6.270 Toolbox: Overview

Lego components
- Bracing and Structure
- Steering configurations
- Gearing
- Best practices

Electrical components
- Actuators
- Sensors
- Best practices

Software
- HappyBoard and JoyOS
- Happyboard Programming Software
- Subversion and the JoyOS sources
- JoyOS API
Lego Components: Bracing

- Use as many cross braces as possible
- Robot chassis should survive a 3 foot drop
Most robots use differential steering and a caster wheel.
Lego Components: Gearing

- Doubly support shafts
- Minimize friction- allow play between gear stages
- For driving, start with a ratio between 50:1 and 125:1
Electrical Components: Motors

- High speed, low torque
- Up to four motors
- Can drive forward or backward at variable speeds (see HappyTest)
- Can hard brake or soft brake (glide), see JoyOS API
- Put an 8-tooth LEGO gear over the axle
- Brace generously, see course notes for sample motor housing
- Use 3-pin header
Electrical Components: Servos

- Position control
- Have ~180 degree range
- Screw a 24 or 40 tooth LEGO gear to the rotating portion
- High internal gear ratio, low speed but high torque
- When attaching to HappyBoard, white wire is signal
- Can unlock the servo, becomes a high torque motor (need to take servo apart for this, see us!)
Electrical Components: Switches

- Digital
- Detect contact with walls, obstacles, or other robots
- Regulate internally moving parts
Electrical Components: Breakbeams

- Count wheel revolutions
- Place high in geartrain, use with lego pulley
- Need to use 330ohm resistor
- Useful for driving straight
Electrical Components: Distance Sensors

- Requires one time calibration (see sample code)
- Determines distance to nearest obstacle
- Valid between around 6" to 3'
- Requires cutting trace on HappyBoard
Happy Board and Joyos

The **Happyboard** is a robot controller board specifically created for 6.270.

FPGA control of motors, sensors, servos.
More specs in the manual.

**JoyOS** Provides:

- Hardware Abstraction (sensors, motors, RF)
- Multithreading
- Software Library (PID control, etc)

Program by holding "Stop" while turning on the board, and see the following slide.
Happyboard Programming Software

Detailed instructions on wiki (see refs)

**Windows:**
Install the lastest **WinAVR** package with all the defaults
Find the serial port **COM number** in Device Manager (Computer->Properties->Hardware->Device Manager->Ports (COM & LPT) with the Happy board attached.
Change the **Makefile** to use COM<X> and "make program" your board with WinAVR.

**Linux (Ubuntu shown):**
```
# apt-get install binutils-avr gcc-avr avr-libc avrdude
(Use SVN to get the current release of JoyOS)
$ cd <path_to_joyOS>/src/robot
Change Makefile to use /dev/ttyUSB* (use ls /dev/ttyUSB*)
$ make
$ make program
```

**Mac OS X:**
Same as Ubuntu, except with Fink/MacPorts and /dev/tty.usb*
Subversion

Linux: sudo apt-get install subversion
OS X: Mac version of SVN <http://bit.ly/2QH3wo>

- `svn up` while in the `joyos` directory.
- `joyos->{doc, inc, lib, src}`

Optional:
- Setup your own SVN repository.
- Use `svn externals` to pull in the `joyos` release
Serial Terminal

Currently, we expect you to debug your code using the serial terminal. It may be possible that we allow you to debug over RF. The baud rate we use is **19200**.

- Linux: cutecom (sudo apt-get install cutecom)

With your robot connected to your computer via USB:

To use the serial terminal for debugging:

- use `printf("Some sort of formatted String", variables)` to output from the robot;
- and use `scanf()` to allow the robot to get input from the serial terminal.

You could use this to create a menu system or terminal for debugging your robot.
JoyOS API

Reading Sensors:
uint16_t analog_read(uint8_t port)
uint8_t digital_read(uint8_t port)

Servos, Motors:
void servo_set_pos(uint8_t servo, uint16_t pos)
  //0-511
void motor_set_vel(uint8_t motor, int16_t vel)
  //vel of 0 sets to coast

RF Broadcasted Information:
float get_self_position_x()
float get_self_position_y()
float get_other_position_x()
float get_other_position_y()
float get_mouse_position_x()
float get_mouse_position_y()
References

Refer to the contestants' page!
https://spacecats.mit.edu:444/~6.270/contestants/

Contains **2010 Course notes**, calendar, JoyOS API