Making Your Own Embedded Linux-Based Robot

All Your Base are Belong to us?

Mike Anderson
Chief Scientist
The PTR Group, Inc.
http://www.theptrgroup.com

What We Will Talk About

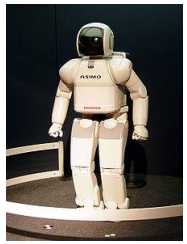
- What is a robot?
- Basic services
- Mobility issues
- Sensors
- Robot software frameworks
- Summary

So, You want to make a Robot?

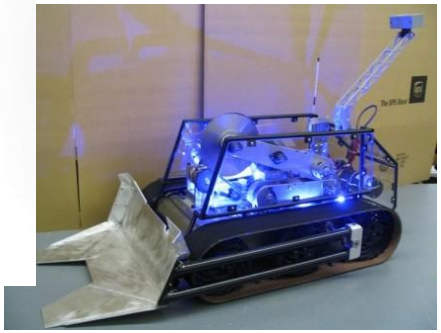
- First, we need to know what a robot is…
  - Generally, they are defined as a mechanical intelligent agent
  - Today, these are typically electro-mechanical devices which are guided by computer or electronic programming
- To many, robots are capable of autonomous behavior
  - Others feel that tele-operated devices can still be considered robots

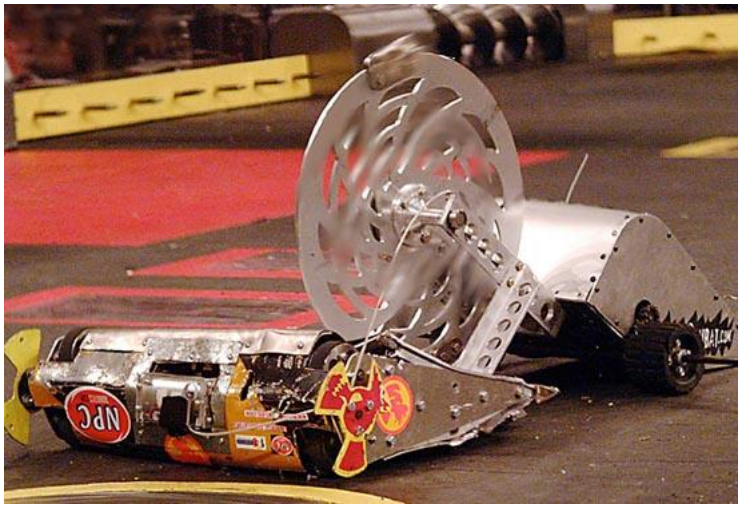
Classes of Robots

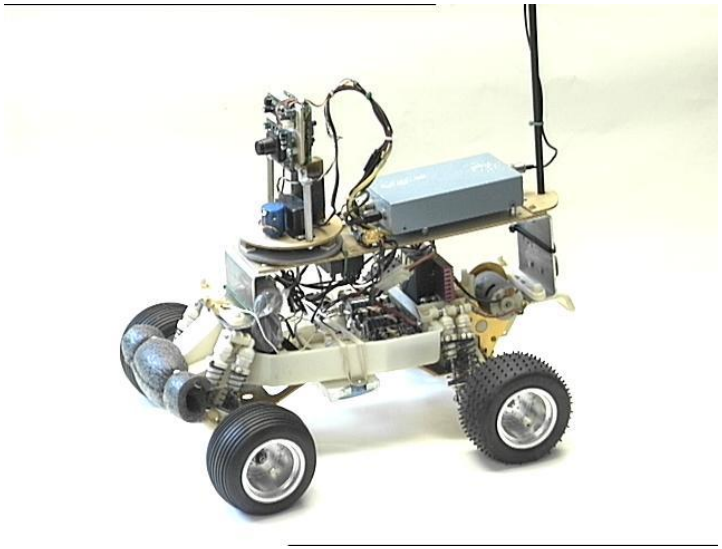
- Today, robots are often divided into different classes based on their use or behavior
  - Mobile robots
    - Robots that are not fixed in place
  - Industrial robots
    - These robots are typically fixed in place and consist of a jointed arm and an end effector
  - Service robots
    - A semi- or fully-autonomous device that performs services useful to the well being of humans



Other Robots







General Characteristics

- Regardless of how you picture robots, there are several characteristics that all of them have in common:
  - Assignment or task
    - What do you want it to do?
  - Autonomous vs. tele-op
    - Does it have a human in the loop?
  - Frame
    - What's it made out of?
  - Mobility
    - Wheels? Tracks? Air cushion? Legs?
  - Hardware controls
    - Motor controllers, steering, gears, etc.
  - Sensors
    - How do we sense the world?
  - End effector
    - How does the robot manipulate its world?
  - Power
    - How do we power the computer, mobility and actuator?
  - Software controls
    - How do we control the hardware controls, sensors, actuator, etc.?

Robot Frames

- What do we want to make our robot from?
  - Wood
    - Easy to work with, but heavy and potentially fragile
  - Aluminum
    - Lightweight and inexpensive, but easily bent
  - Steel
    - Very rugged, but heavy
  - Titanium
    - Lightweight, but expensive

- What tools do you have at your disposal?
  - Lathe, CNC mill, drill press, table saw, hand tools?
- We must plan our frame around how much weight we expect

Commercial Robot Frames

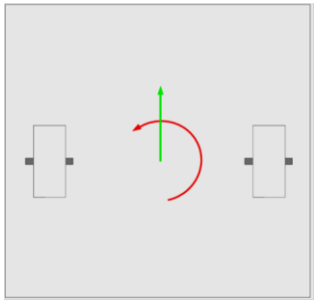
- There are a number of vendors of robot frames and mobility systems
  - http://www.zarosrobotics.com
  - http://www.whiteboxrobotics.com
  - http://www.lynxmotion.com
  - http://www.trossenrobotics.com
- Buying a commercial robot base can save a lot of time and trouble

Choosing the Mobility System

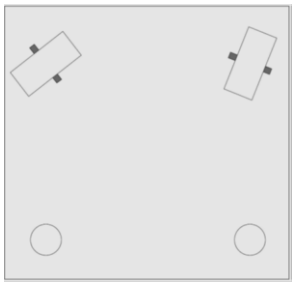
- Should the robot be wheeled or have tracks?
  - Tracks give great traction
    - Good climbing ability
    - However, they're not very fast nor very maneuverable
    - If you throw a track, your robot is DITW
  - Wheels can be 2/4/6WD
    - Fast and very maneuverable
    - Climbing can be a challenge
    - One or even two motors could fail and your robot could still move

Steering Approach

- How will you steer your robot?
  - Skid steer (tank drive)
    - One side moves at one speed while the other moves at a different speed
    - Very simple to implement

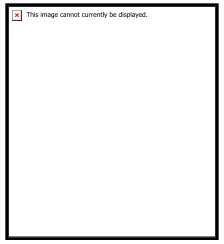


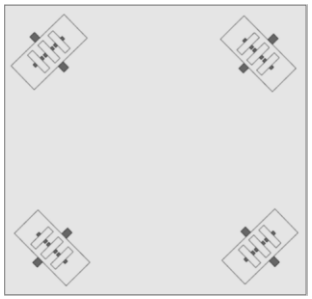
  - Ackerman drive
    - Like your car
      - Requires planetary gears to deal with going around corners
  - Swerve drive
    - Allows the robot to turn the wheels independently



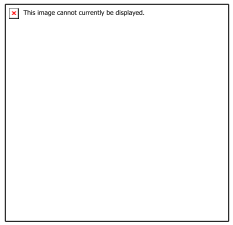
More Steering

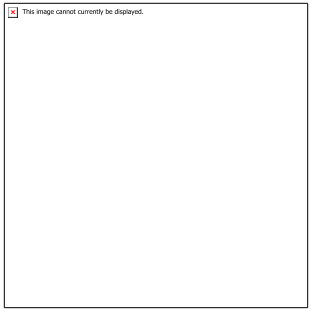
- Holonomic drive
  - Can move in any direction
  - Limited pushing power
  - Limited climbing ability





- Mechanum drive
  - Developed for fork lifts
  - So-so on inclines
  - Can move in any direction
  - Software control is more involved





- Which one you pick depends on your requirements

Wheels!

- There are many types of wheels for different applications:

Motors

- In most cases, you'll be using DC motors
  - Brushed and brushless
  - Voltage requirements range from 5V to 24V DC
- Motors can come with gear boxes attached or you can supply your own
  - Gear ratios vary depending on torque vs. speed requirements

Delivering Power to the Wheels

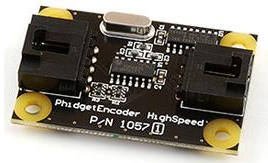
- There are several ways to deliver power from the motors to the wheels
  - Direct drive via a gearbox
    - Least power loss
    - Motor/gearbox weight may be a problem
  - Belt-driven
    - Very lightweight
    - Belts can come loose
    - Possibility of slippage
  - Chain-driven
    - Good middle ground between belts and direct drive
    - May need chain tensioners
    - Chains do stretch

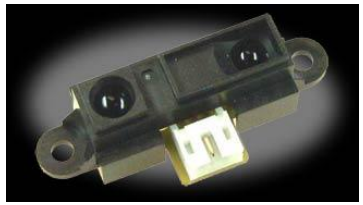
Controlling the Motors

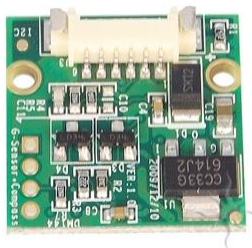
- The speed of a DC motor is dependent on the amount of voltage provided
  - You could vary the speed with resistors
    - But, the heat loss would be significant
  - However, most motor controllers use PWM (RC servo) to vary the speed
  - Make sure that you pick a motor controller that can handle the maximum current draw of your motors
- Typically, we'll use a motor controller
  - Also known as an electronic speed controller (ESC) and is often controlled via RC PWM signals
- Can also be controlled via CAN RS-232, SPI, I2C, or USB

Sensors

- In general, robots must be able to sense their environment
- Sensors can take the form of:
  - Cameras
  - Infrared
  - Sonar
  - Limit switches
  - Shaft encoders
  - Accelerometers
  - Gyroscopes
  - Digital compass
  - Strain gauges
  - Light sensors







- Sensor interfaces can be digital, analog, I2C, SPI, CAN
  - Your computer controller must support the interface

Actuators/End Effectors

- How are you controlling the manipulators of your robot?
- Motor-driven
  - Repeatable if you have shaft encoders
  - Limit switches can also be used
- Pneumatic
  - Fast, but limited options for motion
    - Expanded or compressed but no in between
- Energy can be stored in surgical tubing, springs, etc.

Open or Closed Loop Control?

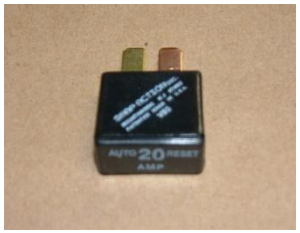
- Will your robot have some sort of feedback?
  - Closed loop control
  - Shaft encoders or other sensors
- Will you simply run for time?
  - Open-loop control
  - Distances vary with battery power
- Or, some mixture of both?
- Impacts how much software you'll have to develop

Powering the Robot

- Speaking of batteries, which should you use?
  - It depends on how long you need to run and how much power your robot pulls
- Basic battery types:
  - Sealed Lead Acid (SLA)
    - Good maximum current
    - Heavy
  - NiCd, NiMH
    - Quick recharge
    - Suffers from memory effect
  - Lion, LiPO
    - Excellent energy density/weight ratio
    - Batteries and recharger are $$$
    - LiPO batteries must be handled with care
      - Shorting out or pulling too much current at once can result in a fire!

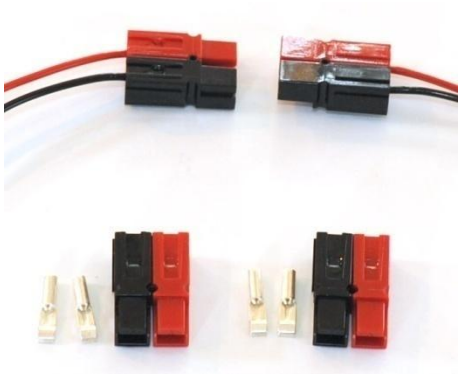
Power Distribution

- You'll need to distribute the power from the battery to the motors, computer, etc.
  - Safety is a key requirement
- Make sure that wire gauges are sized for the current draw
- Make sure all circuits are fused or have fast-reset breakers
- Don't leave bare wires exposed
- Don't attach the battery ground to the chassis
  - Possible ground loops

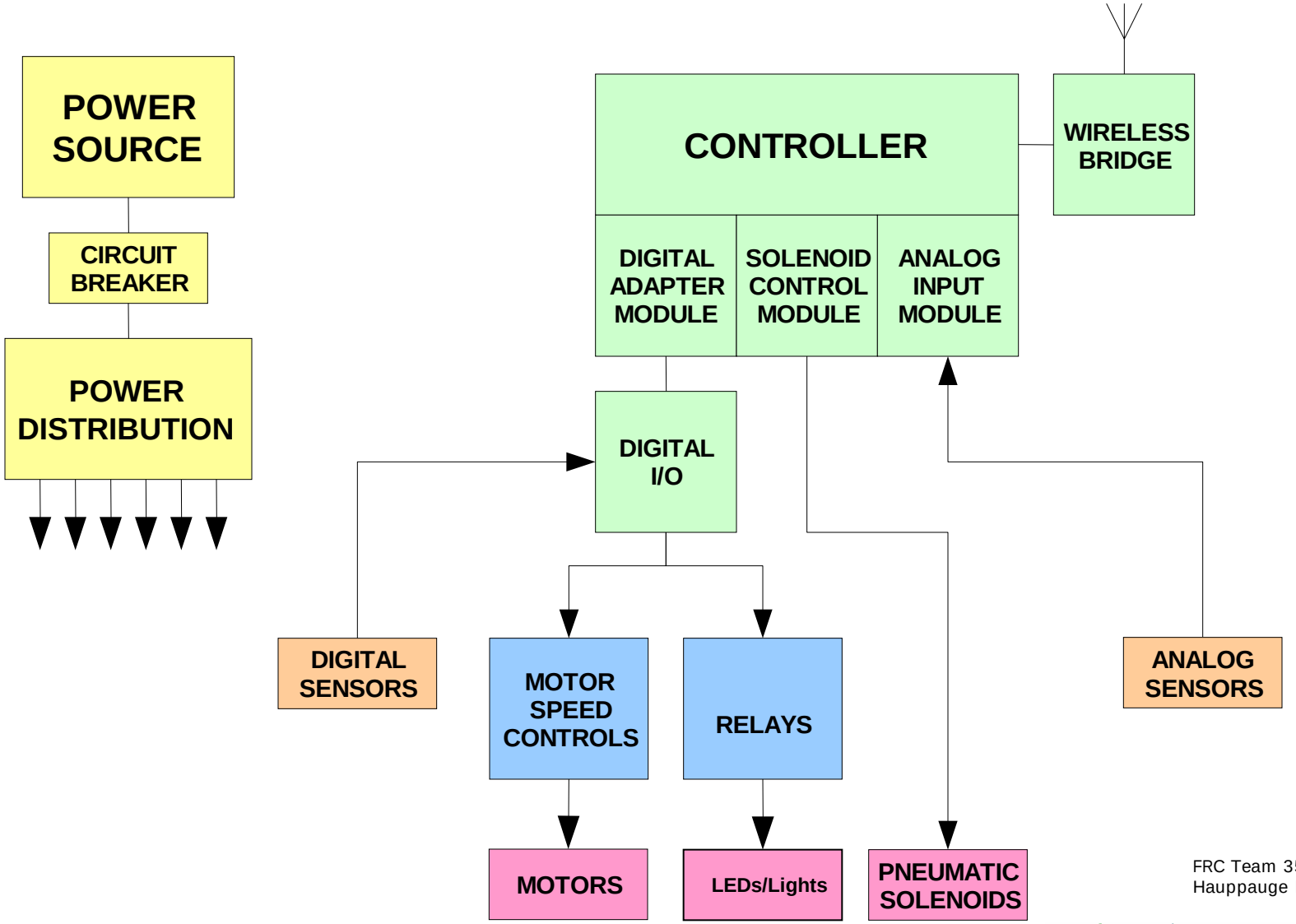


Wire Gauge Recommendations

- 6 AWG 125A
- 10 AWG 50A
- 12 AWG 30A
- 14 AWG 20A
- 16 AWG 12A
- 18 AWG 7A
- 20 AWG 5A



General Electrical Block Diagram

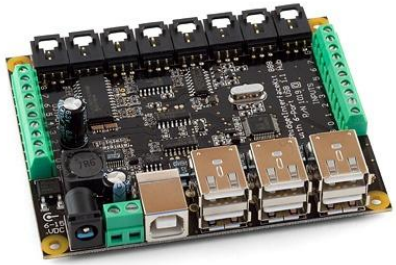


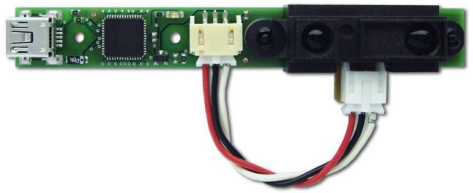
FRC Team 358
Hauppauge H.S.

Computer Hardware

- So, what kind of computer do you need?
  - One that can run from the battery
- x86 or ARM are easy to find
  - Minnowboard
  - Beaglebone Black
  - Raspberry Pi

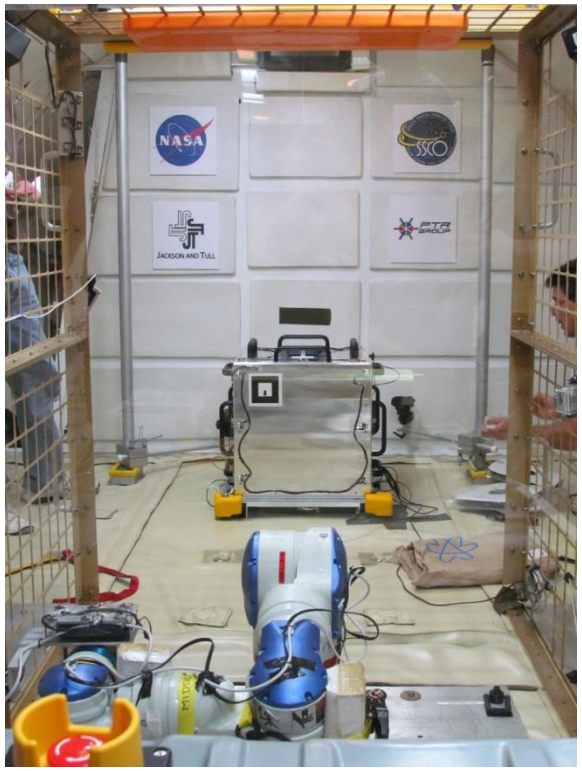
- USB interfaces for I/O
  - Phidgets 8-8-8
  - Toradex OAK sensors





Software Control

- What software will you use to control your robot?
  - Linux, of course!
- Linux has several robot control frameworks
  - Player/Stage
    - Good control of sensors/actuators over IP networks
    - http://playerstage.sourceforge.net
  - Robot Operating System (ROS)
    - Extensive set of software libraries and tools for Ubuntu and Fedora Linux
    - http://www.ros.org
  - Orocos
    - Extensive kinematics and dynamics libraries
    - http://www.orocos.org/
  - Rock Robotics
    - Based on Orocos – but simpler to use
    - http://rock-robotics.org/
  - JAUS
    - Joint Architecture for Unnamed Systems
    - Originally developed by SAE for DoD
    - Used for AEODRS ordinance disposal robots
    - http://www.openjaus.com/
  - Ad Hoc
    - Use Linux to front end devices like Arduinos
    - Ser2net exports serial interface from Arduino to network connection
    - Leverage Arduino ecosystem for motor controllers and sensors



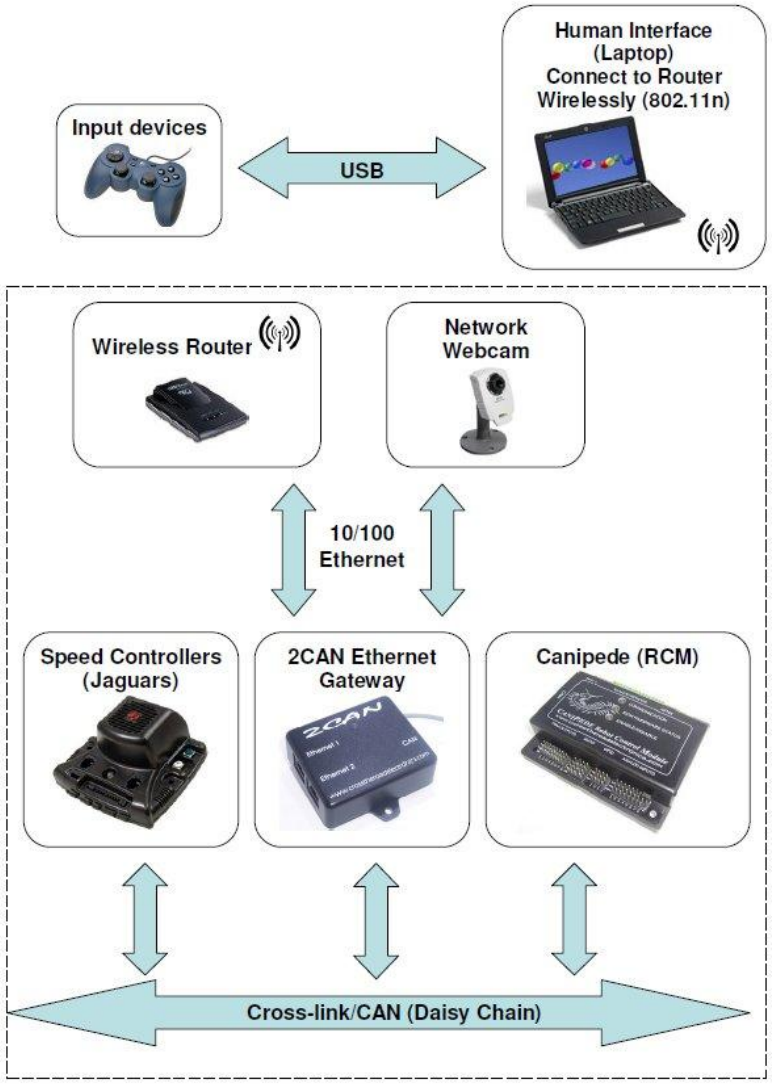
Libraries Abound

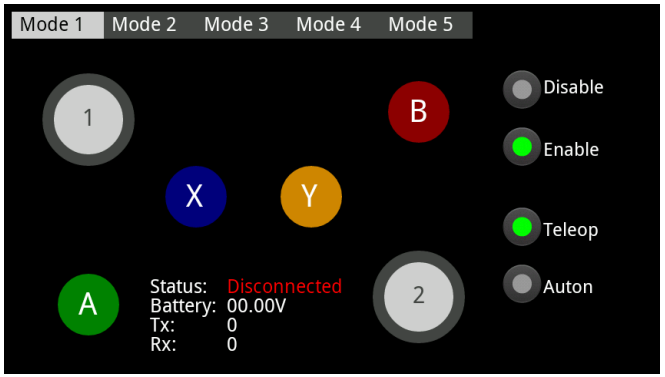
- Linux is a very popular platform for robotics
  - Cheap, fast, efficient
- Many developers have created libraries for the popular robotic interfaces
  - E.g., libkondo4 for the RCB-4 servo controllers
- Manufacturers like Phidgets and Toradex supply open-source libraries for their boards
  - C/C++, C#, Java, Python bindings
- Active user communities exchange code via github, sourceforge and others
  - E.g., check out the Phidgets Linux forum for discussions
    - http://www.phidgets.com/phorum

Software Control #2

- You can also roll your own using jstest and the Linux joystick interface
  - WiFi, Bluetooth or serial RF modem for communications
- A quick check on a search engine will highlight several on-going Linux robotics projects

A Quick Way to Robot Control





- Eight 10-bit analog inputs (5 Volt tolerant) with jumper to measure supply voltage
- Eight PWM outputs for servo and H-bridge motor control (6 volt supply rail)
- Eight solenoid outputs (100 mA per channel)
- Four quadrature encoder inputs for direct quadrature decoding
- Four general purpose input and output
- Four relay outputs for controlling H-bridge or opto-coupled type relays
- Over current/short circuit protection for ALL exposed pins
- User programmable

Linux Robotics Distributions?

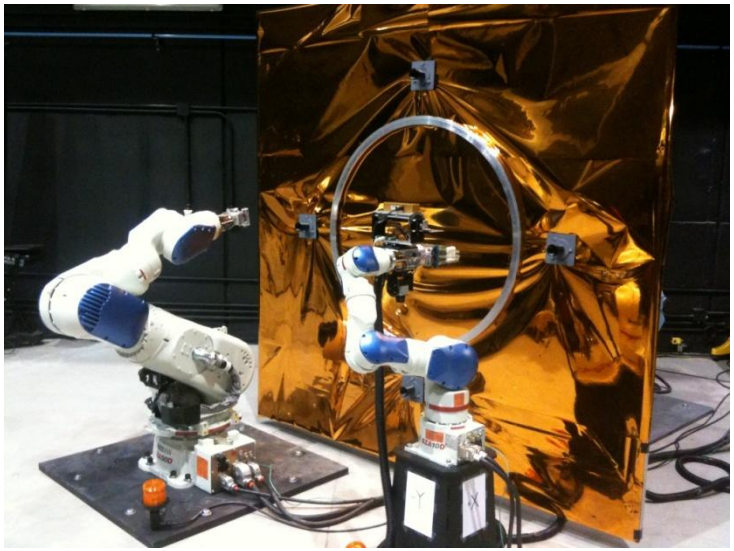
- There isn't an identifiable robotics-oriented distribution at this point
- Many folks simply use Ubuntu on the x86/ARM or Angstrom on the ARM
- Cross compilers for front-ends like the Arduinos and ARM-based platforms are available as standard packages in distributions like Ubuntu
- Installation to SD or CF require the same care as running from flash
  - I.e., Don't enable swap, don't log to /var/log, etc.

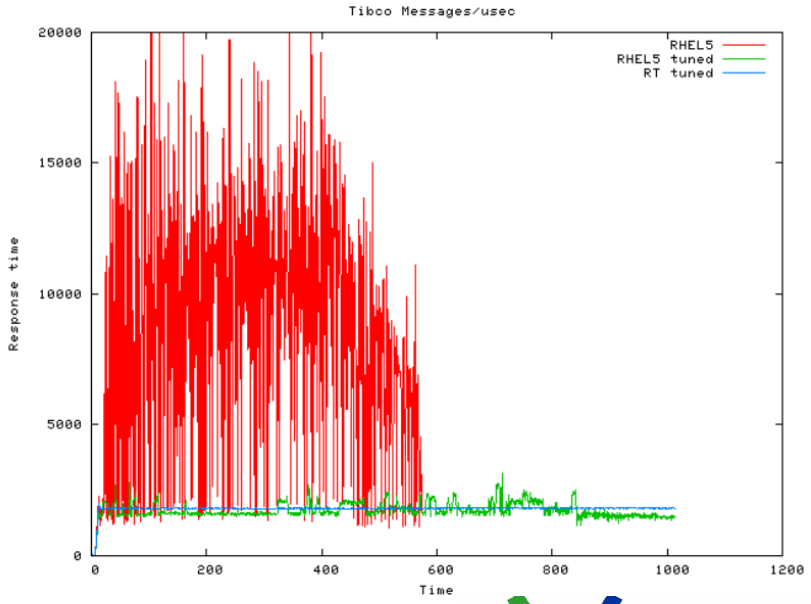
Linux Robotics-Aware Distros

- ROS is a very popular base software for building Linux-based robots
- Fedora and Ubuntu both have ROS repositories
  - Ubuntu is officially supported by ROS
  - Fedora support is via a SIG
- Primary ROS support is on x86
  - Sources have been compiled for ARM

Can Linux Do *REAL* Robots?

- Yes!
  - The PREEMPT_RT patch provides sufficient determinism that most robotics applications work fine
- You can also use Xenomai or RTAI



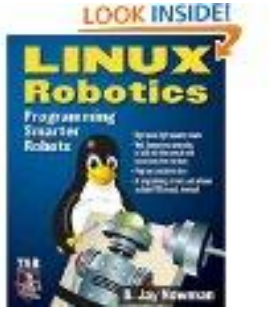


Which Languages can I use?

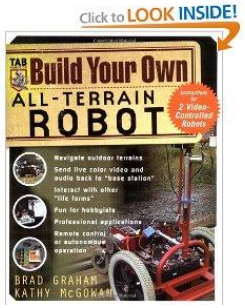
- There are many languages that you can use
- The usual suspects:
  - C/C++
  - Java
  - Python
  - Bash shell scripts
- An example of C/C++ can be found here:
  - http://www.adamsinfo.com/using-the-phidget-interface-kit-under-linux/
- So far, there doesn't appear to be a particular language that focuses on robotics any better than any other
- There are bindings for these and more available in the user forums and from suppliers of the hardware interfaces

Good Robotics Books

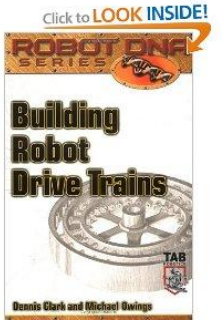
- There are a couple of good books for building robots
  - Linux Robotics
    D. Jay Newman
    ISBN: 9780071444842



  - Build Your Own All-Terrain Robot
    Brad Graham
    Kathy McGowan
    ISBN: 007143741X



  - Robot Drive Trains
    Michael Owings
    ISBN: 9780071408509



Summary

- Building a robot can be a very fun, frustrating and rewarding effort
  - Just don't tell you S/O how much money you've spent!

- Be sure to use your web resources to keep from reinventing the wheel, so to speak
- If you really like robots, consider becoming a mentor to an FRC Robotics Team
  - http://www.usfirst.org