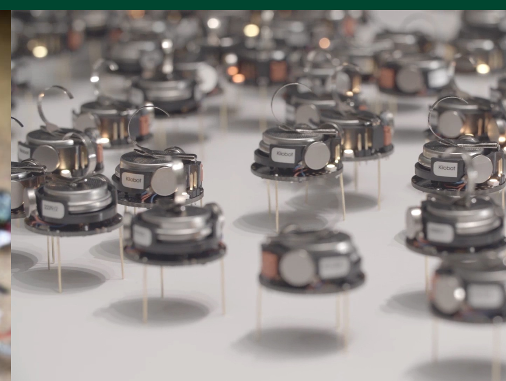
ECE693H, Spring 2025:
Multi-robot System Design

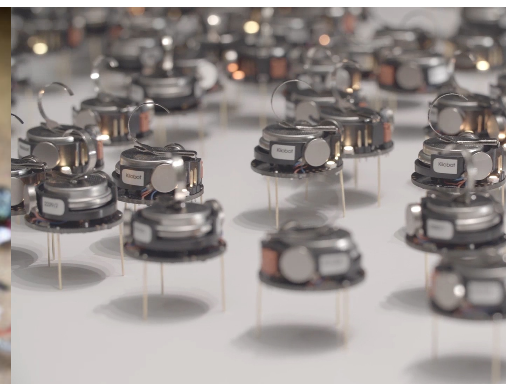
“Autonomous Mobile Robots”

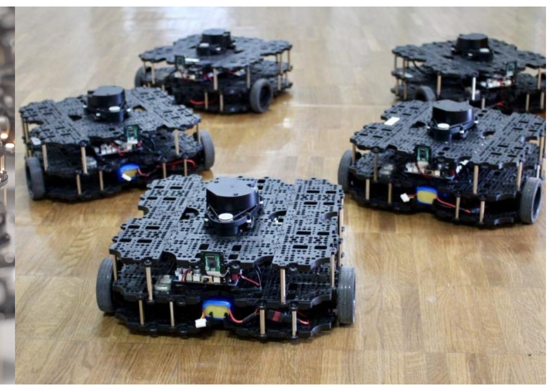




Dr. Daniel Drew

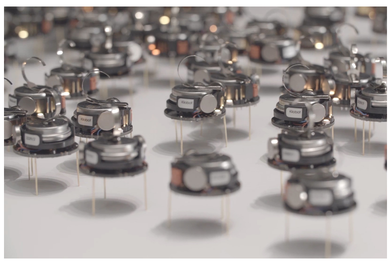
COLLEGE OF ENGINEERING
UNIVERSITY *of* HAWAIʻI *at* MĀNOA

Multi-robot Systems

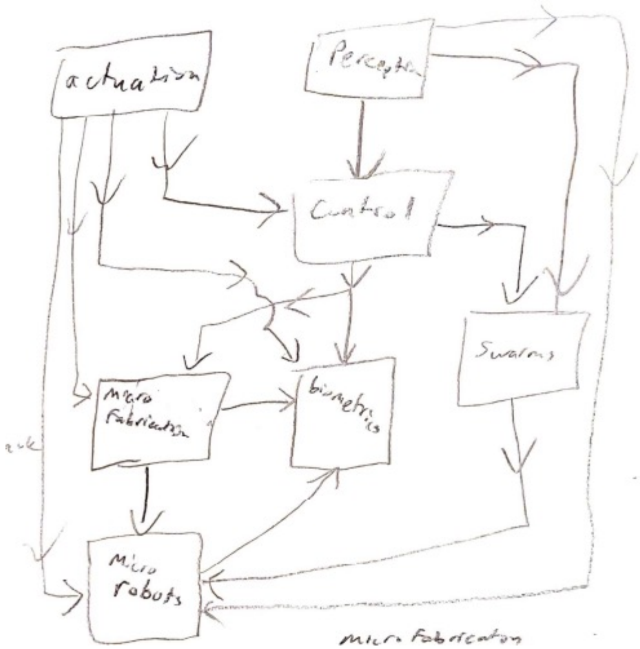


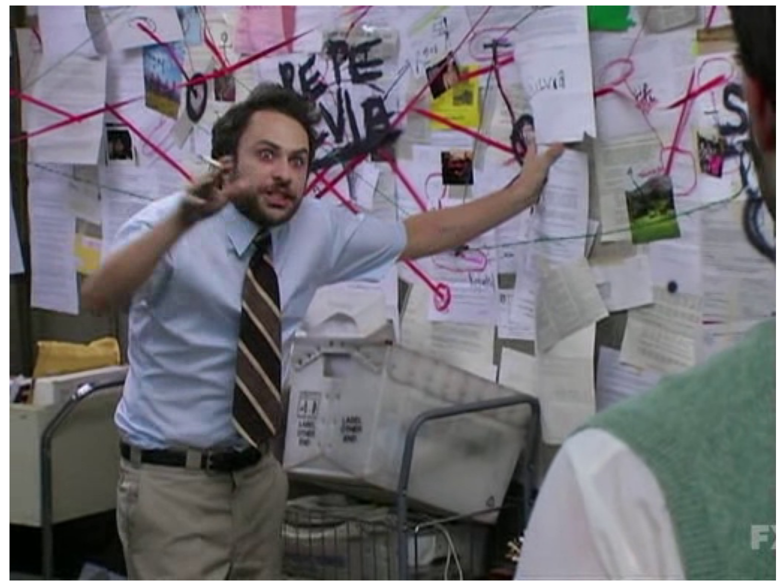


Multi-robot Systems



Course One-Pager: Robot Design Schematic



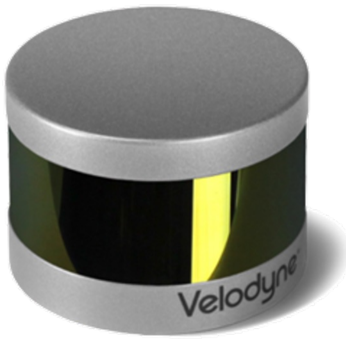


Hierarchy and structure in these conceptual diagrams is hard!

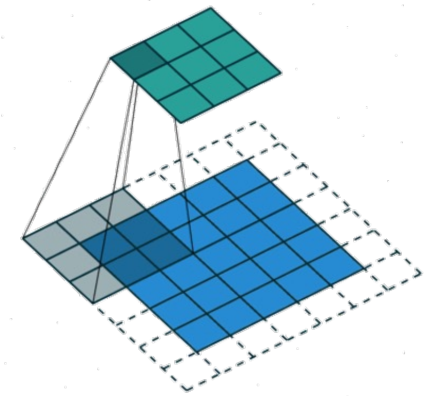
In general, as with most engineering problems, work from constraints

Pillars of Robotics

Action
(not just moving)



Perception
(not just sensing)



Intelligence
(not just classification)

Pillars of Robotics: Action

Action

Modality: Ground? Air? Water?

Base: Wheels? Treads? Legs?

Transmission: Direct drive? Gearbox? Belt?

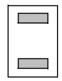
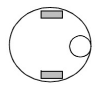
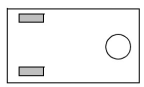
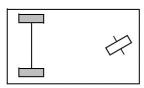
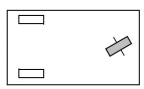
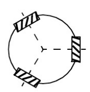
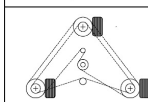
Steering: Differential? Ackermann? Omni?

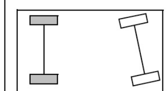
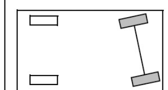
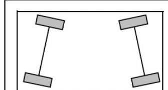
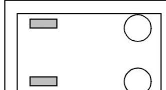
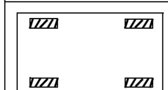
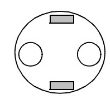
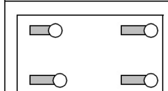
RPM: PID? LQR? MPC?

Motion planning: Cascaded PID? Model-based?

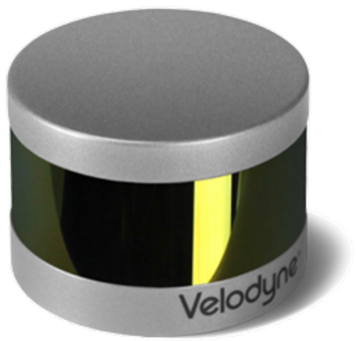
Pillars of Robotics: Action

Action

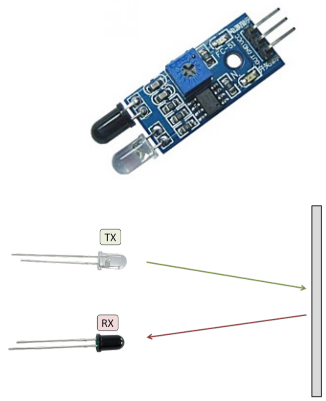
# of wheels	Arrangement	Description	Typical examples
2		One steering wheel in the front, one traction wheel in the rear	Bicycle, motorcycle
		Two-wheel differential drive with the center of mass (COM) below the axle	Cye personal robot
3		Two-wheel centered differential drive with a third point of contact	Nomad Scout, smartRob EPFL
		Two independently driven wheels in the rear/front, 1 unpowered omnidirectional wheel in the front/rear	Many indoor robots, including the EPFL robots Pygmalion and Alice
		Two connected traction wheels (differential) in rear, 1 steered free wheel in front	Piaggio minitrucks
		Two free wheels in rear, 1 steered traction wheel in front	Neptune (Carnegie Mellon University), Hero-1
		Three motorized Swedish or spherical wheels arranged in a triangle; omnidirectional movement is possible	Stanford wheel Tribolo EPFL, Palm Pilot Robot Kit (CMU)
		Three synchronously motorized and steered wheels; the orientation is not controllable	"Synchro drive" Denning MRV-2, Georgia Institute of Technology, I-Robot B24, Nomad 200

# of wheels	Arrangement	Description	Typical examples
4		Two motorized wheels in the rear, 2 steered wheels in the front; steering has to be different for the 2 wheels to avoid slipping/skidding.	Car with rear-wheel drive
		Two motorized and steered wheels in the front, 2 free wheels in the rear; steering has to be different for the 2 wheels to avoid slipping/skidding.	Car with front-wheel drive
		Four steered and motorized wheels	Four-wheel drive, four-wheel steering Hyperion (CMU)
		Two traction wheels (differential) in rear/front, 2 omnidirectional wheels in the front/rear	Charlie (DMT-EPFL)
		Four omnidirectional wheels	Carnegie Mellon Uranus
		Two-wheel differential drive with 2 additional points of contact	EPFL Khepera, Hyperbot Chip
		Four motorized and steered castor wheels	Nomad XR4000

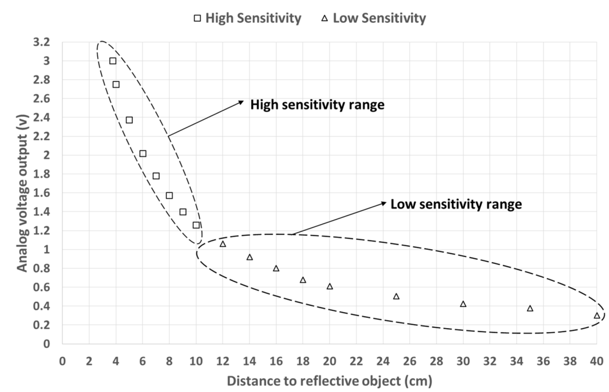
Pillars of Robotics: Perception



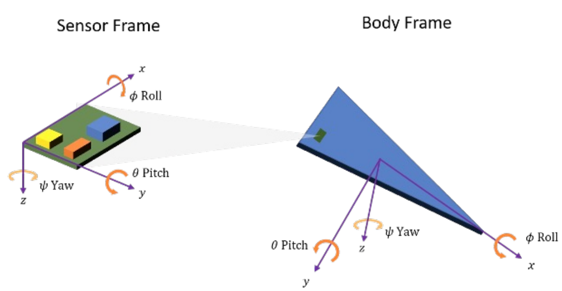
Perception



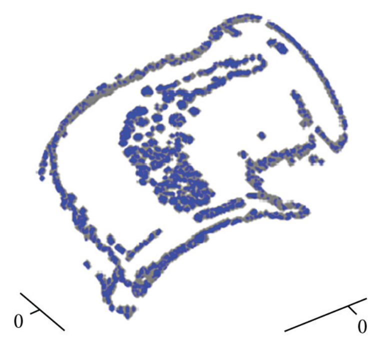
Physics



Calibration

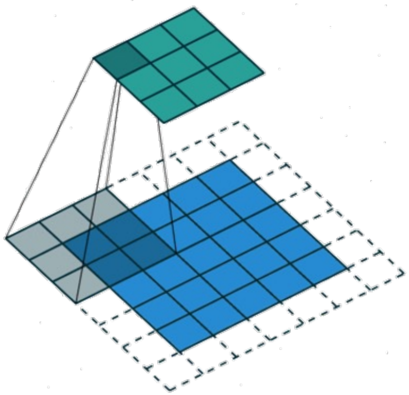


Referencing

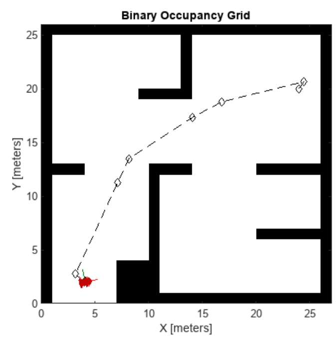


Processing

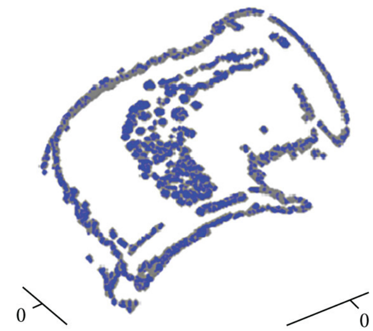
Pillars of Robotics: Intelligence



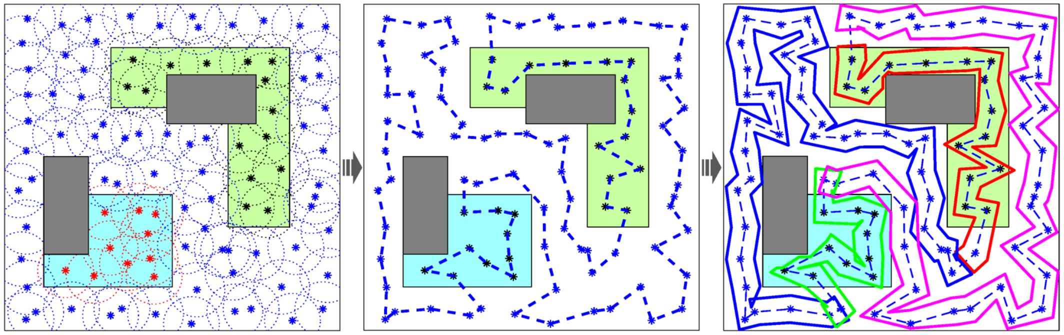
Intelligence



Motion Planning

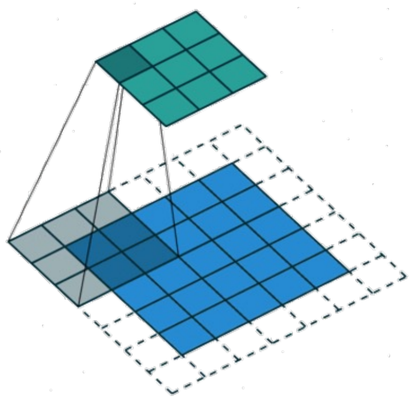


Processing



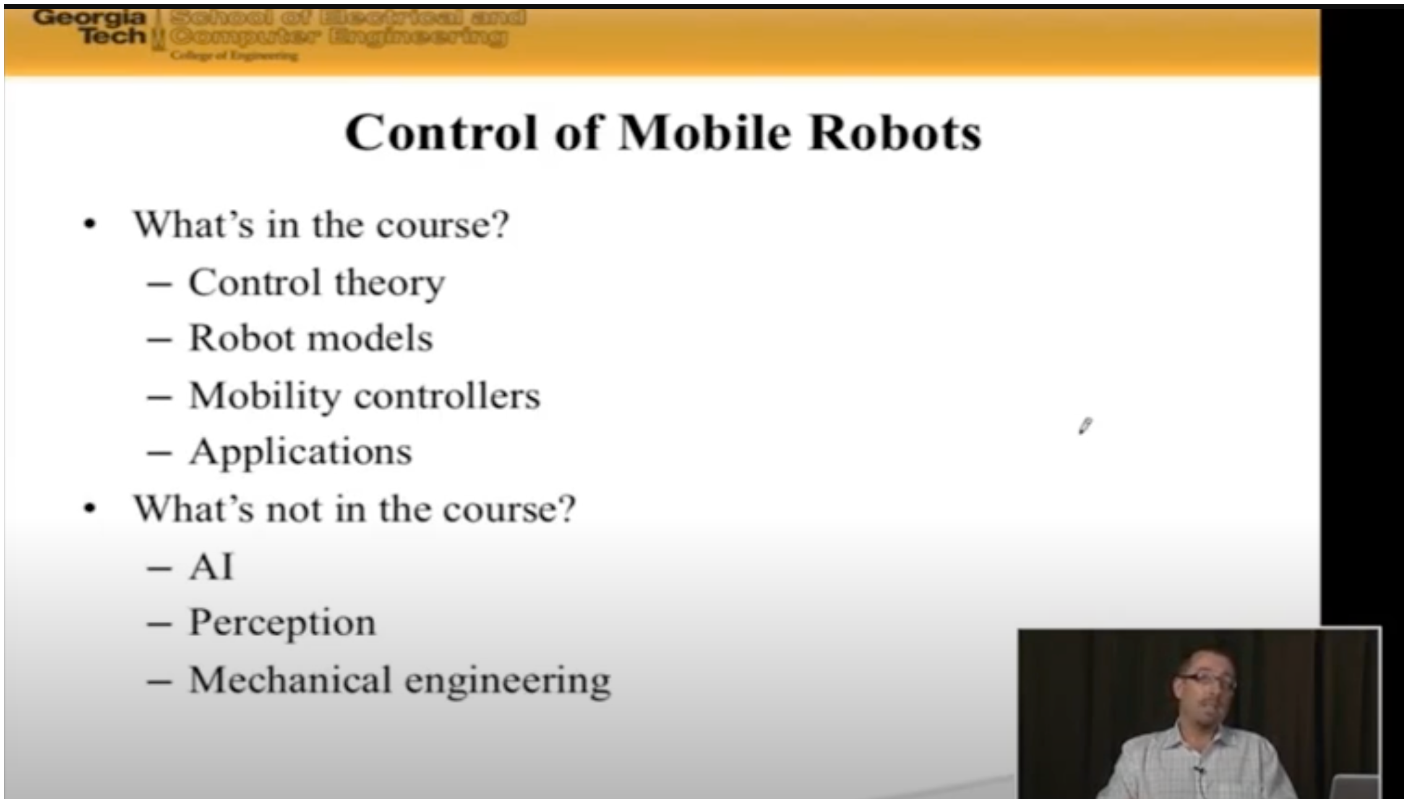
Collision avoidance, coverage targets, …

Pillars of Robotics: Intelligence



Intelligence

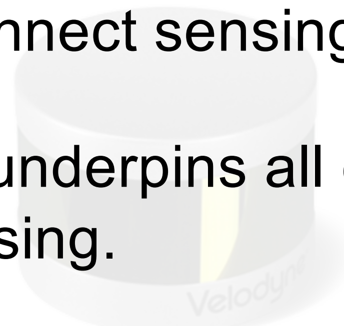
Pillars of Robotics: Not One Course of Material



Magnus Egerstedt, Georgia Tech (now CoE Dean at UC Irvine)

The Robotarium: Swarms in the Cloud

Pillars of Robotics: No “Clean” Boundaries

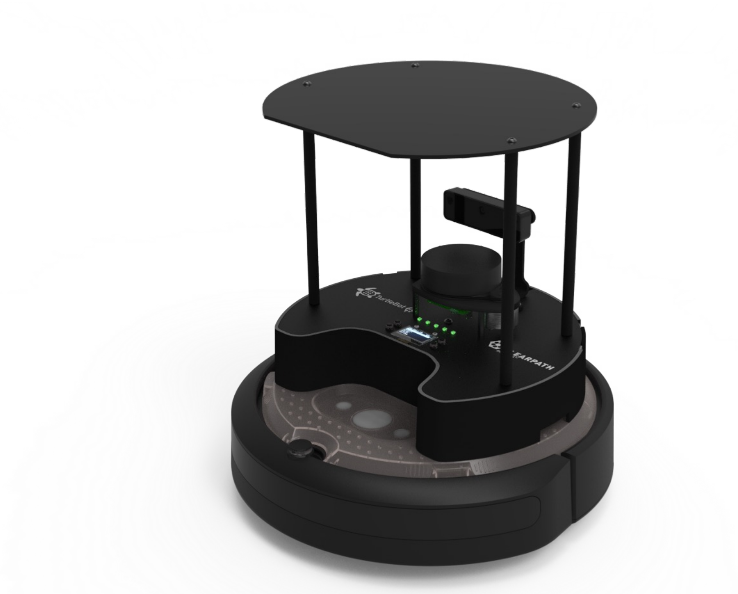


Encoders connect sensing to action.

Intelligence underpins all control and signal processing.

Complex systems have hierarchy of closed system blocks, so robots are really composed of lots of little complex multi-domain building blocks. (see TurtleBot, to follow)

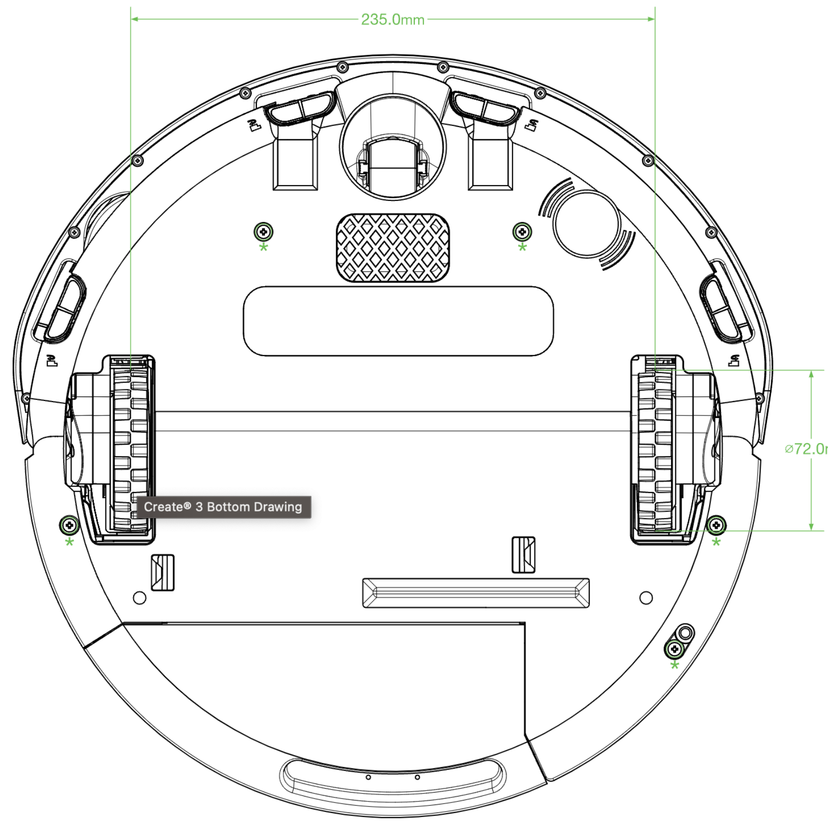
Case Study: The TurtleBot



TurtleBot 4

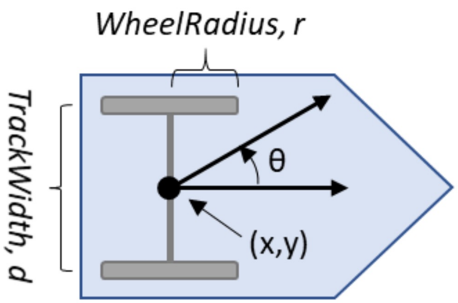
TurtleBot 4	
341 x 339 x 351 mm	**Dimensions**
3.9 kg	**Weight**
15 kg & 0.31 m/s	**Max. Payload & Speed**
2.5 - 4.0 hrs (load dependent)	**Operating Time**
OAK-D-PRO	**Camera**
RPLIDAR-A1	**LiDAR**
Yes	**Accessible Power & USB Ports**
Yes	**OLED Display**
Yes	**Mounting Plate**
ROS 2	**Software**
Raspberry Pi 4B (4 GB)	**Computer**

Case Study: The TurtleBot: Action



iRobot Create 3

Case Study: The TurtleBot: Action

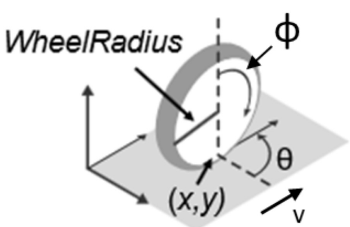


$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \frac{r}{2}\cos(\theta) & \frac{r}{2}\cos(\theta) \\ \frac{r}{2}\sin(\theta) & \frac{r}{2}\sin(\theta) \\ -r/d & r/d \end{bmatrix} \begin{bmatrix} \dot{\phi}_L \\ \dot{\phi}_R \end{bmatrix}$$

Simple Differential Drive Model

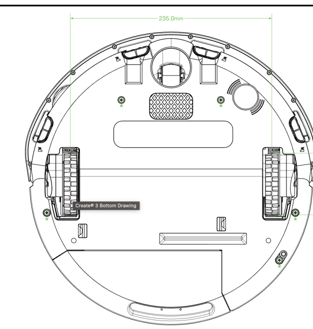
(both described by $[x\ y\ \theta]$)

$[x, y]$ = Vehicle position
$\dot{\phi}$ = Wheel speed
ω = Vehicle heading angular velocity



$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} r\cos(\theta) & 0 \\ r\sin(\theta) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \omega \end{bmatrix}$$

Unicycle Model



Case Study: The TurtleBot: Action

ros2 run teleop_twist_keyboard teleop_twist_keyboard

```
This node takes keypresses from the keyboard and publishes them
as Twist messages. It works best with a US keyboard layout.
---------------------------
Moving around:
   u    i    o
   j    k    l
   m    ,    .

For Holonomic mode (strafing), hold down the shift key:
---------------------------
   U    I    O
   J    K    L
   M    <    >

t : up (+z)
b : down (-z)

anything else : stop

q/z : increase/decrease max speeds by 10%
w/x : increase/decrease only linear speed by 10%
e/c : increase/decrease only angular speed by 10%

CTRL-C to quit

currently:      speed 0.5       turn 1.0
```

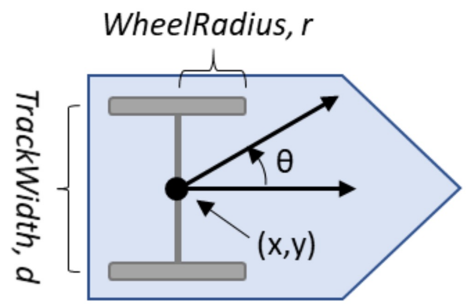
```
ros2 topic pub /cmd_vel
"twist:
  linear:
    x: 0.0
    y: 0.0
    z: 0.0
  angular:
    x: 0.0
    y: 0.0
    z: 0.0"
```

linear.x = drive forward / back
angular.z = rotate left / right

Case Study: The TurtleBot: Action

```
ros2 topic pub /cmd_vel
"twist:
  linear:
    x: 0.0
    y: 0.0
    z: 0.0
  angular:
    x: 0.0
    y: 0.0
    z: 0.0"
```

linear.x = drive forward / back
angular.z = rotate left / right



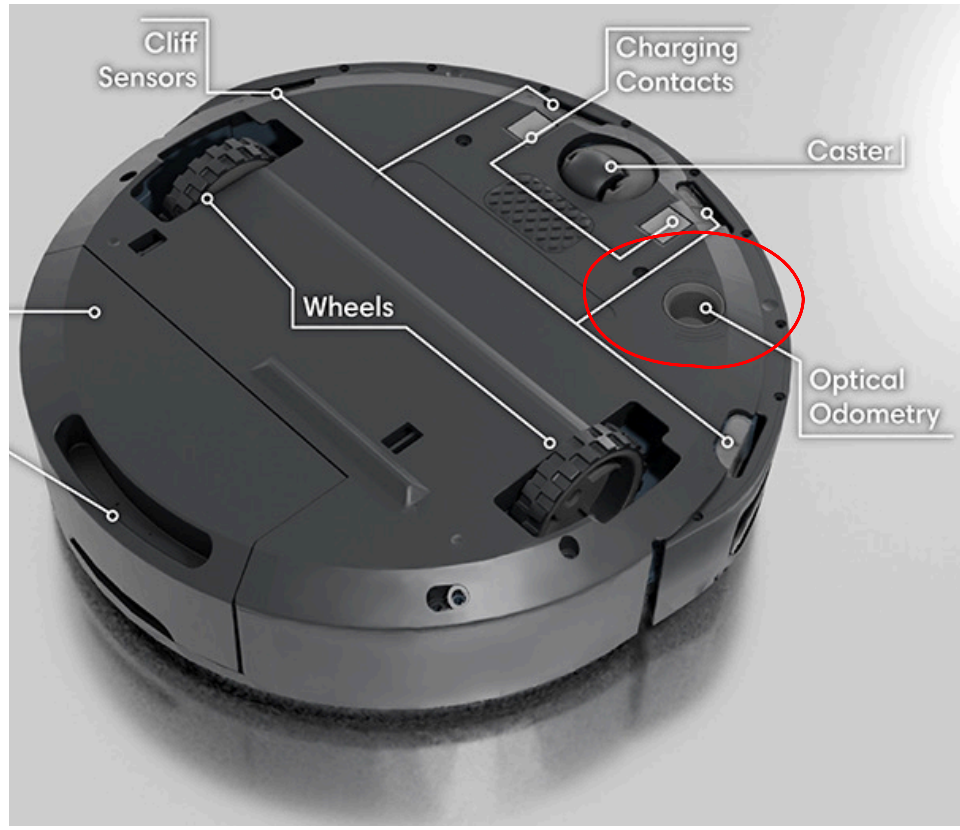
$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \frac{r}{2}\cos(\theta) & \frac{r}{2}\cos(\theta) \\ \frac{r}{2}\sin(\theta) & \frac{r}{2}\sin(\theta) \\ -r/d & r/d \end{bmatrix} \begin{bmatrix} \dot{\phi}_L \\ \dot{\phi}_R \end{bmatrix}$$

Simple Differential Drive Model

$[x, y]$ = Vehicle position
$\dot{\phi}$ = Wheel speed
ω = Vehicle heading angular velocity

/cmd_vel $\xrightarrow{\text{kinematic model}}$ wheel speed setpoints

Case Study: The TurtleBot: Perception

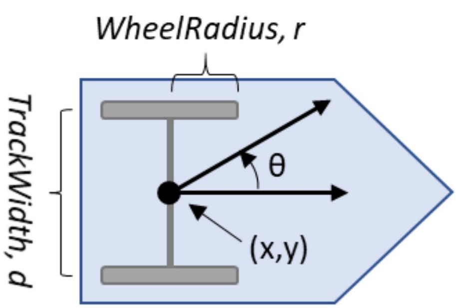


Not shown: 9-axis IMU, wheel encoders

Case Study: The TurtleBot: Perception

$$\text{/cmd\_vel} \xrightarrow{\text{kinematic model}} \text{wheel speed setpoints}$$

What information do we need?



$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \frac{r}{2}\cos(\theta) & \frac{r}{2}\cos(\theta) \\ \frac{r}{2}\sin(\theta) & \frac{r}{2}\sin(\theta) \\ -r/d & r/d \end{bmatrix} \begin{bmatrix} \dot{\phi}_L \\ \dot{\phi}_R \end{bmatrix}$$

$[x, y]$ = Vehicle position
$\dot{\phi}$ = Wheel speed
ω = Vehicle heading angular velocity

Case Study: The TurtleBot: Perception

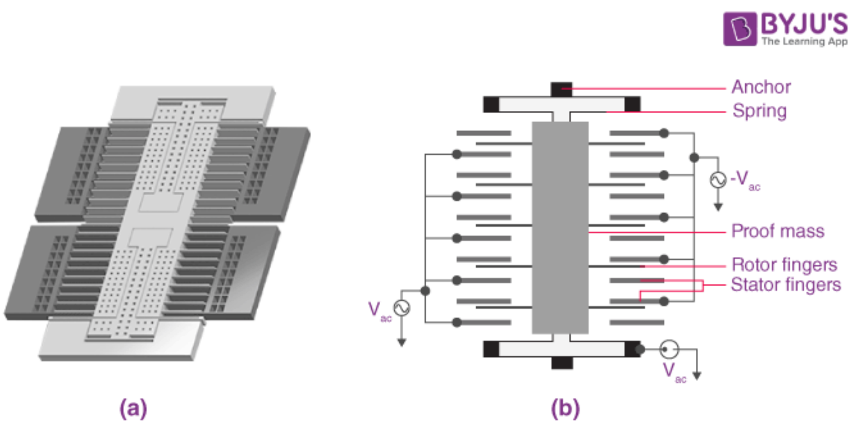
Wheel speed? Wheel encoders.

Case Study: The TurtleBot: Perception

Heading angle? IMU.

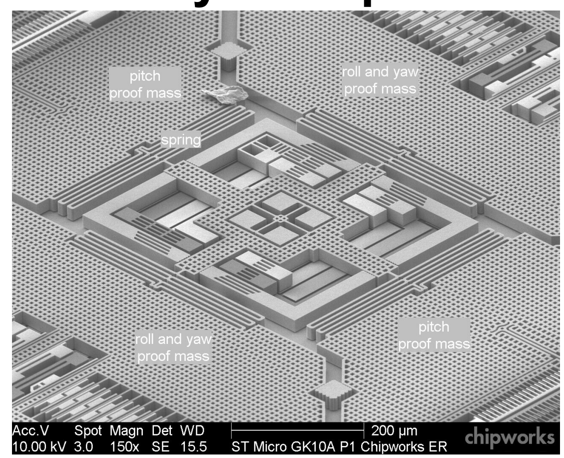
“6-axis IMU” = Accelerometer, gyroscope. “9-axis IMU” = Accel., gyro., magnetometer.

Accelerometer

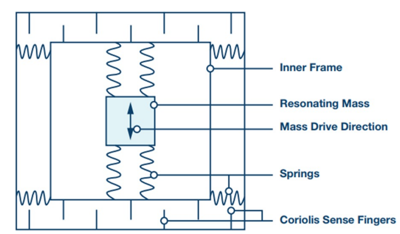


$$F = ma$$

Gyroscope



$$F_C = -2m(\Omega \times \boldsymbol{v})$$

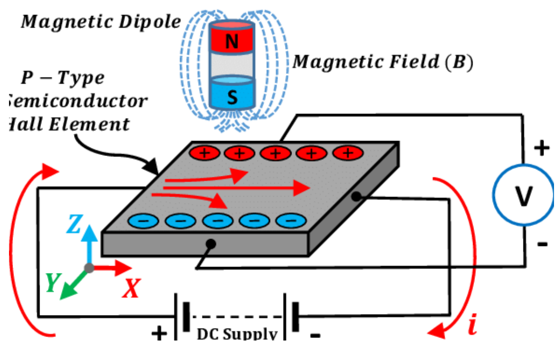


Case Study: The TurtleBot: Perception

Heading angle? IMU.

"6-axis IMU" = Accelerometer, gyroscope. "9-axis IMU" = Accel., gyro., magnetometer.

Magnetometer
aka "compass"



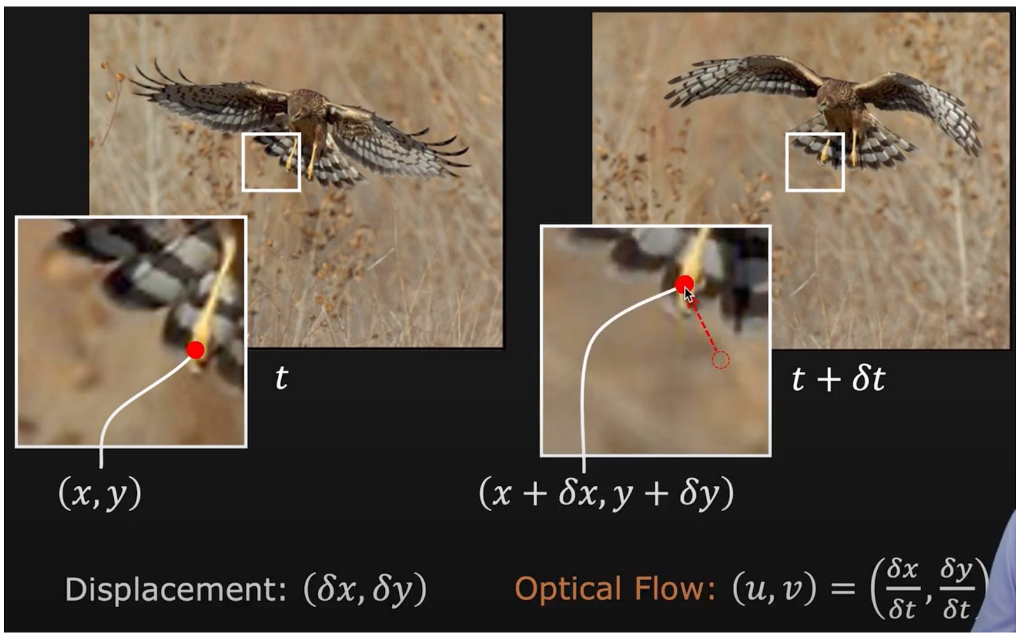
Case Study: The TurtleBot: Perception

Heading angle? IMU.

“6-axis IMU” = Accelerometer, gyroscope. “9-axis IMU” = Accel., gyro., magnetometer.

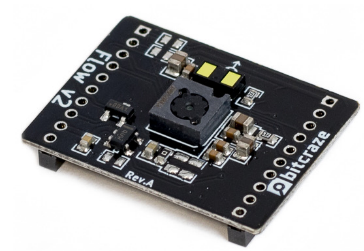
Case Study: The TurtleBot: Perception

Change in position? Optical flow.

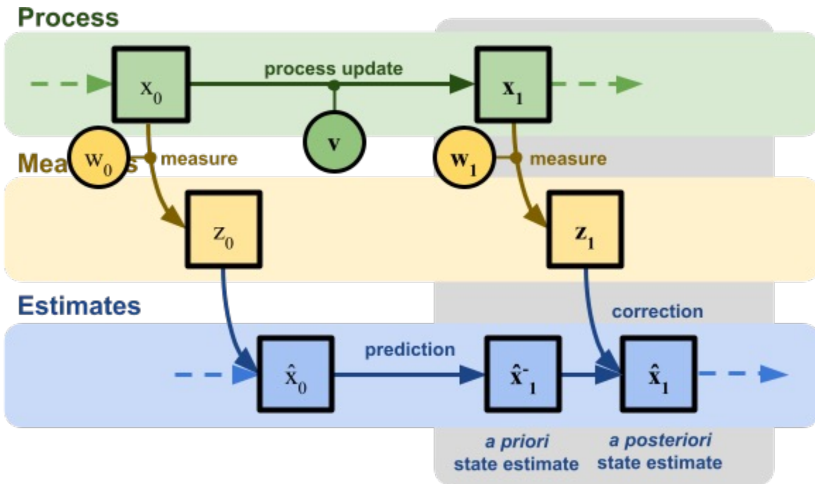


Major assumptions:
constant pixel intensity,
displacement and time step are small

Requires:
camera calibration,
feature recognition



Case Study: The TurtleBot: Perception

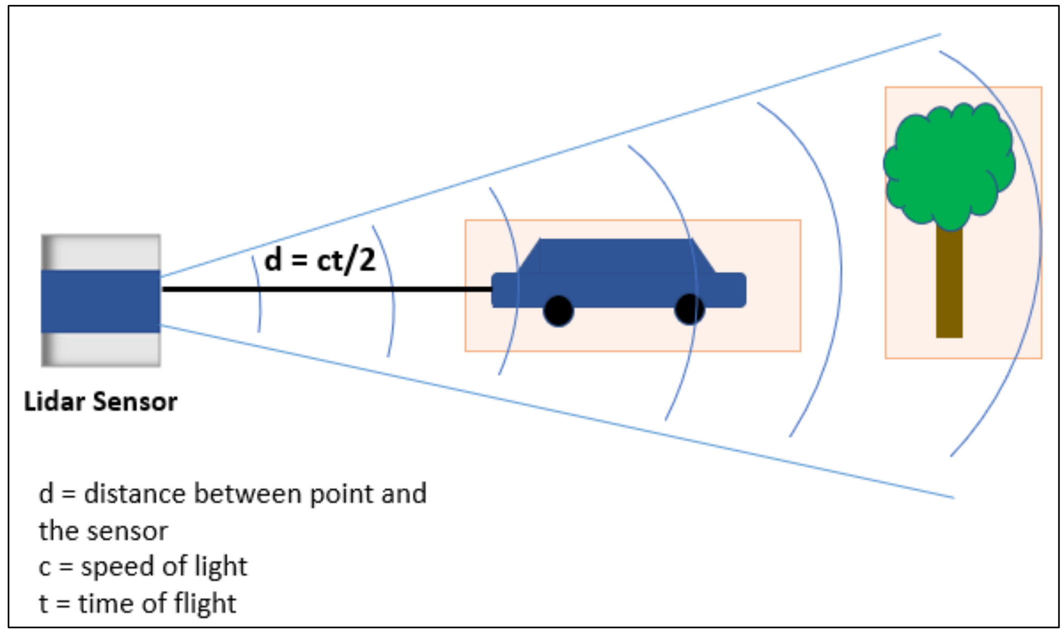


Extended Kalman Filter (EKF)
Alan Zucconi

Note: in reality, we rely on sensor fusion between all three modules!

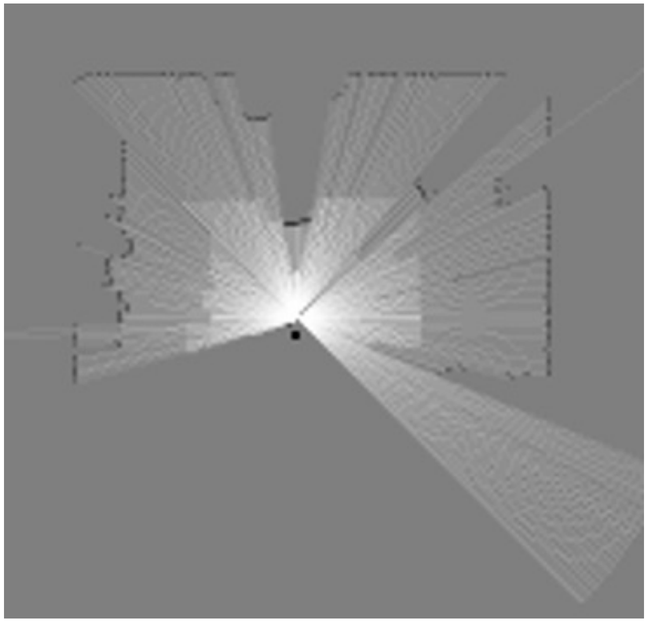
e.g., optical flow can also give rotational velocity to fuse with IMU;
integral of accelerometer data can give linear velocity to fuse with flow

Case Study: The TurtleBot: Perception

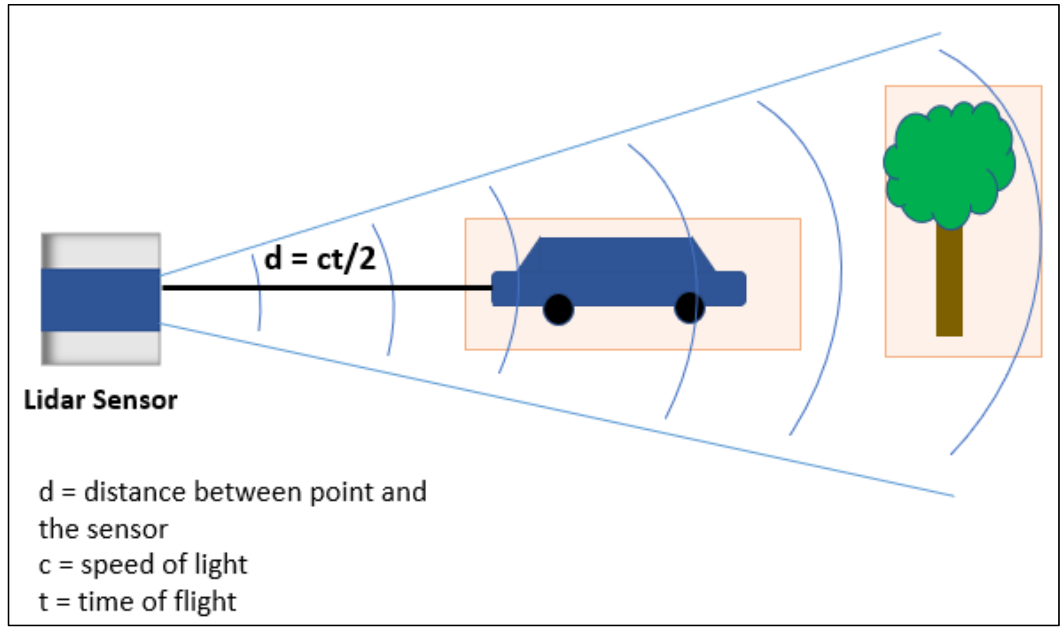


Light Detection and Ranging
(ToF range-finding)

Case Study: The TurtleBot: Perception

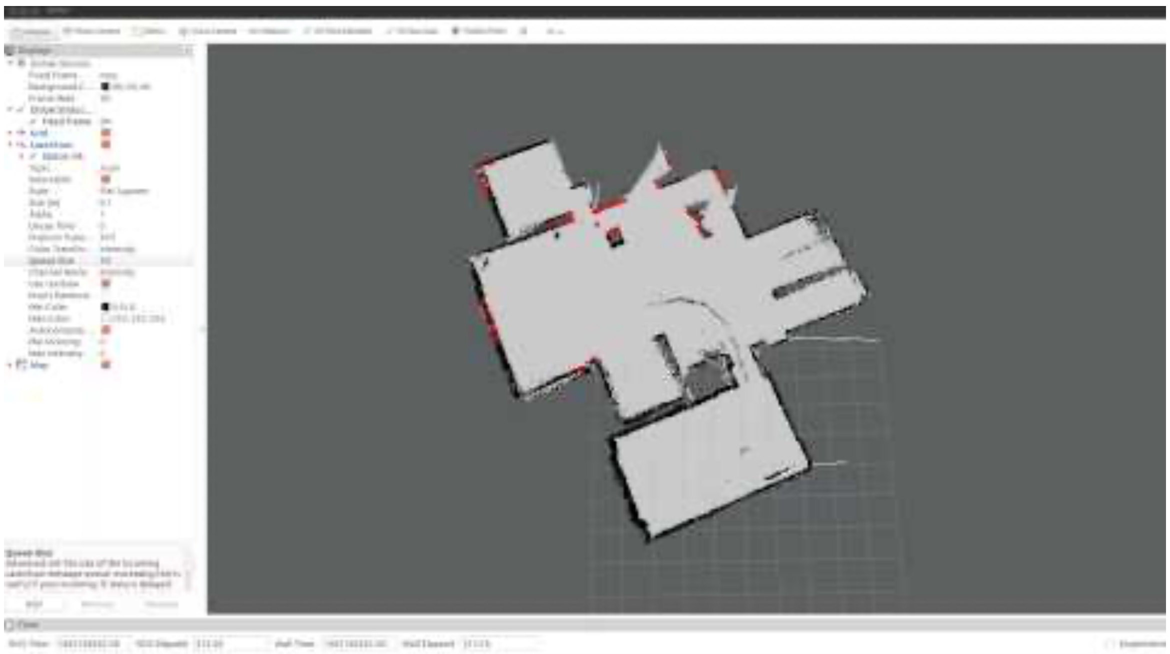


Pointcloud



Light Detection and Ranging
(ToF range-finding)

Case Study: The TurtleBot: Perception

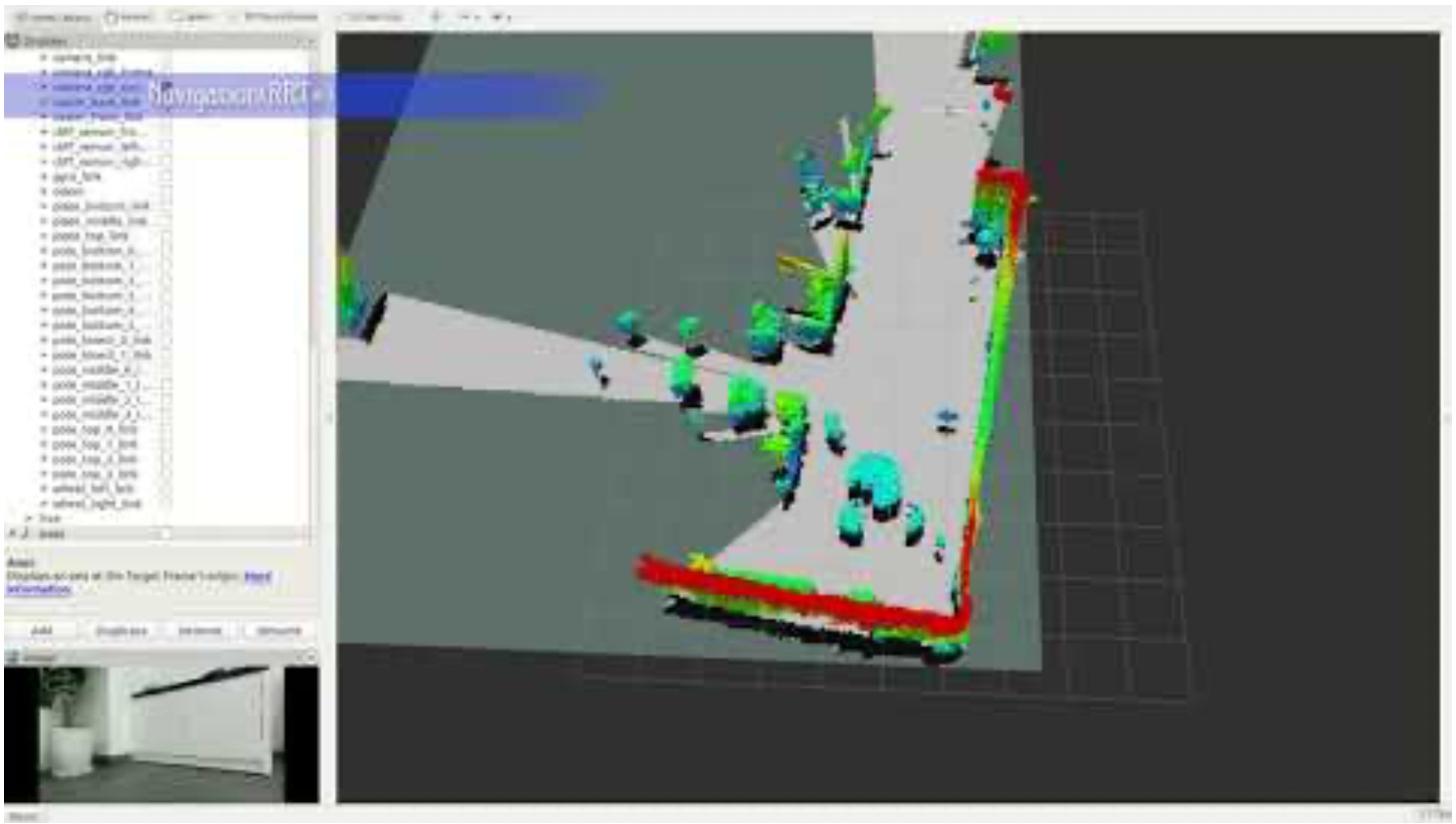


LiDAR Slam

SLAM
(connects all three pillars)

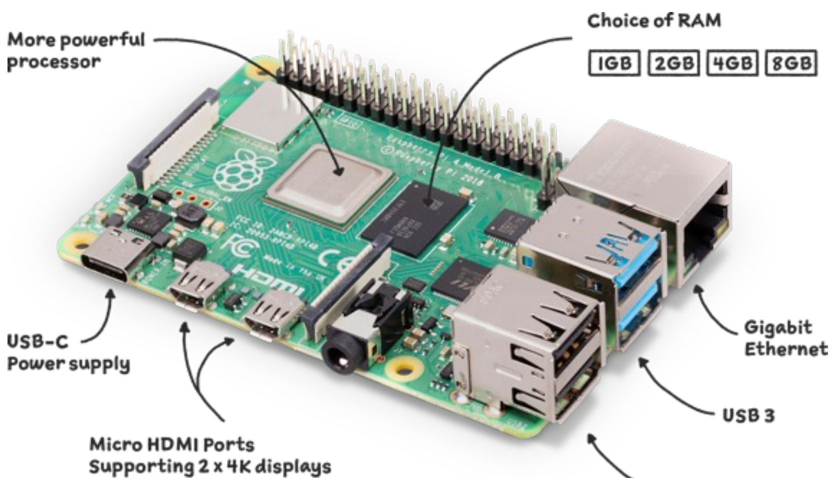
Case Study: The TurtleBot: Perception

Case Study: The TurtleBot: Perception



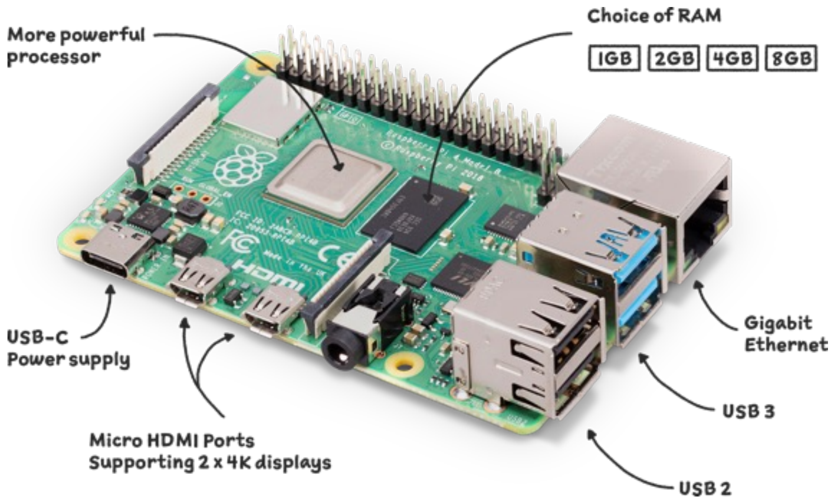
Visual Slam

Case Study: The TurtleBot: Intelligence

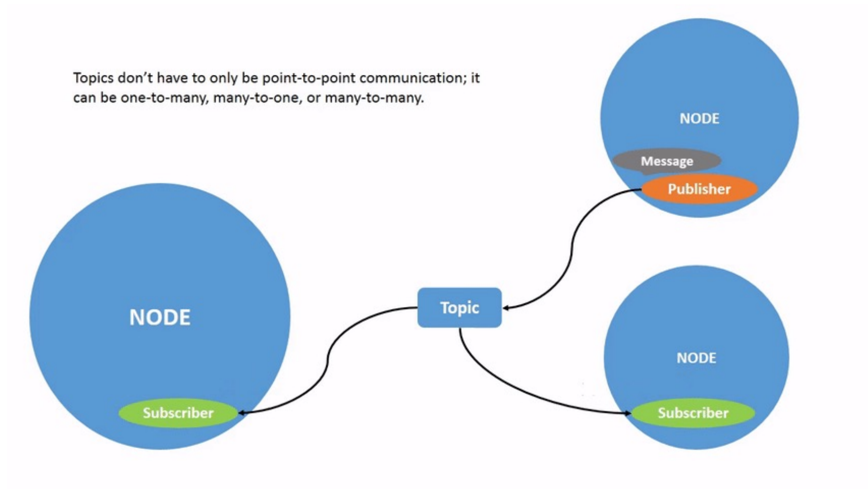


Onboard: Raspberry Pi 4 (4GB)

Case Study: The TurtleBot: Intelligence



Onboard: Raspberry Pi 4 (4GB)



ROS2: pub/sub model, modular
e.g., the LiDAR "node" publishes distance information
Note: "services" are response driven, not pub/sub like topics

Case Study: The TurtleBot: Intelligence

Swarm-on-a-Stick

We will create an open-source swarm robotics kit for education, from scratch.

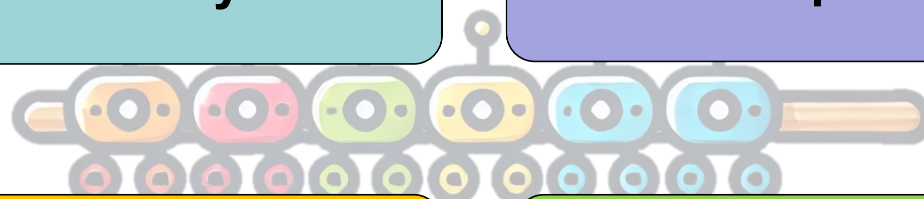
693H Robotic Subsystems

Motors, drive electronics, encoders, PID controllers, top-down tracking

Mobility

Sensor suite (IMU, prox.), calibration code, general IO (buttons and LEDs)

Perception



Compute and Power

Battery and charging, wireless comms, GPIO and power distribution routing

Control and Planning

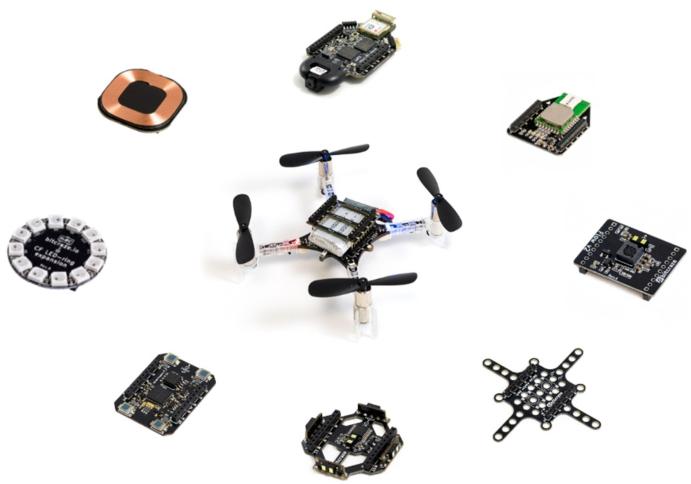
microROS, ROS2, Gazebo simulation, enclosure design

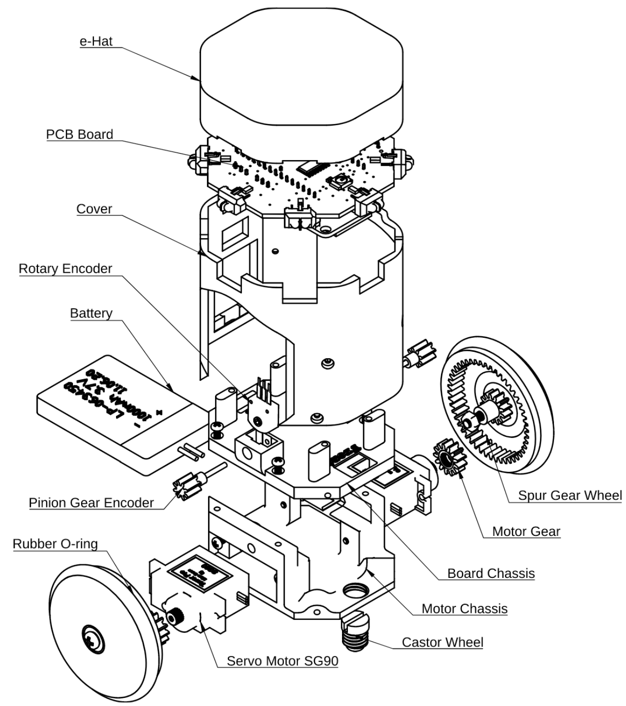
Mechanics / Prototyping

Coding

PCB Design

693H Robotic Subsystems





<u>Each team (besides Control & Planning) designs their own PCB + chassis section</u>

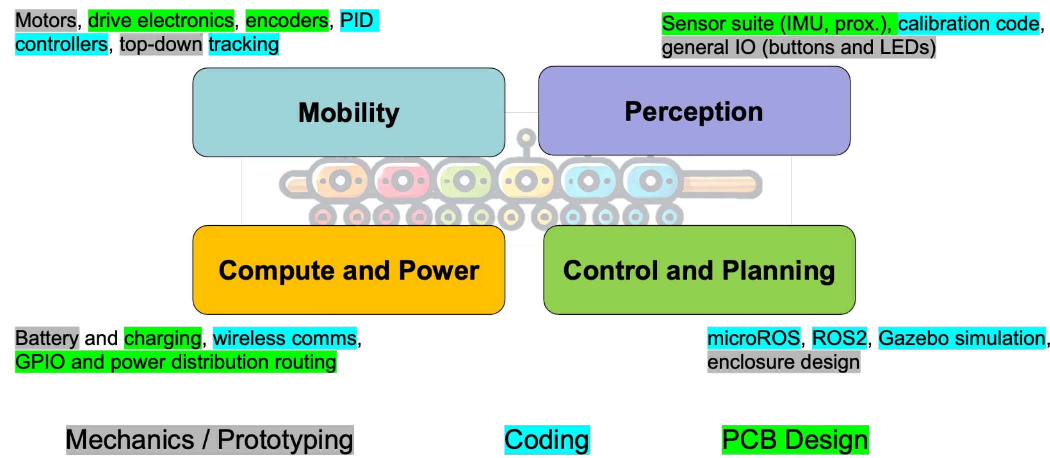
Integration challenges: which signal lines route through? how to connect chassis?

Subsystem Group Formation

Form groups of ~4 for each of the four subsystems:
Mobility, Perception, Compute and Power, Control and Planning

Suggestions:

Interested in more coding / ROS?
    Control and Planning
Interested in more mechanical design?
    Mobility
Interested in more PCB layout?
    Perception, Compute and Power



To do today:
Send me an email (<u>one</u> per group, everyone cced): subject: [693H] Mobility Group, body: your names, at least one time you have <u>each</u> blocked off to work per week, and how your group will primarily communicate with each other (e.g., discord, email).

Course Schedule Reminder

Week	Module	Date	Lecture Topic	In-Class Activity
Week 1	Intro	1/13	Course Intro	Icebreakers
		1/15	Autonomous Mobile Robots	Subsystem Group Formation
Week 2		1/20	MLK Day	
	Design Phase 1	1/22	Action 1	
Week 3		1/27	Action 2	Astound the Class! 1
		1/29	Perception 1	
Week 4		2/3	Perception 2	Subsystem Check-ins 1
		2/5	Intelligence 1	
Week 5		2/10	Design Review Presentations	
		2/12		

- **Astound the Class 1 --- signups and slides posted on website**
- **I will respond to subsystem group emails with additional info for Design Review 1**
- **We don't meet for a week: do the Readings!**
- **Some course slots opened up: feel free to recruit a friend**