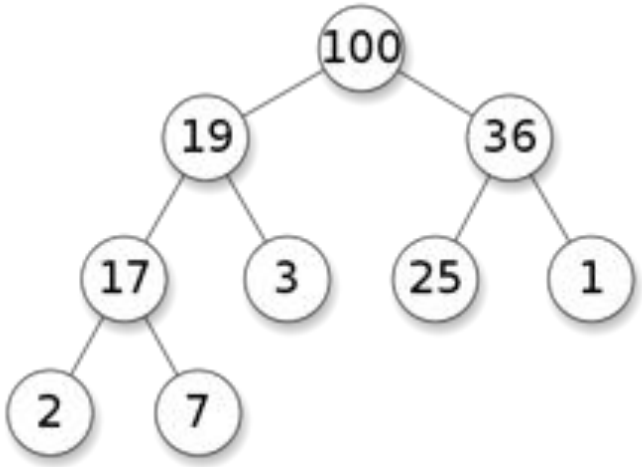
g_score: distance along path

h_score: best distance to goal

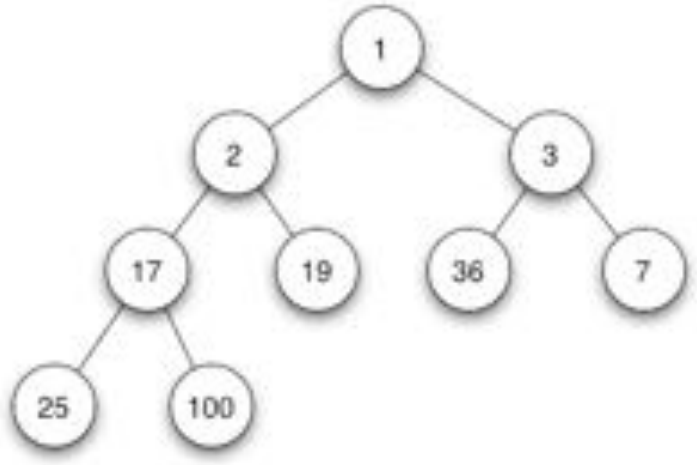


# Heaps

A heap is a tree-based data structure satisfying the heap property:
every element is greater (or less) than its children



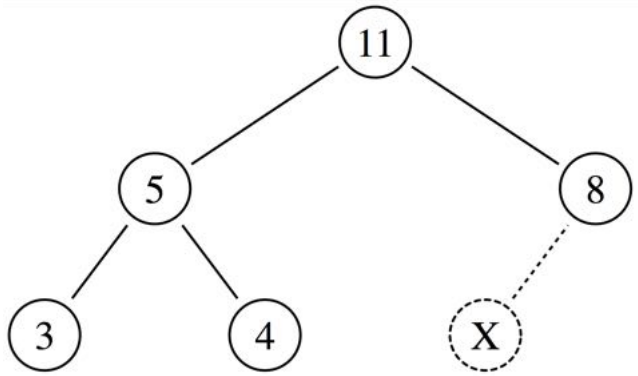

max heap



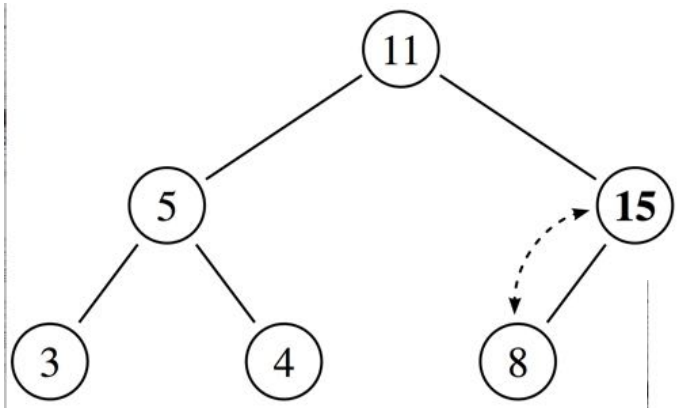

min heap

# Heap operations: Insert

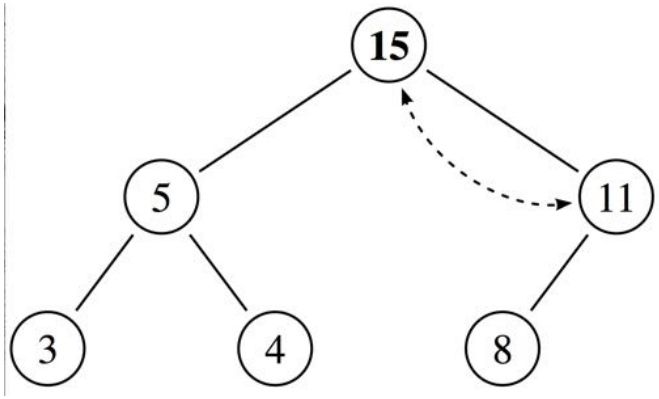
1) add new element to end of tree




2) swap with parent

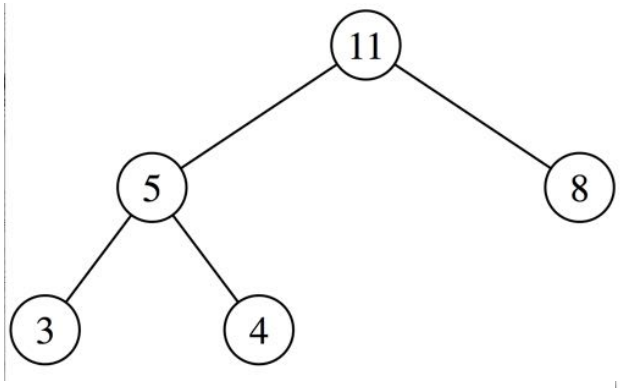



3) until heaped, do (2)

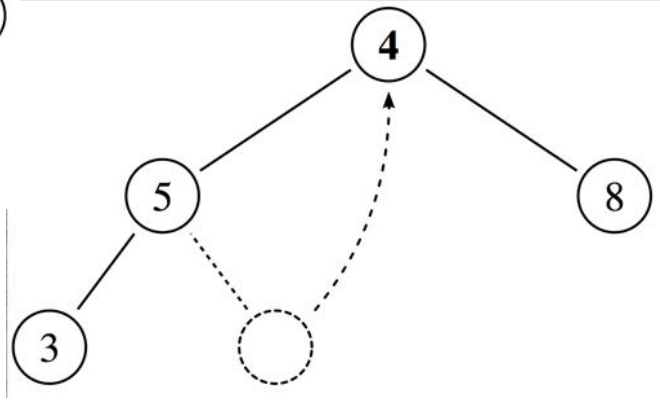


# Heap operations: Extract

1) extract root element

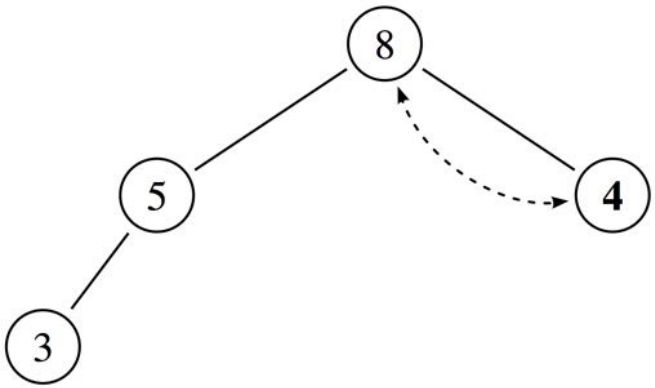



2) put last element at root




3) swap with smaller child

4) until heaped, do (3)



**heapsort.html**

```
<!DOCTYPE html>
<html>
  <head>
    <title>Heap Sort by Chad Jenkins</title>
  </head>
   <body onload=startMeUp()>
       <a href="http://ohseejay.org"><h1>My Heap Sort</h1></a>
       <canvas width=900 height=200 id="myCanvas"></canvas>
       <div id="output">hello world</div>
  </body>
</html>

<script>
function startMeUp() {
    .. executable code to perform heapsort; will depend on heap.js …
}
</script>
```

HTML5 document will invoke JavaScript function at start

# git basics

- Create a git repository from gitlab, github, or bitbucket website
- Install git on your machine
  - https://git-scm.com/book/en/v2/Getting-Started-Installing-Git
  - OSX: https://code.google.com/p/git-osx-installer/

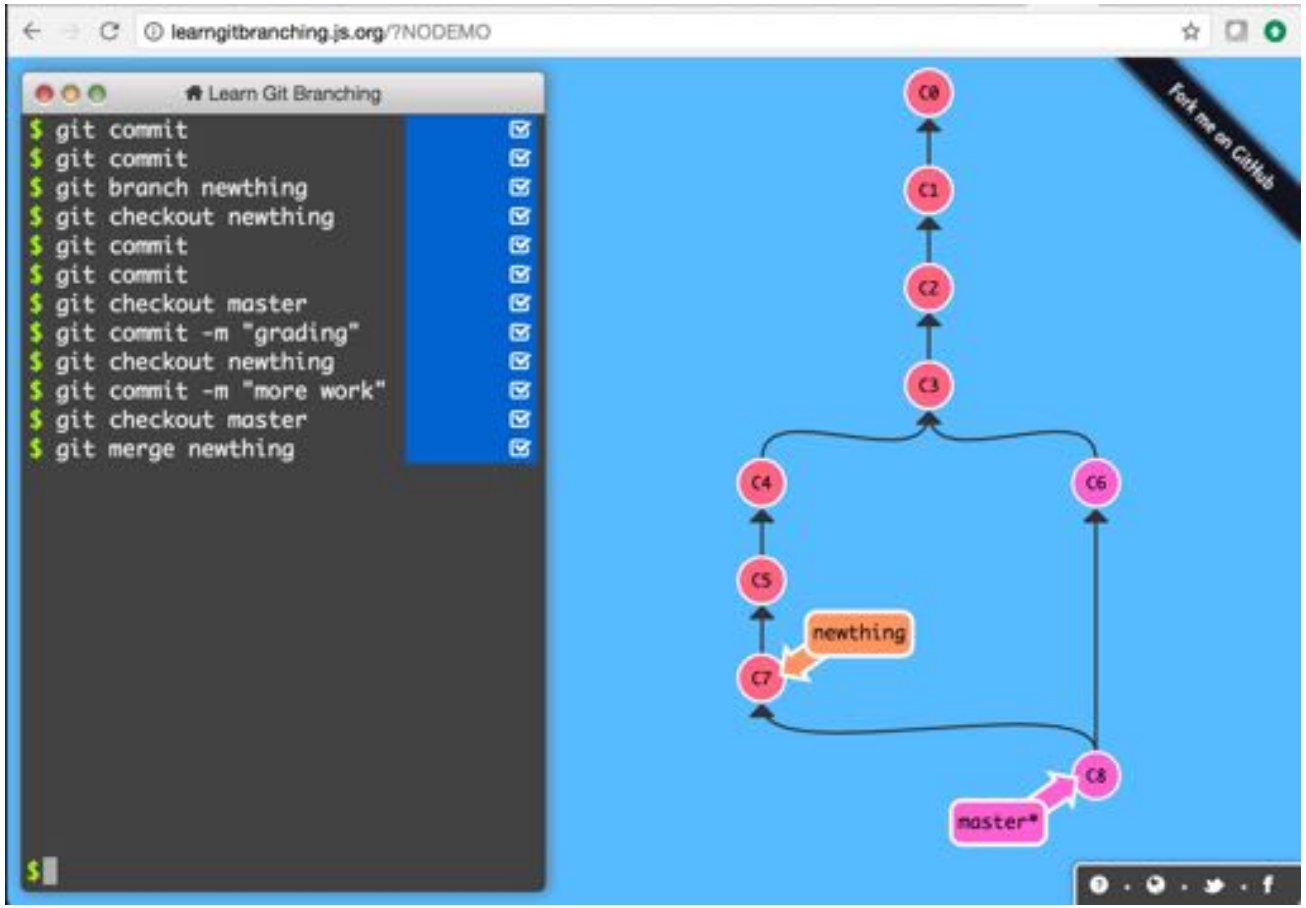
# git basics

- create a local copy of a repository: `git clone <repo url>`
- add files to a repository: `git add <file listing>`
- commit changes to local repository: `git commit -a -m "<msg>"`
- push local changes to a remote repository: `git push`
- pull remote changes to a local repository: `git fetch` or `git pull`
- create a code branch in a repository: `git branch <branch name>`
- checkout a code branch from a repository : `git branch <branch name>`
- merge branches in a repository:

```
git checkout <branch name>

git merge <other branch name>
```

# Highly recommended tutorial




http://learngitbranching.js.org/

# Can I enroll in this class?

- Most likely!  Probably yes!
- We will try to enroll all who want to take the course
- Enrollment issues can be discussed at office hours or lab section
  - Friday 2:30-3:30pm in EECS 1500
  - Monday 3-5pm and Tuesday 1-3pm in Beyster 3644

# What I need from you **now**

- Accept invitation to the course discussion channel (coming tonight)
  - https://autorob.slack.com
- Install git and setup your working environment
  - create a git repository: https://gitlab.eecs.umich.edu/
  - ensure you can clone, commit, and push files to your repository
- Over the discussion channel, send me:
  - informal introduction confirming your name, email, and enrollment
  - pointer to your git repo for the course

# What I need from you **soon**

- Get started on Assignment 1 (Path Planning)
  - Clone kineval-stencil repository (release this weekend)
  - Study examples in tutorial_js subdirectory
  - Complete tutorial_heapsort (noted with “STENCIL” in files)
  - Complete project_pathplan (concepts covered next week)

# What is a robot?

(be sure to ready over Lecture 0)