ECE 2031 Final Design Project

Introduction

ECE 2031 Fall 2019

Lab 8 and Project

- We really cannot cover all the details until you've done Lab 8
- The Fall Semester calendar leaves us with a partial week, when we cannot do Lab 8
- We'll cover part of the project today, and also cover part of the long Lab 8 lecture

Partial Lab 8 lecture

- Subroutines…

Design Project Motivation

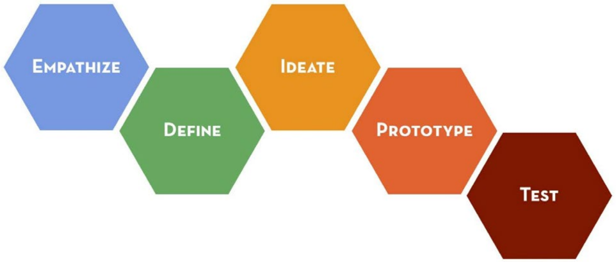
- ECE 2031 includes the sophomore-level team design experience
- You are developing a useful set of tools
  - eventually including an entire computer within the DE2 board
- Using tools creatively to solve problems is what engineers and computer scientists do

ECE 2031 Project Components

- Propose a solution to a problem:
  - What's the best approach for completing a given task within the constraints of the project?
  - More details next week on proposal
- Implement the proposed design on the "DE2Bot"
- Demonstrate, present, and document your solution

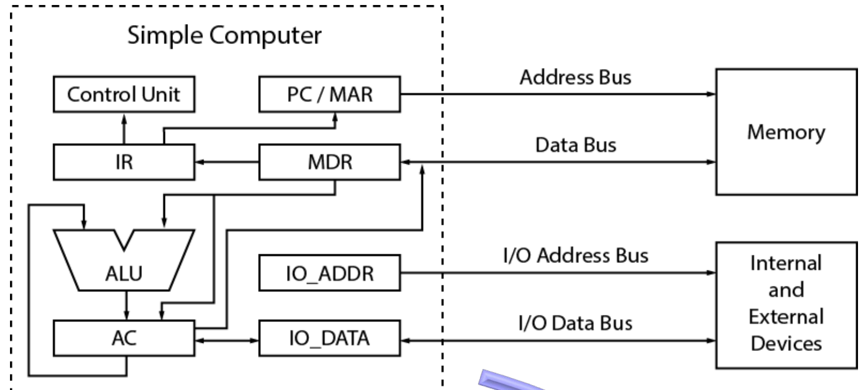
Open-ended Design

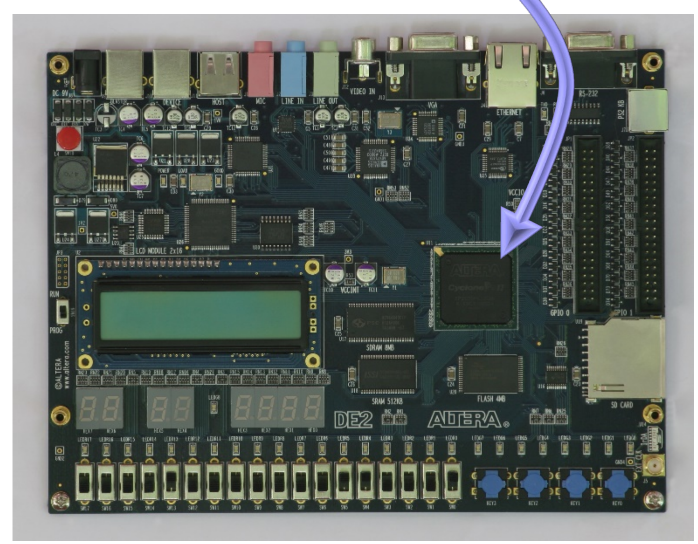
- Recall the very first lecture in 2031, about the design process
  - You will only need to do a little empathizing, but you will go through every other step
- You will also experience many other aspects of open-ended design
  - We will use some of today or next week to talk about limitations, debugging, and efficient design



The Simple Computer (SCOMP)

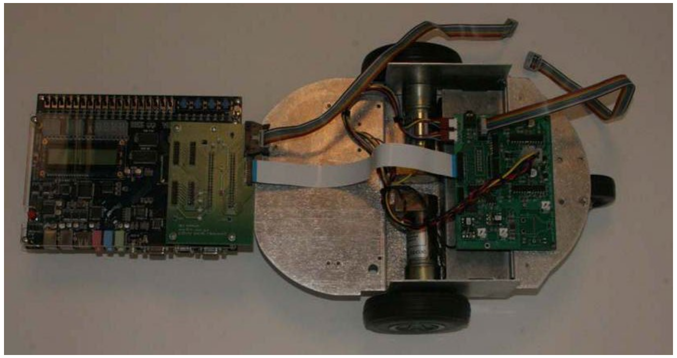
- In Labs 7 and 8, you implement a computer in the FPGA
  - Both hardware (a description of gates, flip-flops, etc. that form the computer)
  - And software (various programs that you load onto your computer, in its RAM)





Background on DE2Bot

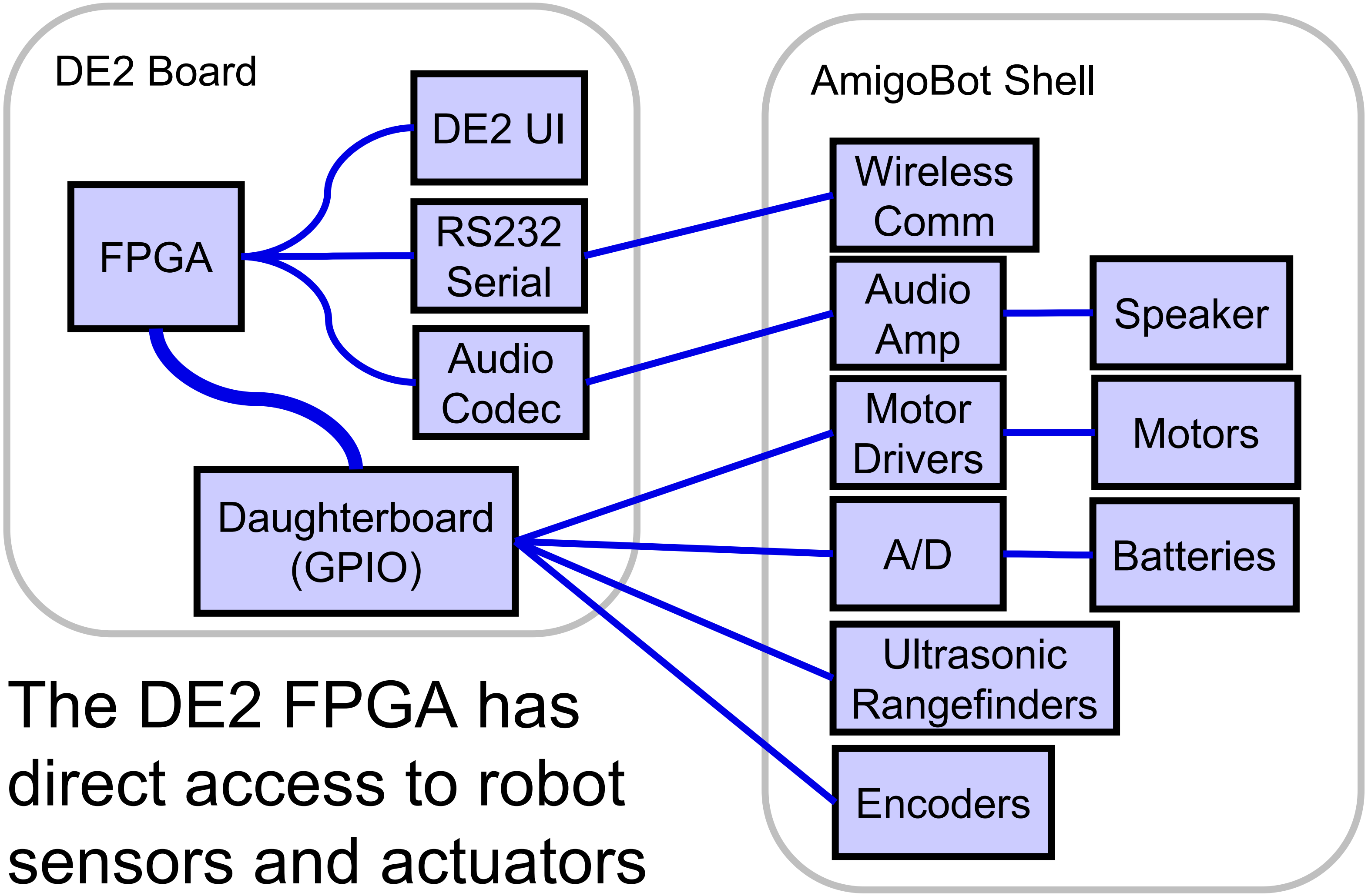
- Many semesters ago, older lab robots were gutted, adding a new internal controller board and a connected DE2 on top
- Then, each semester a new capability has been added, or a new application has been demonstrated



ECE2031 DE2Bot Past Projects

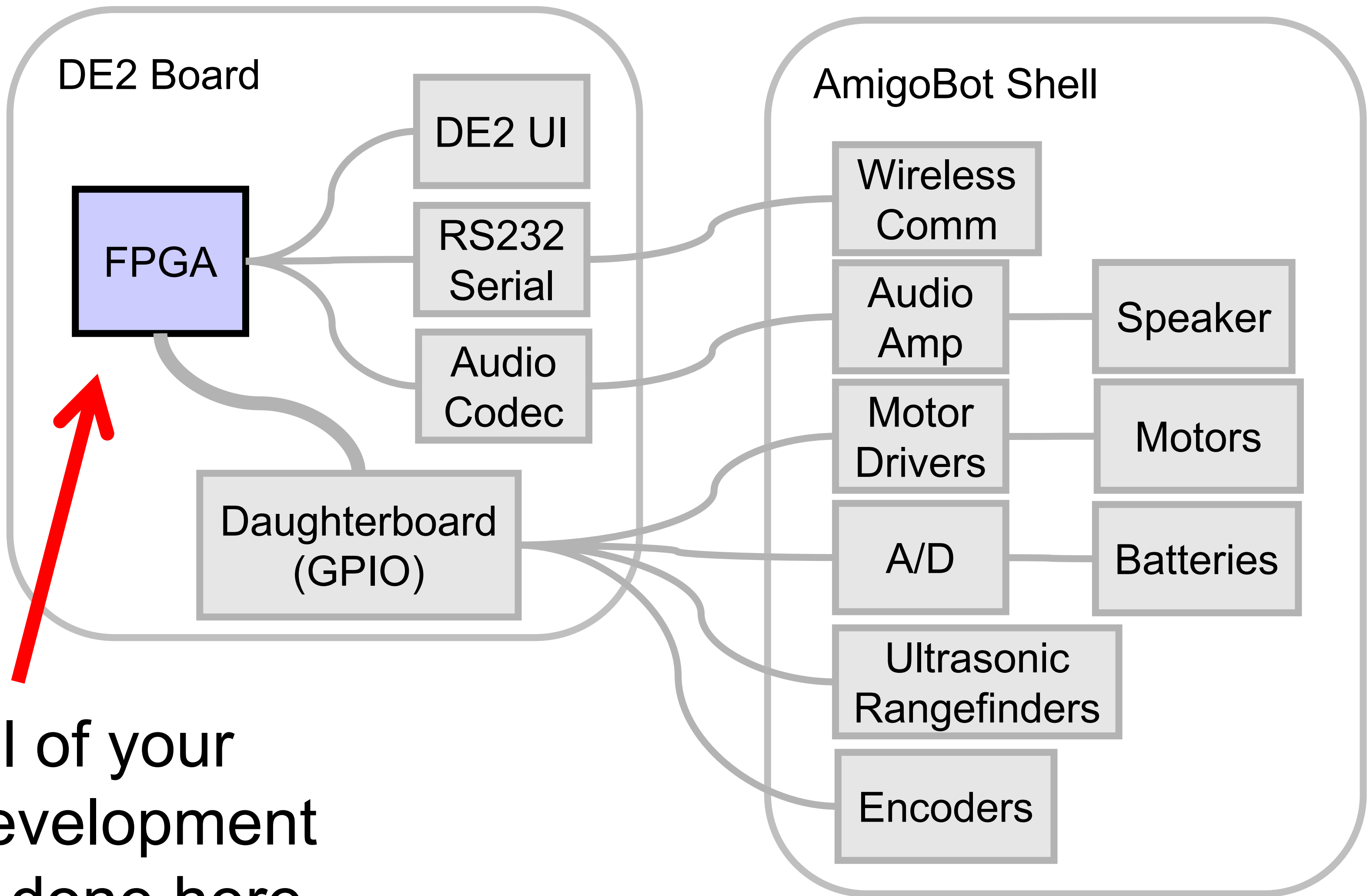
- Position/velocity feedback, and motor velocity control
- Processing of sonar transducers
- I^2C interface for battery monitoring and audio codec control
- Odometry (position estimation from wheel rotation)
- Audio codec interface and digital sound generation
- Robot Self-test
- Infrared signal detection and “remote control” demonstration
- UART for serial communication
- Implementation of hardware interrupts for SCOMP
- Complex mathematical functions in software (ATAN)
- Analyzing sonar data to locate objects and make contact
- Many sensing, localization, and navigation demonstrations

DE2Bot Hardware Architecture



- The DE2 FPGA has direct access to robot sensors and actuators

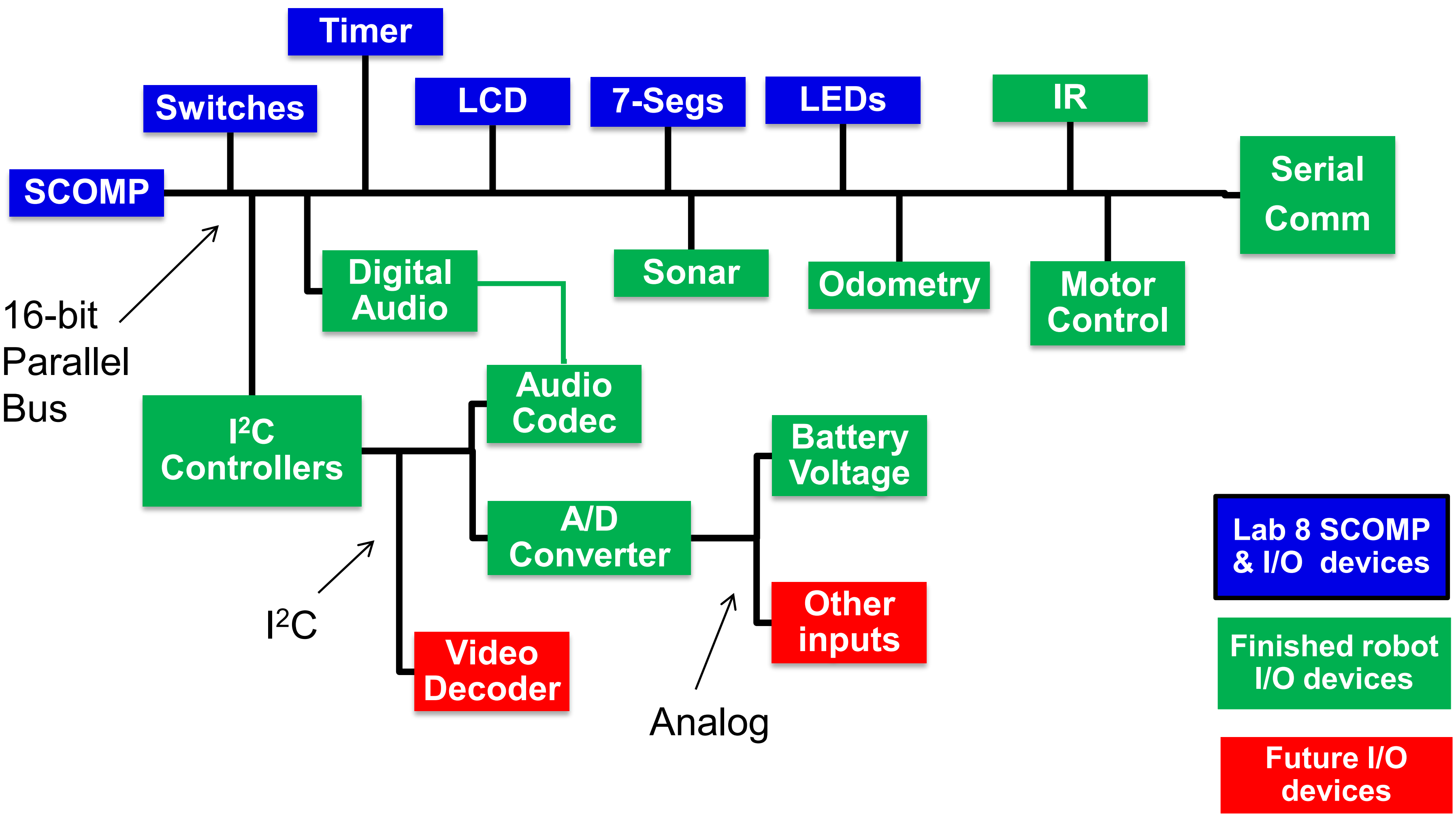
Project Development



All of your development is done here

DE2 and FPGA System Architecture

This will be clearer after Lab 8



Working with Real Systems

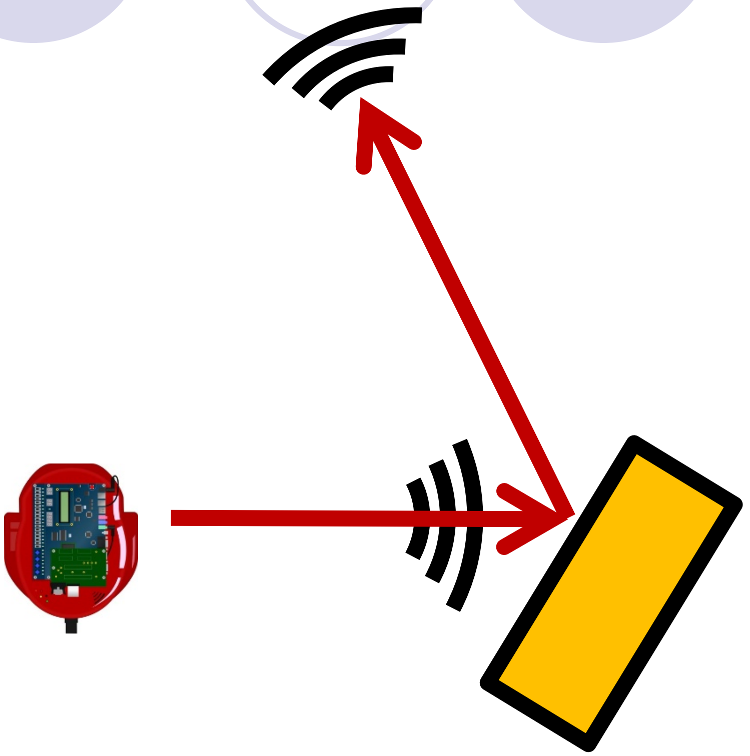
- Part of this project is interfacing with real systems and interpreting real data
- The robot is <u>imprecise</u> – turning 90° might actually be 92°, and moving 5 ft. might be 5.2 ft.
- The data will be <u>messy</u> – it is low-resolution and it will have noise and gaps
- That's part of what makes this project interesting! This isn't a contrived situation, and there is no perfect solution.

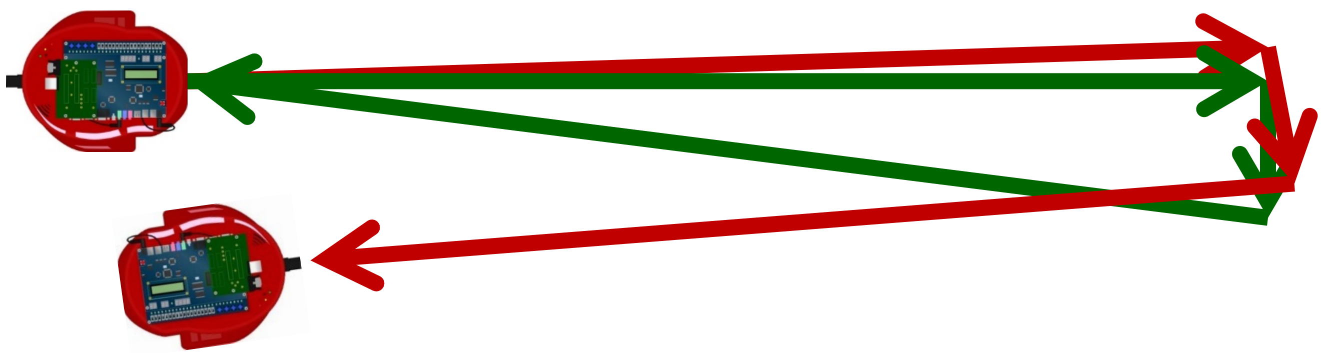
Acknowledging Limitations

- Engineering is a process that involves tradeoffs and limitations
- You never have unlimited speed, silicon, space, memory, time, money, data, people…
- In this project, wishing for different sensors, a smoother floor, more time – is not useful
- The limitations are *the same for everyone*, and it's your job to implement the best solution you can within those confines

DE2Bot considerations

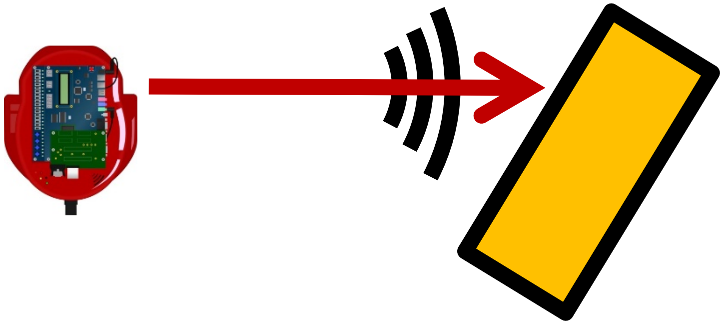
- An upcoming online activity will describe odometry and sonar
- Both are likely useful
- Both have their limits





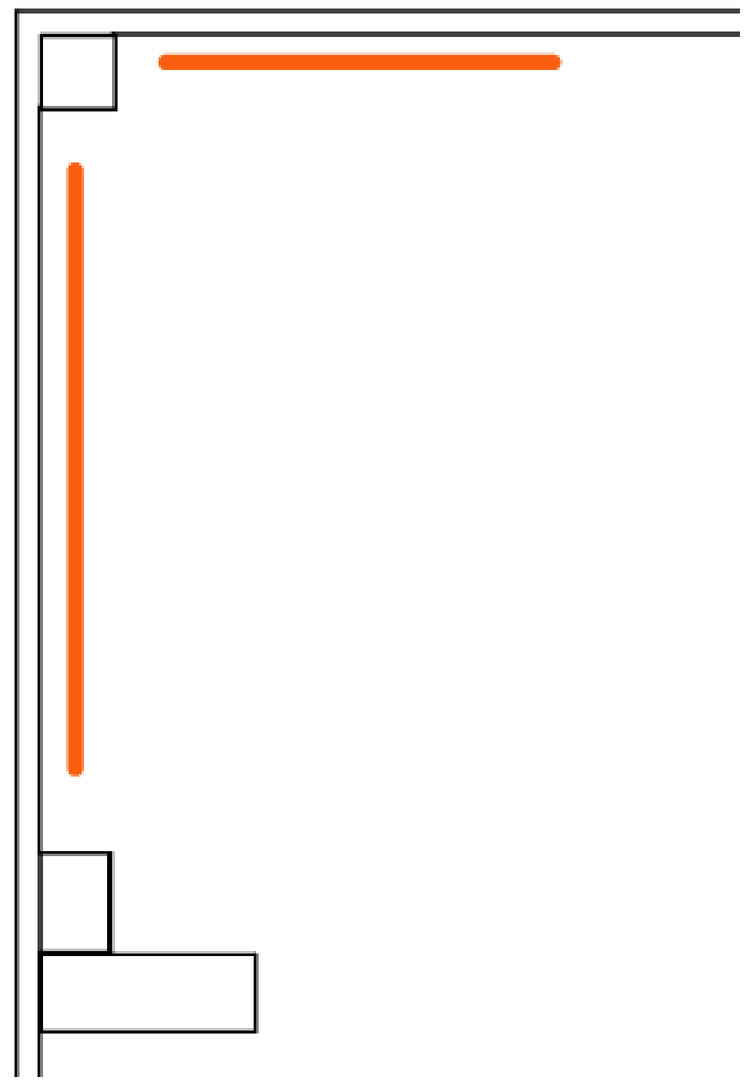
Pros and Cons of Sonar

- Interface is simple
  - 8 sensors, with a single numerical reading from each
  - A reading is just a distance in the direction of the sensor (it is not an x/y position in the robot's coordinate system)
- Objects may be missed
  - Usually because of reflection from oblique surfaces
- But sonar is the ONLY way to get information about the surroundings



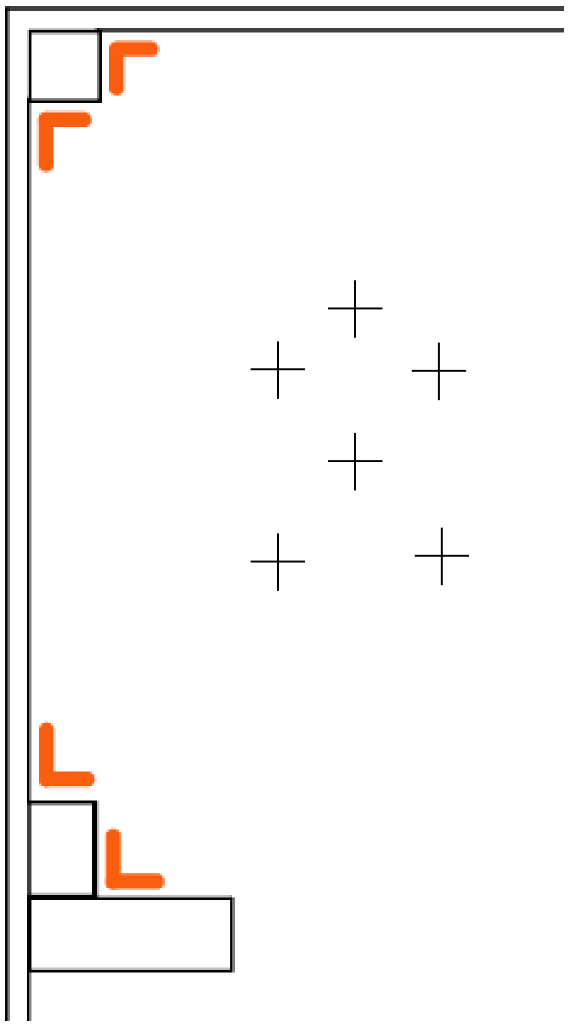
Using sonar

- There are some stretches of flat, orthogonal surfaces
- These may show up clearly in sonar data
  - But only if a sonar transducer is roughly perpendicular
  - And at a distance greater than the minimum sonar reading (~15 cm)



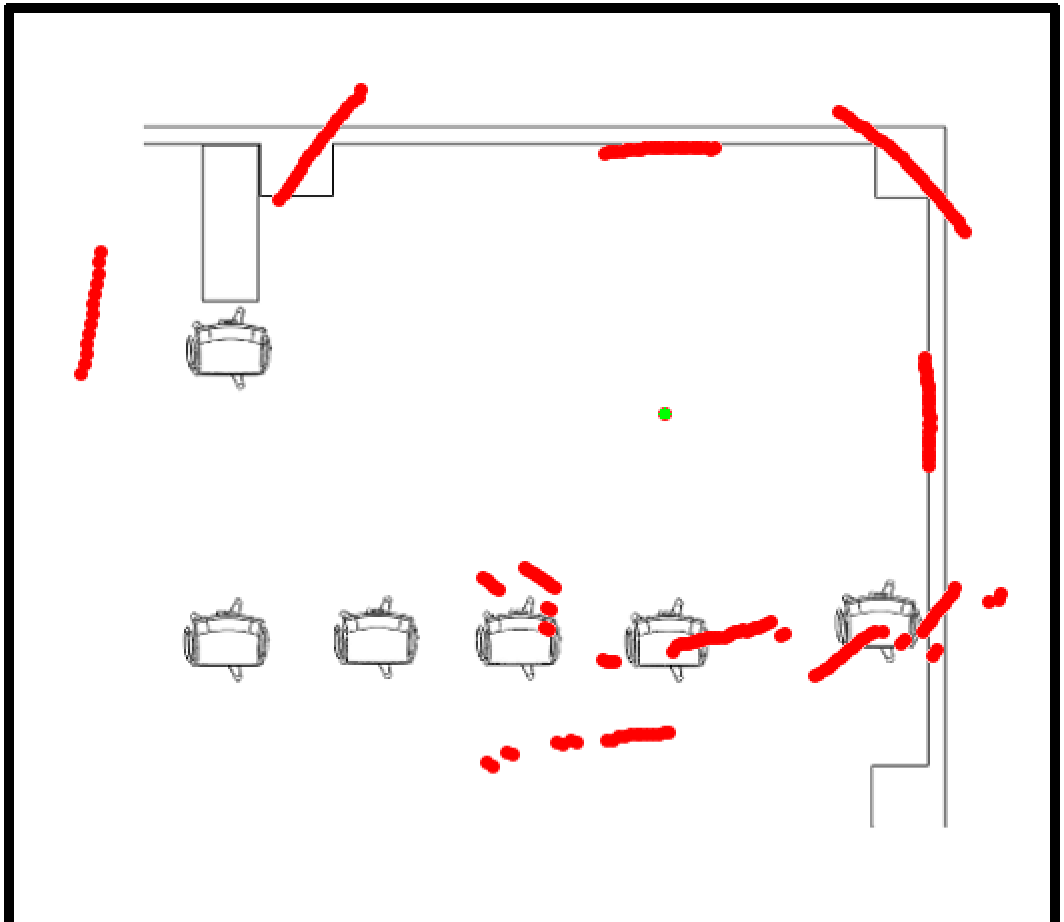
Sonar – corners

- Inside corners make great reflectors (see: corner reflector)
- But they do not look like “corners”, due to sonar beam spread
- This is a major consideration this semester – we will be using corner reflectors

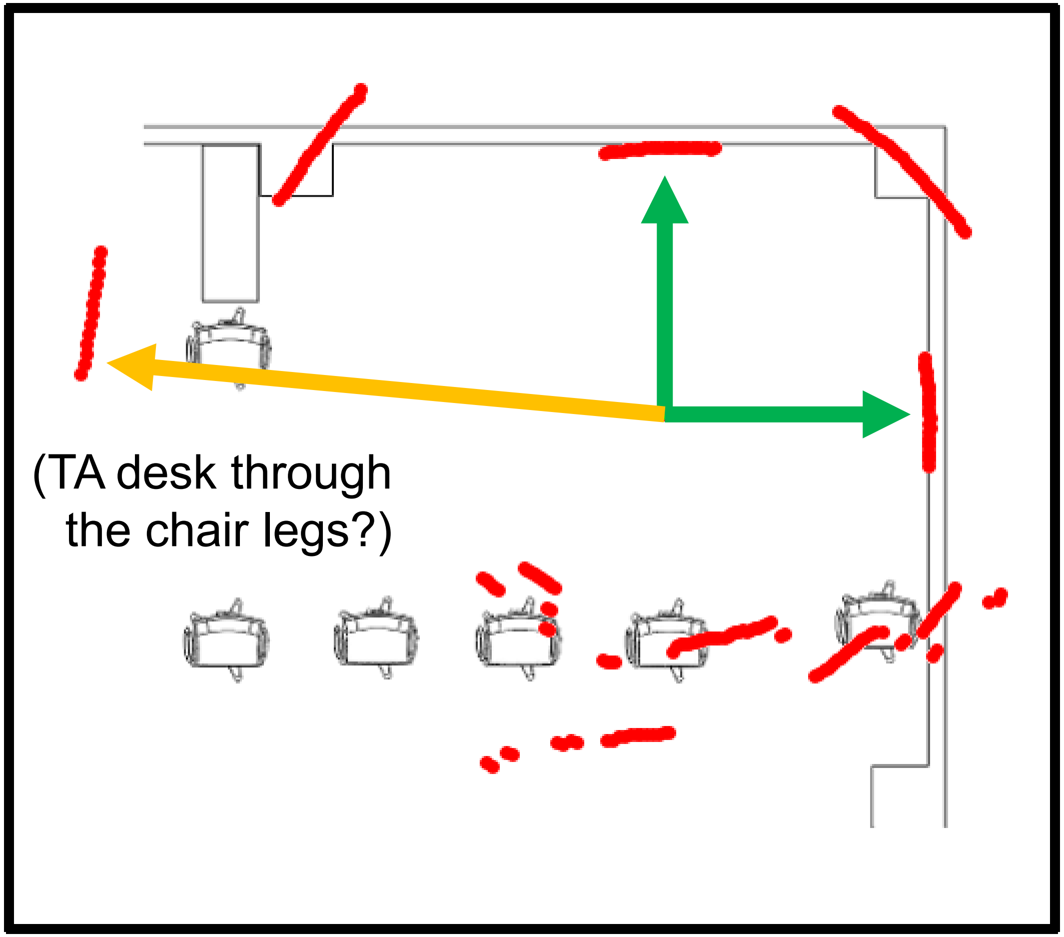


Example Sonar Data

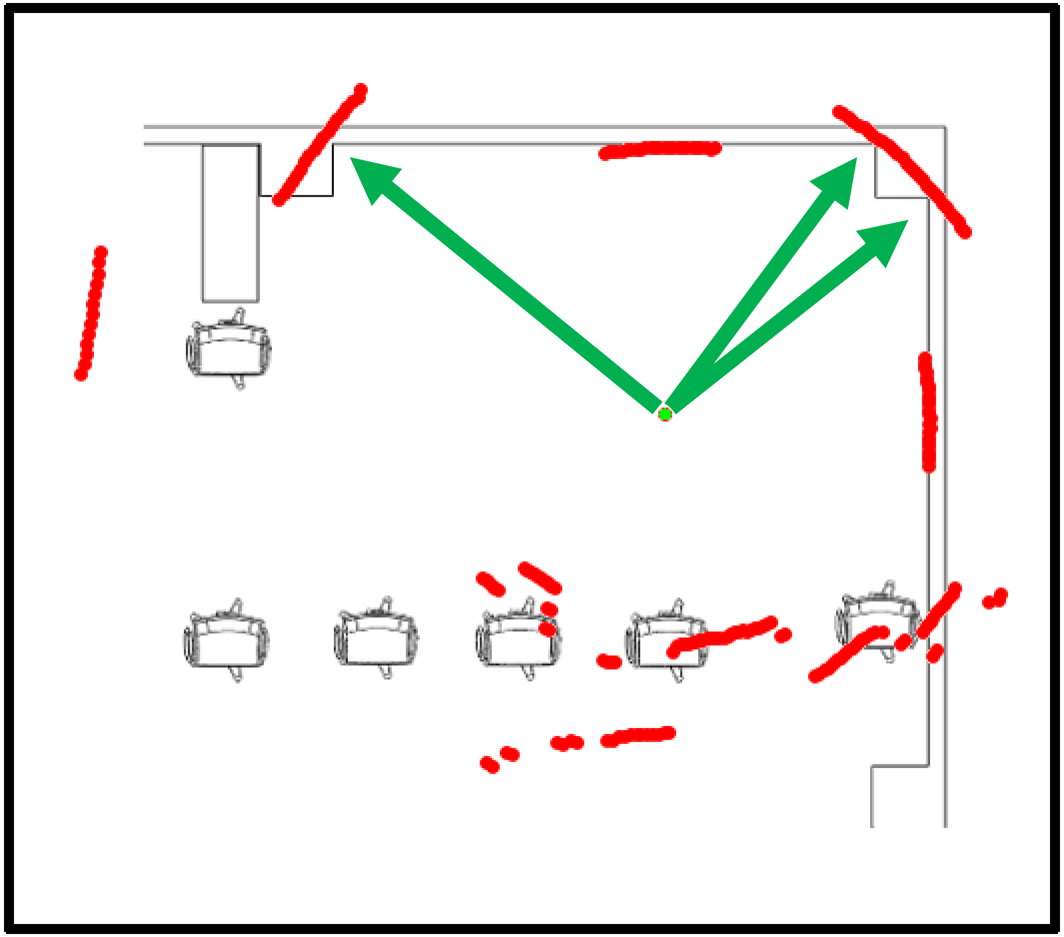
- One sonar value stored for each of 360 degrees, as the robot spins in-place
- Green dot is robot location



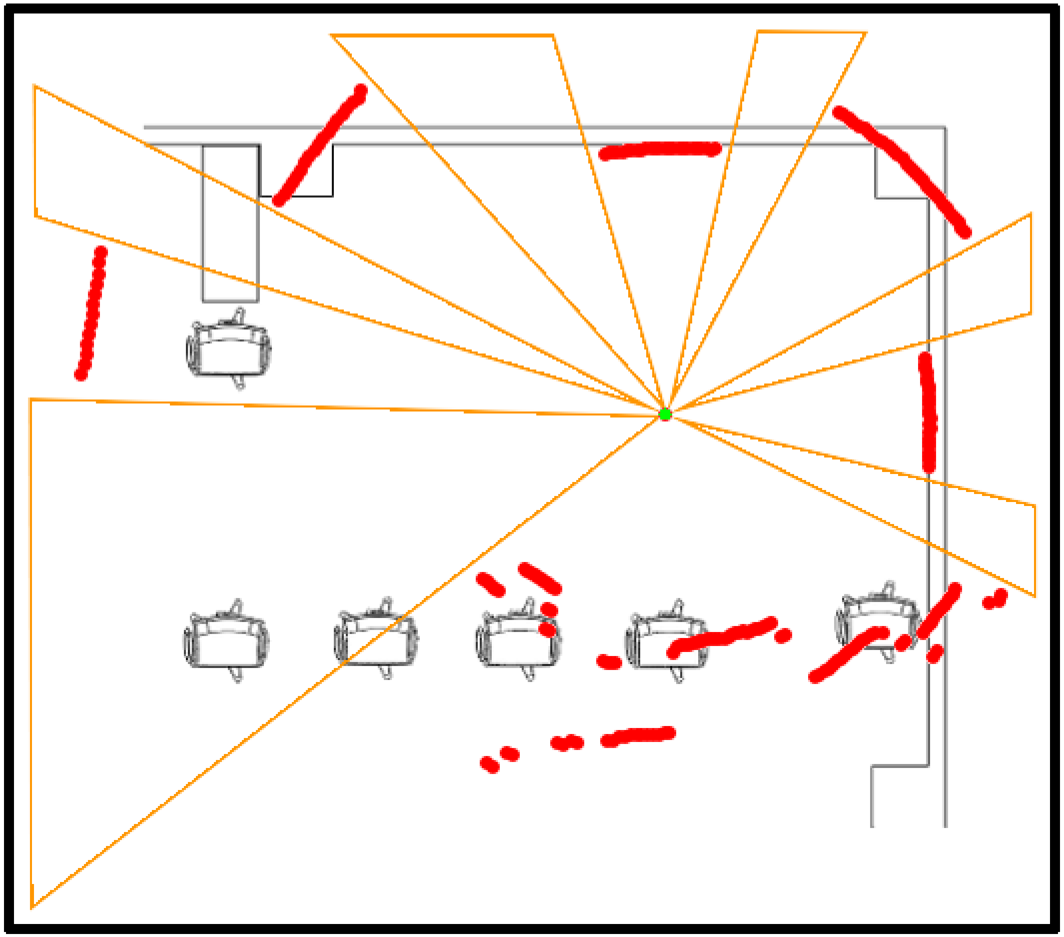
Perpendicular Flat Walls



Corner Reflectors



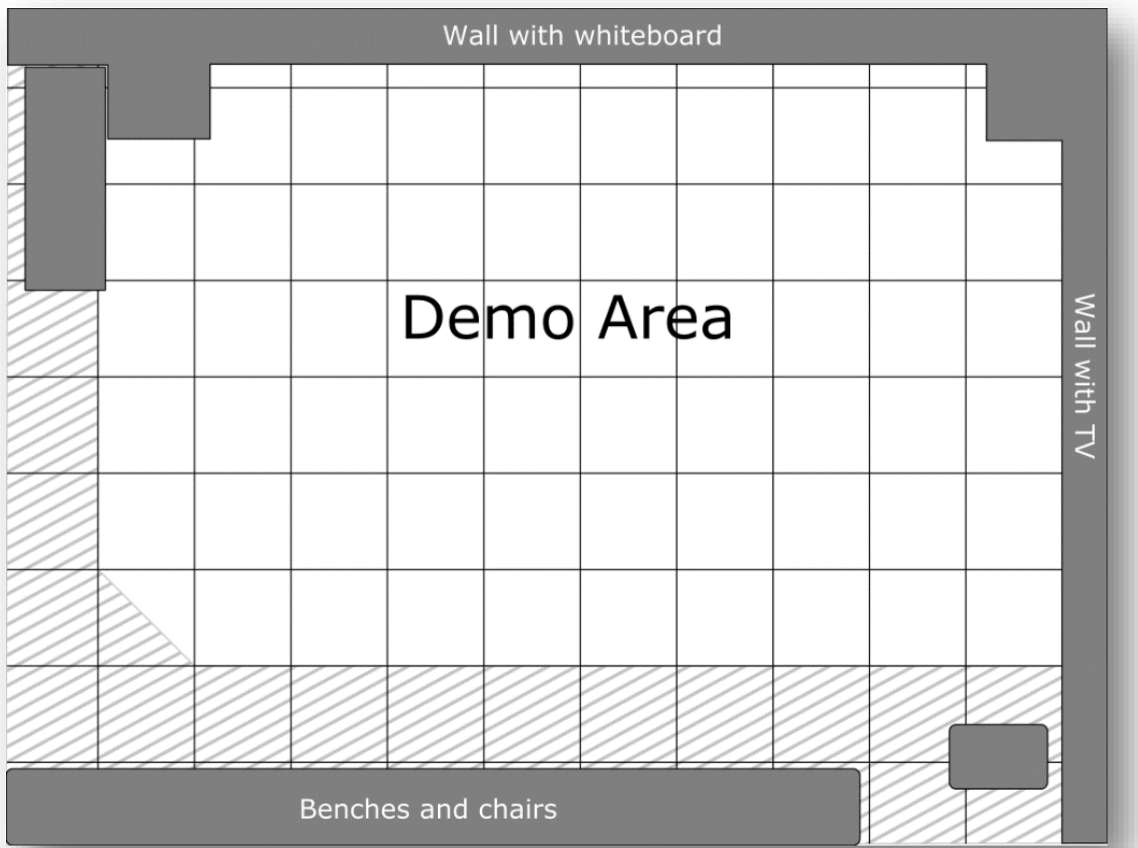
No Data



The robot did scan in these directions, but it got no return ping

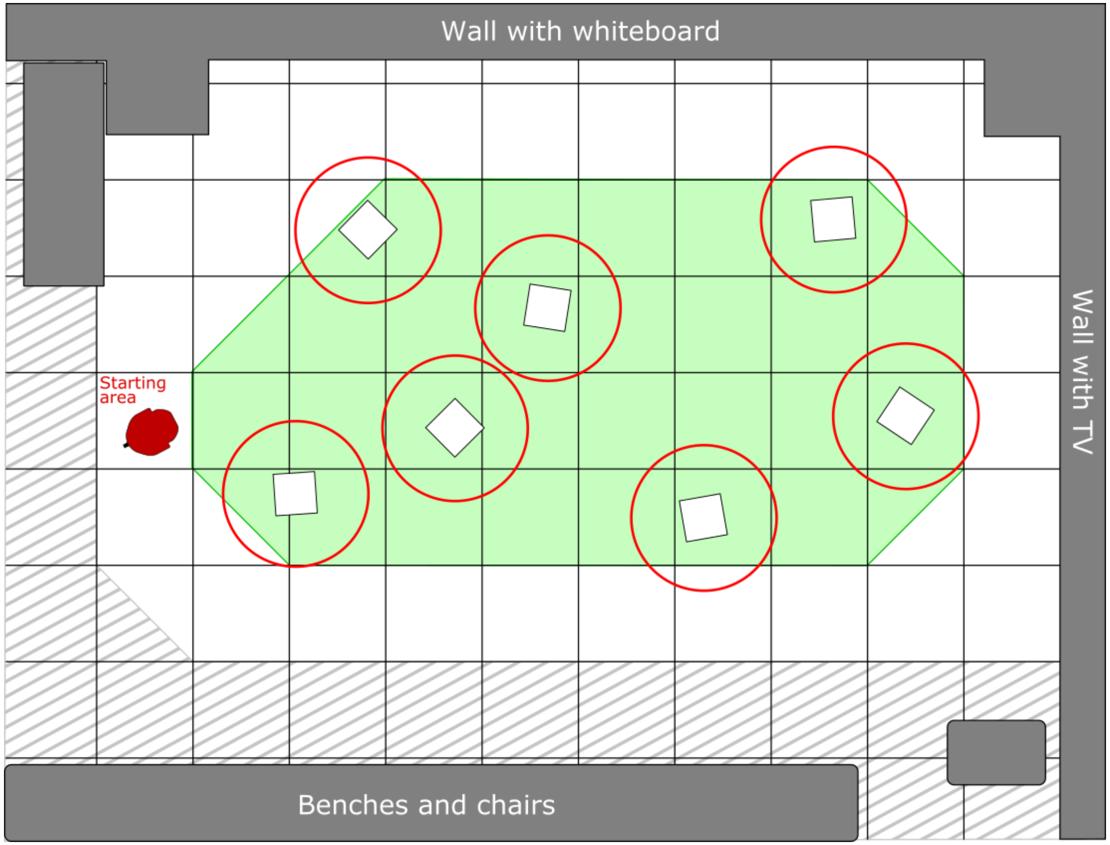
Your Design Task for Fall 2019

- Given a scattering of seven sonar reflectors in the area between the TA desk and the TV, have the robot find and circle as many as possible in two minutes



Demo Area Details

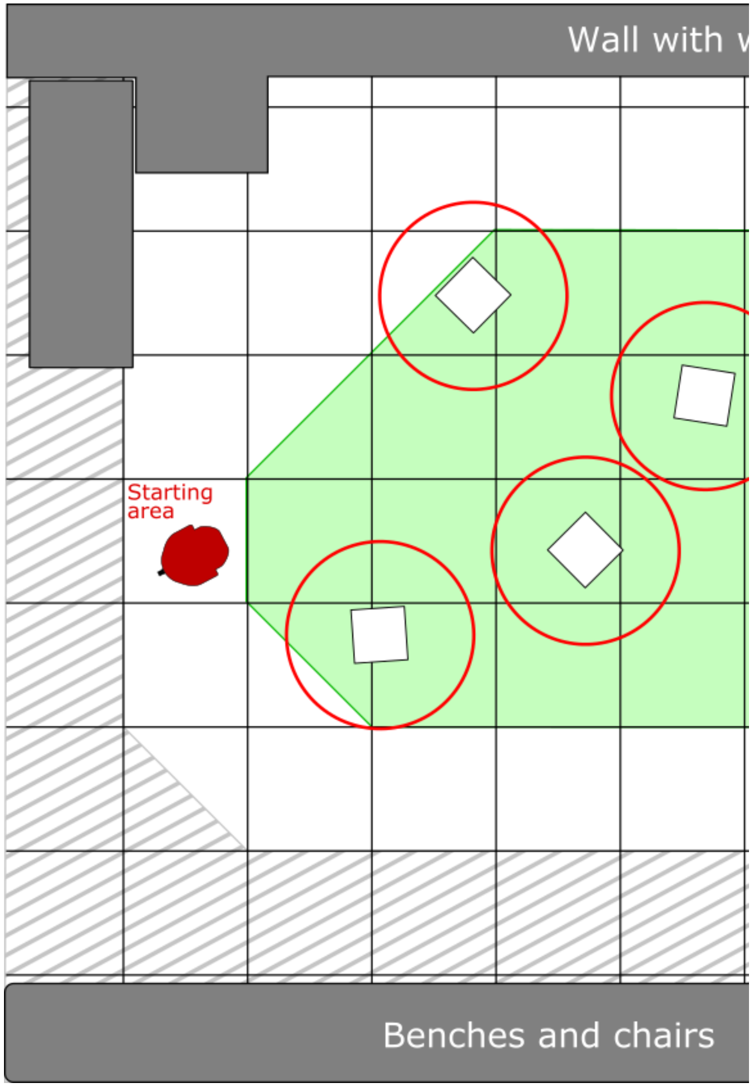
- Reflectors are shown here as squares. This is an EXAMPLE layout. Placement will vary.
- All reflectors will be:
  - in the green area
  - separated by a minimum of three feet (shown here with 1.5 ft. radius circles)
- The white and green areas will be free of obstacles (no people, other robots, etc.)



Demo Area Visualization

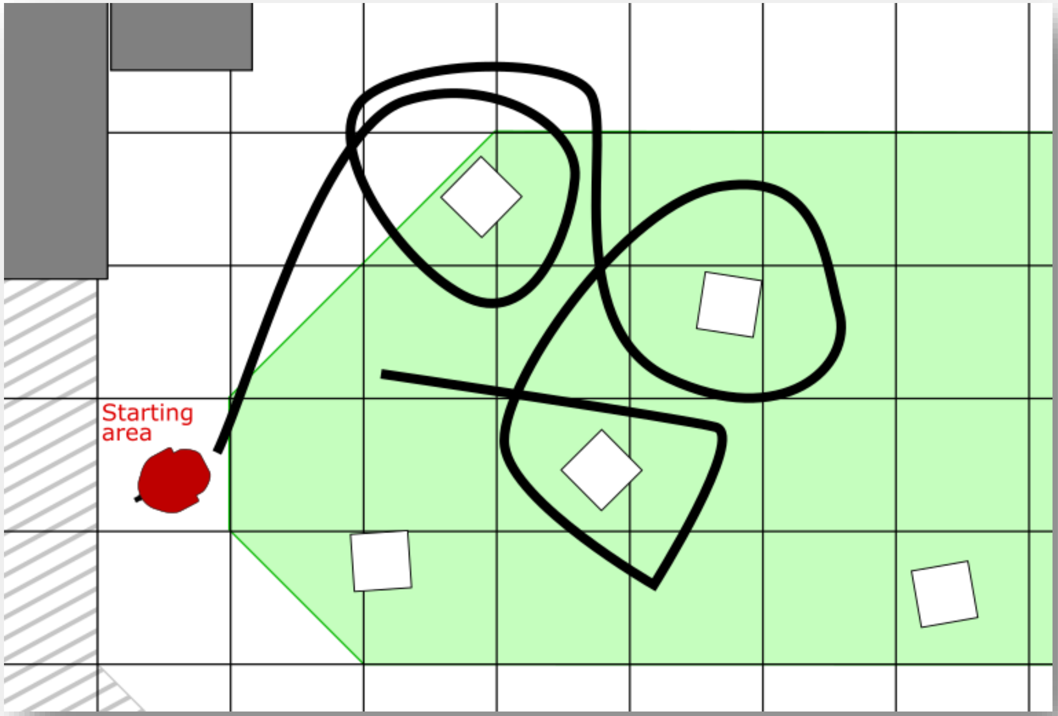
Starting Conditions

- The robot can start anywhere in the 2’ grid square shown in figure
- A member of your team will place the robot and start it (by pushing a button on the DE2)
- Run time begins as soon as you activate the robot



Circling reflectors

- Circling requires completing a circumnavigation
- Only one reflector per circle
- Intent-to-circle must be reasonably believable
- More details are in the project description document.



Resets

- During a run, your team can stop the robot and manually place it back at the starting location
  - Whenever you want, as many times as you want
  - You can reset SCOMP, reprogram the DE2, whatever
- If you restart you lose all current scoring information, both positive and negative
- Clock does NOT pause or restart
- Before demos, think about what situations are worth resetting for so you don’t waste excessive run time deciding during a run

Demo Scoring

- Positive points for circling reflectors
  - Significantly more points for the first time each reflector is circled, but some points for circling the same one multiple times.
  - More points for circling reflectors earlier in the run
- Negative points for colliding with reflectors or walls
- The lowest score from the three runs will be discarded
- More details in the project description document

Design Space (factors that drive design choices)

<u>Reliance on odometry</u>:

- How long can you rely on odometry?
- Is odometry useful at all?

<u>Integration of sonar and odometry</u>:

- Can sonar inform/update odometry, and/or vice versa?
- Can odometry assist with optimal use of sonar?

<u>General</u>:

- Speed and type of robot movement (e.g. curves vs. stop-and-turn)
- Use of one sonar vs. multiple sonars
- Effort dedicated to not repeating reflectors
- Other factors?? (That’s why you brainstorm)

What is reasonable?

- This is deliberately open-ended
- There is no “perfect result,” and no “this score will earn this grade”
  - We don’t know what the results will be
- Do not overreach (and over-propose)
  - Proposing a “perfect solution” is doomed
- DO propose a <u>progressive</u> path with incremental performance improvements

Time management

- Focus on how you can build towards a certain result by completing smaller tasks.
- If you spend normal 2031 time and use that time wisely, you'll end up with an acceptable project
  - Typically, most 2031 projects are split A/B, with only a very few C grades
  - A conscientious effort is what we expect, and no more time in lab than you would normally spend (splitting effort among the team)

Project tasks vs. tasks in Labs 1-8

- Replace time spent on prelab work with preparation
  - Adjust your overall plan according to what you've finished and how much time you have left
  - Plan how you will use robot time in lab
- Use lab time to make incremental progress and work on deliverables
- The time previously spent assembling lab results can be used to assess current progress

Effective use of lab time

- Do not all work on one piece of code
  - One or two team members can code
  - One or two can run tests
  - Some can work on deliverables like the proposal
- Open hours are still available
  - Offered for convenience, not because we require you to use them
  - Maintain a balance between this class, other classes, and personal time

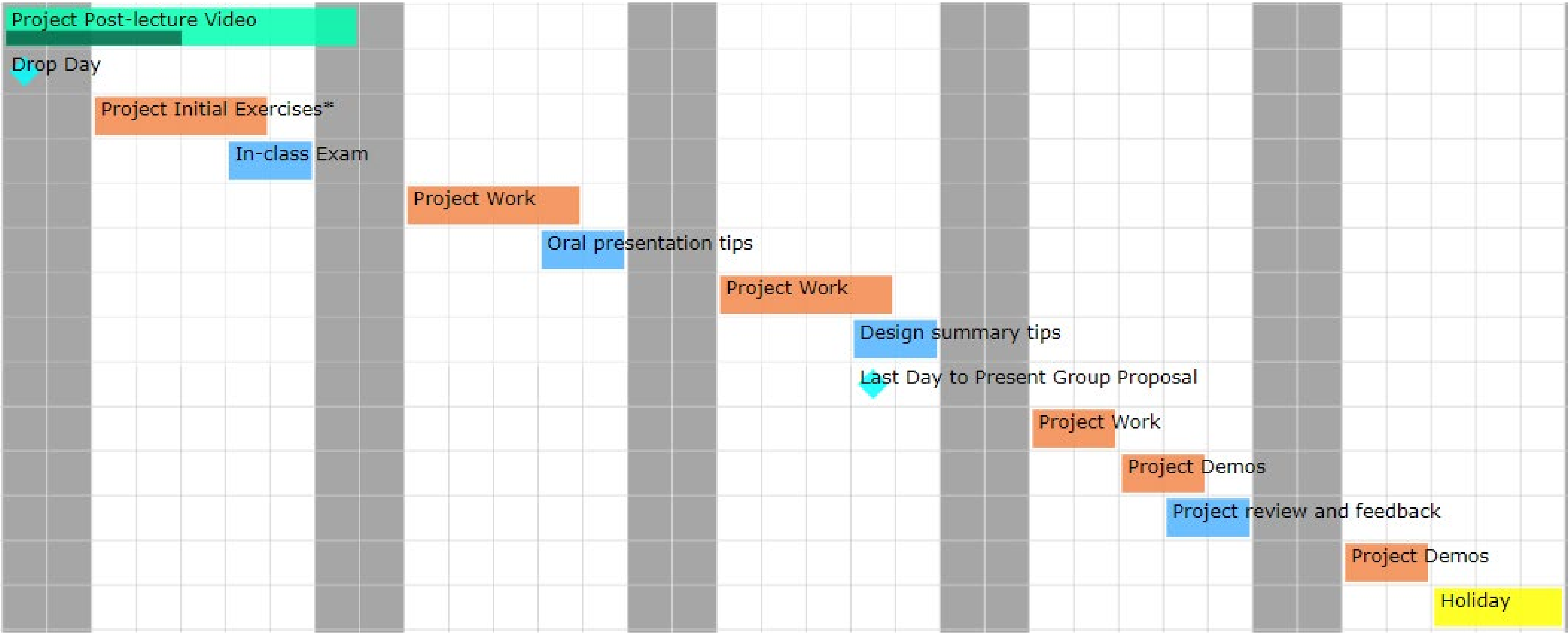
Robot Logistical Details

- ONE robot per team
- Check out robots with a BuzzCard
  - You will get a long USB cable as well
- During your lab section, you will always have a robot available to you
- During open hours, robots are first-come first-served
  - With time limits imposed if robots are in high demand

Project Phases and Key Dates

- Prelab project preparation (before lab next week)
  - Some activities on Canvas.
  - Start your logbook, which you will maintain throughout the project.
- Introductory exercises (next week in lab)
  - Form project groups and discuss ideas
  - Complete some guided tasks
- Proposals presented by Nov. 14th
  - Incorporate brainstorming ideas into a polished presentation
- Complete your design by Tuesday, November 19th
  - You will not be able to work in the lab after this day
- Final demonstrations in lab November 20th – 26th
  - Demonstrate your solution in your section
- Final design summaries due December 3rd (last day of classes)

Project Schedule



Clarifications

- Additional announcements and clarifications will be posted **on Canvas or Piazza**
  - You are responsible for information posted there
  - Could include changes to rules or assignments
  - Make sure you are monitoring it!
- Use Piazza to ask questions
  - If a general question is asked, everyone can benefit from the answer
  - If your question contains details specific to your design, you can limit the visibility to only instructors
  - Especially if you think your idea might be against the "spirit" of the project, ask us about it.

Bottom-up Design

- More possible sources of problems means more difficult debugging
- It's in your best interest to build and test small pieces that you can then integrate
  - Each piece is easier to test and debug
  - Pieces can be used in multiple places, simplifying later stages of the design, and providing more options for "plan B"
  - Changes and improvements to individual pieces can benefit the whole design

Monolithic Example

- End goal: traverse a square
- Move forward for some distance, stop, turn, move, stop, turn, move…
- What happens when you need to change the path?
  - Go through every step of the routine changing values and copy-pasting sections of code

Bottom-up Example

- End goal: traverse a square
- Bottom-level components: turn to face a destination; check if at a destination
- Mid-level component: Move forward while turning towards destination as needed and stopping when at destination
- High-level integration: Each time a destination is reached, change the destination and repeat