International Aerial Robotics Competition Mission 7

2017-2018 Intro Meeting

Meeting Structure

- Intro to IARC
- Subteam Presentations
- Demo
- Short intro meetings with subteam leads

Important Info

- Leads: Aaron, Levi, Quentin, Caroline, Andrew, Liam
- Slack: #iarc7
- Large commitment, **minimum of 8 hours per week**
- Subject areas: Signals, image processing, computer vision, ROS, mechanical simulation, UAV controls, motion planning, power electronics, machine learning, AI
- Competing late July 2018

What is IARC?

- Teams must solve “challenges that are currently impossible for any flying robots owned by government or industry”
- Began with Mission 1 in 1991
- 7th mission began in 2014

What is Mission 7?

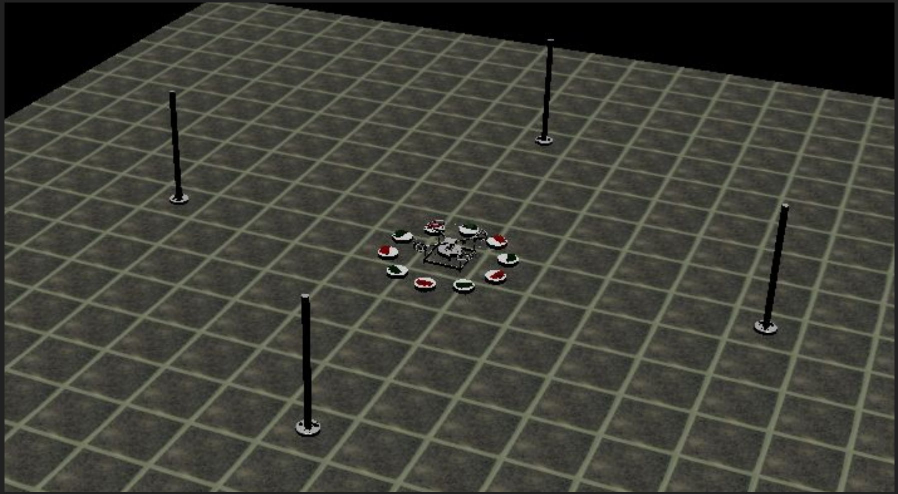
- Affectionately named “herding roombas,” the goal of this competition is to design a drone that:
  - Is fully autonomous
  - Can interact with robots on the ground to direct them toward a destination
  - Can navigate without reference points like GPS or nearby walls

How’d we do?

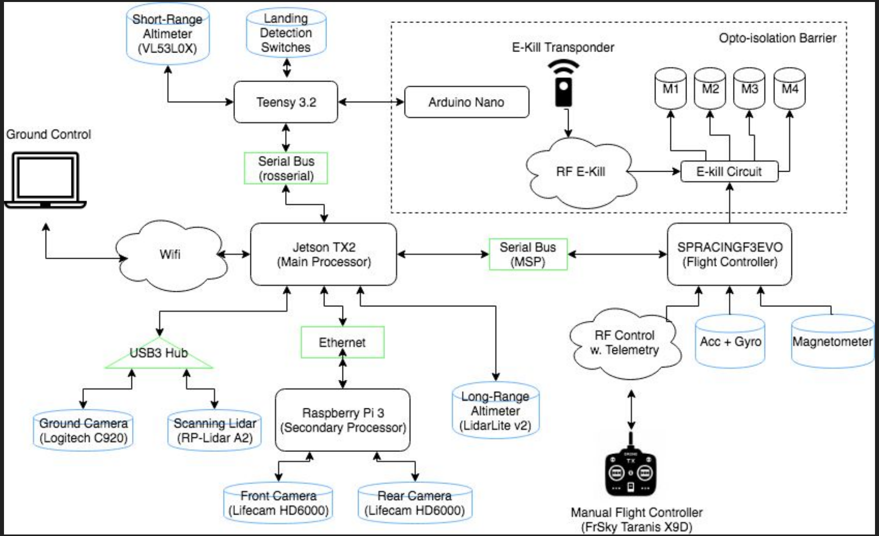
- Best System Design
- Most points overall
- Achieved autonomous flight

What’s the plan?

- V1.0 - Current drone from IARC 2017
- Simulator - Built on MORSE, allows us to test higher-level algorithms
- V2.0 - Improved design to carry more sensors and cameras



Overview of where we are now



Subteams

4 Subteams, looking to expand to 6

• AI / Perception	Aaron Miller, Liam Berti
• Electrical / Controls	Levi Burner
• Mechanical	Caroline Collopy, Quentin Torgerson, Jackie Sharpe
• Planning	Andrew Saba

Future Meeting Structure

- Will have once a week status meetings for everybody
  - Will be sending out a when2meet to schedule it
- Will hold three times a week shop hours
  - Come when it makes sense for you
  - Message in #iarc7 to get sub-team leads you need
- Shop hours:
  - Sunday 4pm-9pm
    Monday 5:30pm-9pm
    Thursday 5:30pm-9pm
- This info will be in the getting started guide

Mechanical

Goals:

- Increased sensor capacity
- Increase strength/reduce weight
- Easier to service
- Design for wires
- Maintain fail-safe behaviors

Prop Guards

Pros:

- Printed for strength
- Designed to fail without breaking carbon fiber
- Drone bounces off walls

Cons:

- Ten different parts
- Stress concentration area
- SUPER HEAVY
- Thermal deformation

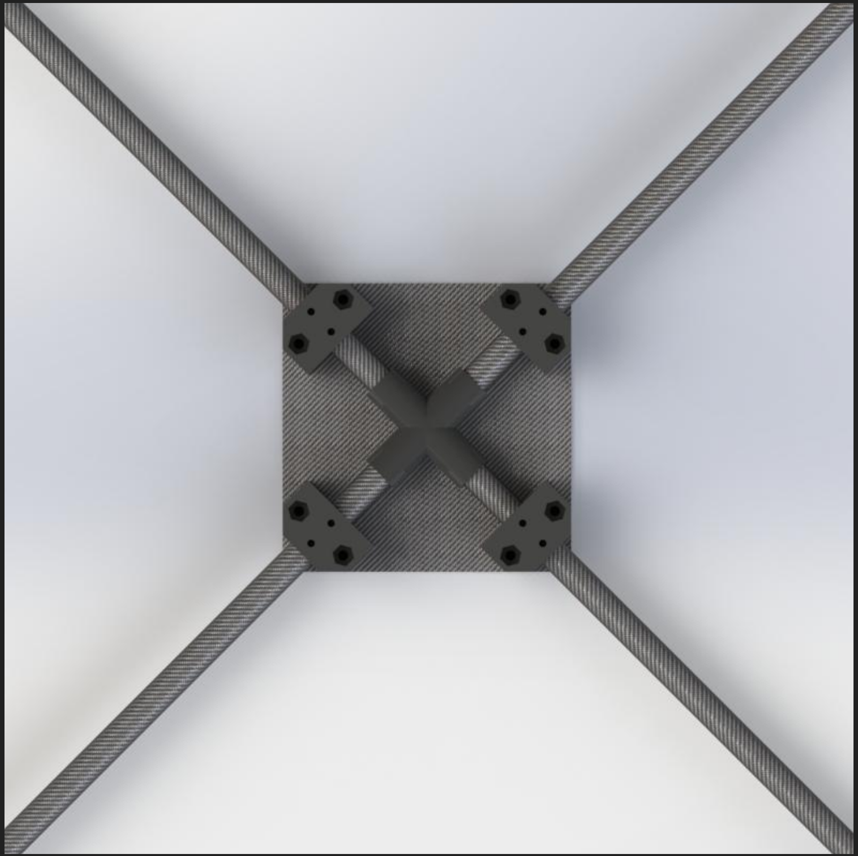
Center Frame

Pros:

- A normal load to tube's axis breaks plastic parts first
- Is firm during normal usage
- All Carbon Fiber parts are relatively inexpensive

Cons:

- Round tubes = difficult mounting
- Hard to assemble
- Uses gigantic heavy bolts
- Replacing 3D printed parts requires a lot of print time



Shown without top plate

Landing Gear

Pros:

- Springs for shock absorption
- Low friction sliding pads to lessen stress on frame when landing
- Plungers to detect ground contact

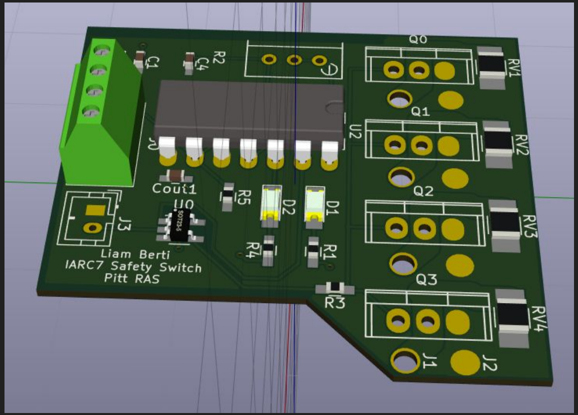
Cons:

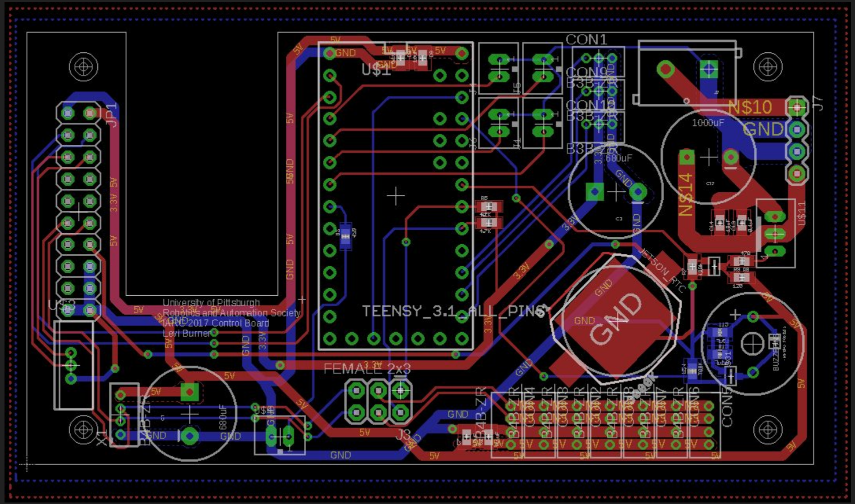
- Plungers break
- Slider pads/mounting pieces are heavy
- Shock absorption not optimized

Electrical

Batteries, Computers, Motors, Wiring Harness, Sensors…

- Support new computational, propulsion requirements
  - Integrate wiring harness, boards into mechanical design
  - Optimize batteries for payload
- Improve existing electrical elements
  - Improve E-Kill
    - New/Smaller MOSFETs, GaN FETS
  - Miniaturize boards
  - Interfaces for more sensors
  - Un-jank motor voltage monitoring
- Improve and plan for wiring harness

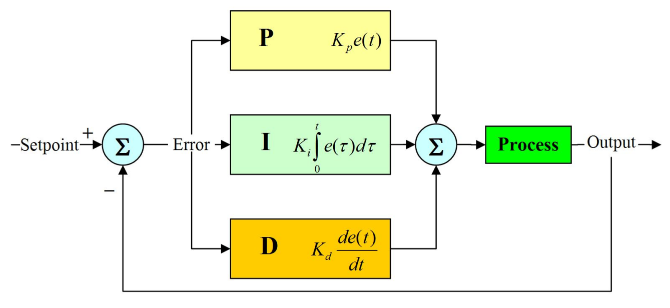


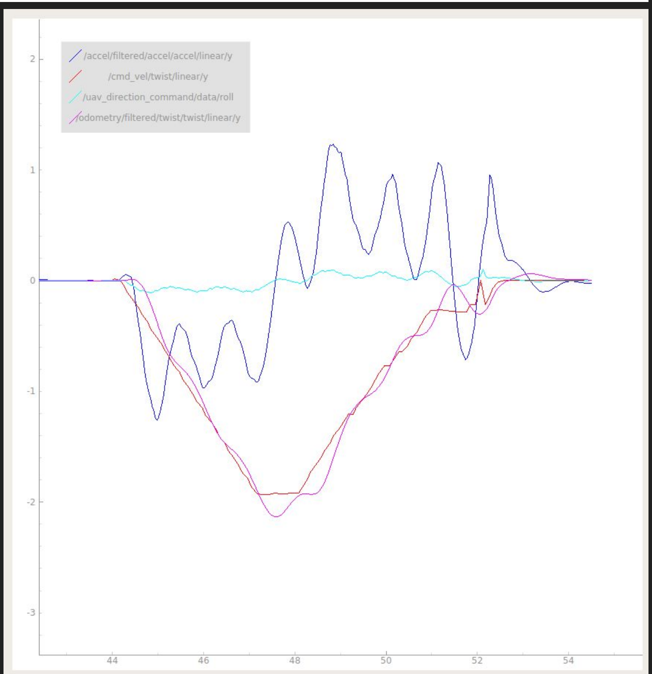


Controls

What is a controller?

- Translates a primitive motion request to reality
  - Receives target speed in m/s translates
  - Pitches drone forward
  - Levels drone out when speed is reached.
- There are several layers of controllers
  - Primitives run in the flight controller
    - Control the pitch, roll, yaw
  - Translation controllers run in Low Level Motion
    - Control velocity
  - Most are PID controllers
    - Use feedforward
    - Thrust modeling to linearize controller output

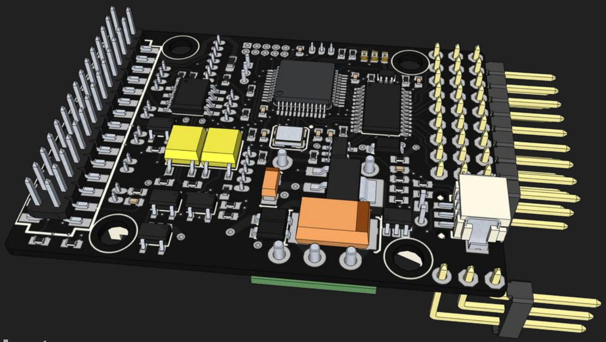


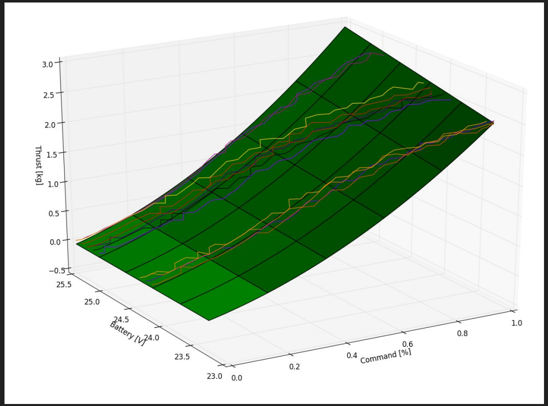


Controls

This year's goals:

- Replace racing drone FC with PixHawk and ArduPilot
  - Good for larger drones
  - Vastly improved pitch, roll, yaw controllers
  - Needs to not break software stack for V1.0
- Improve translation controllers
  - Currently the controllers are intolerant of sensor latency
  - Need multi-layered acceleration controllers to achieve reliability
  - Support smoother transitions between controller types
- Controls is a fantastic subject area:
  - Combine firmware, signal processing, modeling, and controls
  - Make the real world useable





Planning

What is “planning”

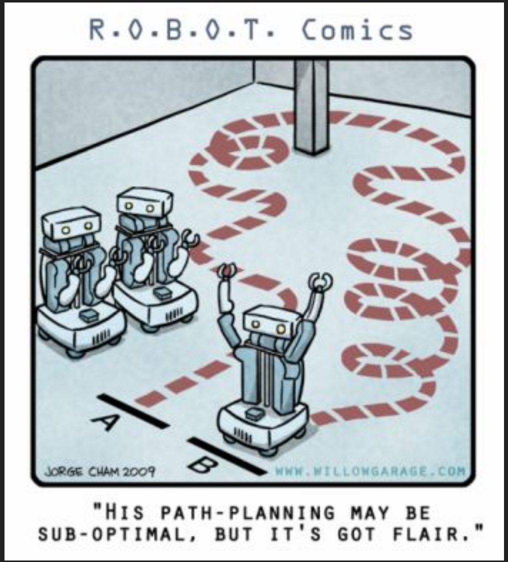
- Using desired targets to determine desired next position, velocities, etc
- Correcting for obstacles/determining “best path”

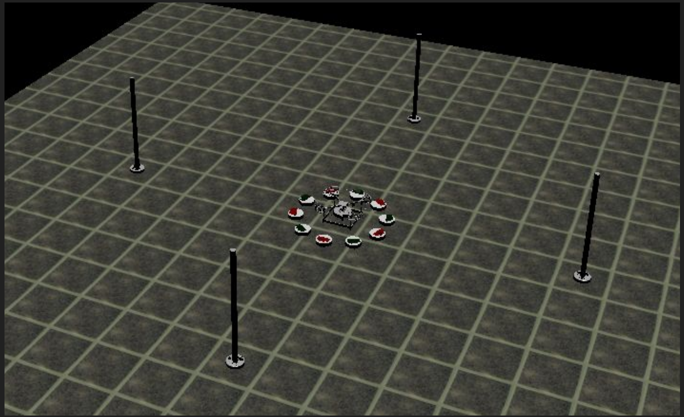
Above LLM sits High Level Motion (HLM)

- Separate tasks for each high level motion plan
  - Track roomba, hit roomba, height hold...
  - Provides velocity targets for LLM
- motionplanner.py handles task targets
- iarc_task_action_server.py handles task life cycle

Moving Forward:

- Need obstacle avoidance
- Improved path planning
- Better task/state transitions

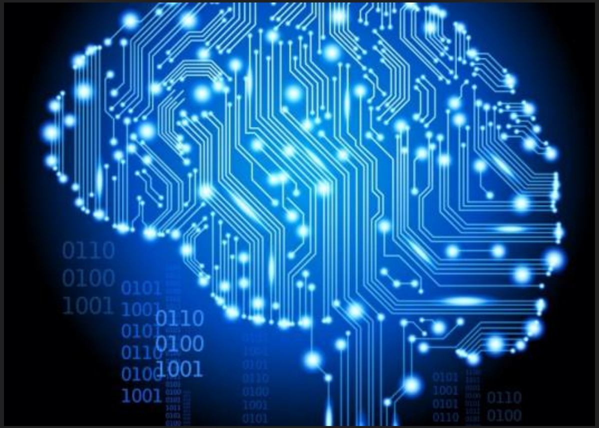




Abstracts

Abstracts:

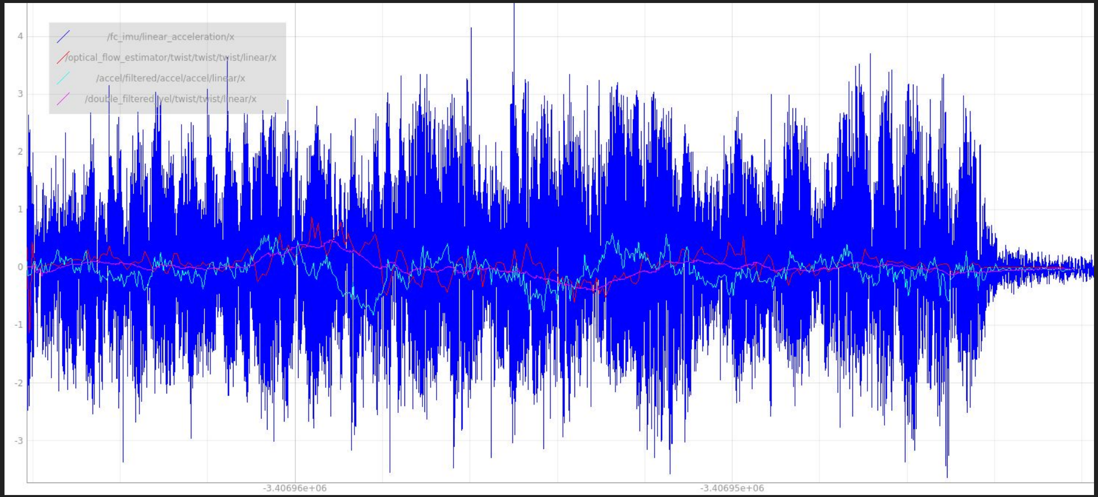
- “Way you and I think”
- Combine multiple tasks to execute even higher level behaviors
  - Ex. takeoff, then track a roomba, then land
- Used for testing of tasks
- Will be utilized/taken over by AI at highest level

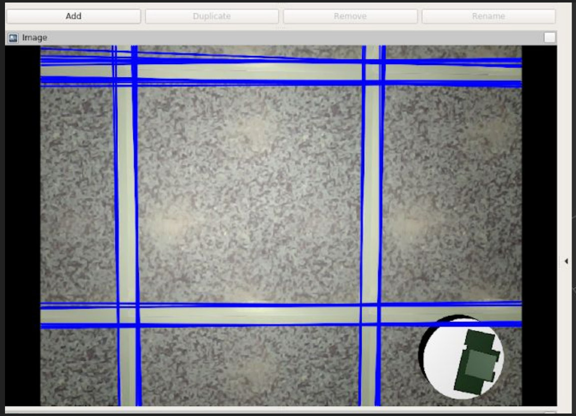


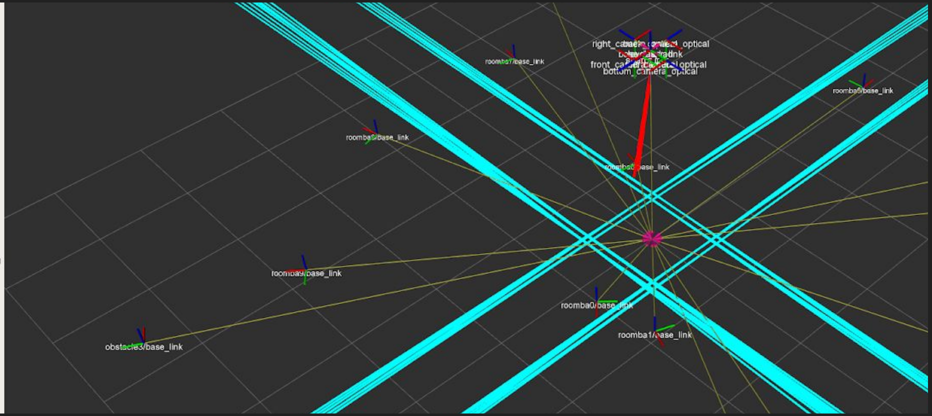
Perception

State Estimation

- Fusion of multiple sensors
    - Accelerometer, Optical Flow, Altimeters, Grid Finder/Counter
- Extended Kalman Filter



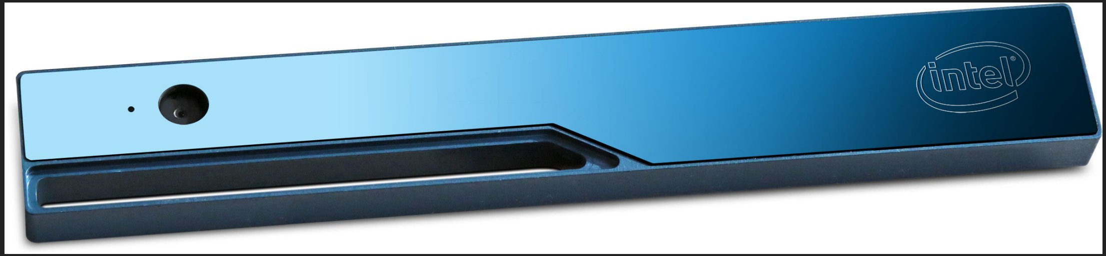




Perception

Obstacle Detection

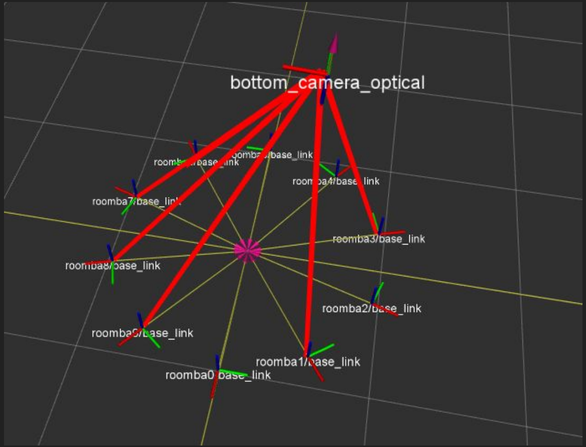
- v1.0: Scanning LIDAR
- v2.0: Depth Cameras



Perception

Roomba Localization

- Downward- and side-facing cameras
- Currently using GHT, upgrading to CNN

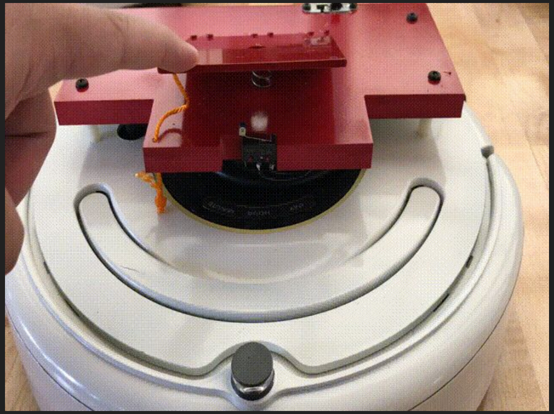


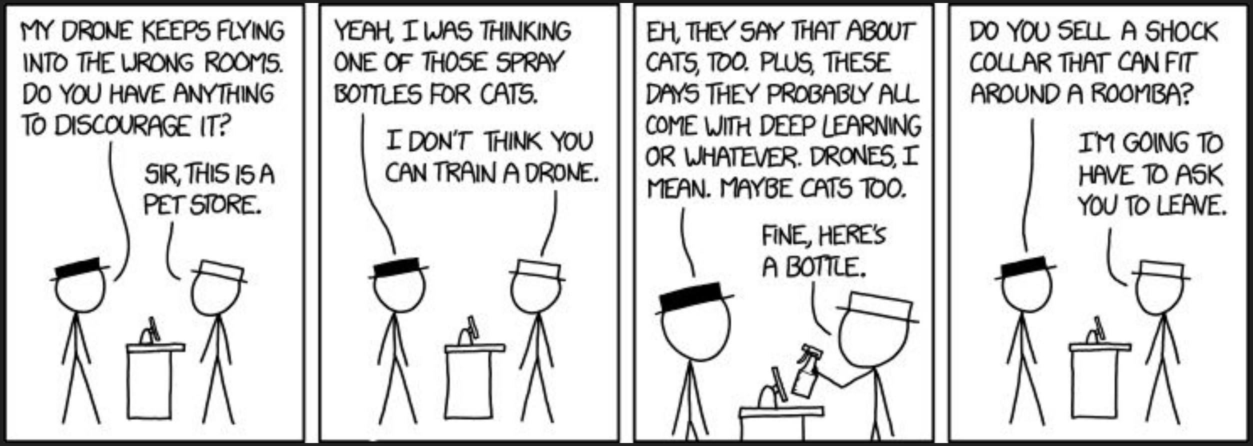
Artificial Intelligence

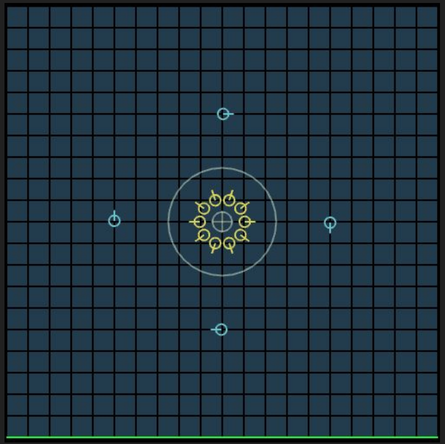
Drone must decide which actions to take

- Roomba interaction
- Exploration

Approaches: state machine, calculate approximate reward function, reinforcement learning (e.g. Q-learning)



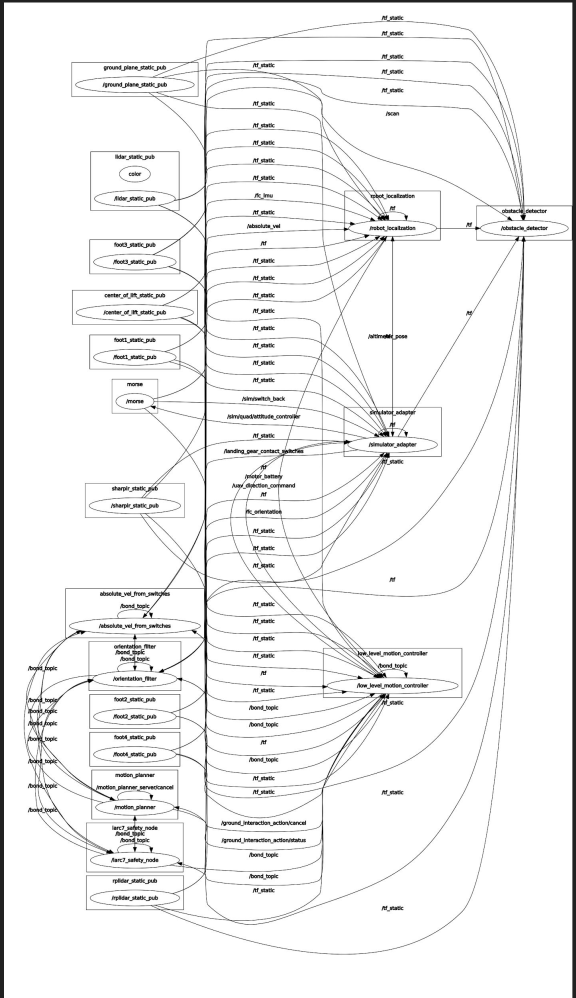




Robot Operating System (ROS)

It's like a social network, for sensors and transforms…

- Introduces the "Publisher/Subscriber" model to robots
- Makes systems more flexible
  - ROS communicates over networks
    - Android phone is subscribed to a publisher on AWS, etc
  - Nodes are independent of each other
  - Many subscribers can listen to a publisher
- Officially supports Ubuntu GNU + Linux
  - Experimentally supports Android, Debian, Arch Linux, OS X



Going Forward!

- Use the rest of the meeting to meet sub-team leads and people
- Join our slack!!!! pittras.slack.com
  - Join the #iarc7 channel
- If you want to do something:
  - Talk to your subteam lead
  - Follow the getting started guide: goo.gl/qiU6FM