18-642 Recitation #1

January 19, 2018
Updates

- Homework #2 due @ 11pm Sunday 1/21
- Project 1 released, due tonight 1/19 @ 11pm
  - Introduction to ROS
- Project 2 due 1/26 @ 11pm
- Fill out question sheet to receive attendance points today!
Today

• Homework #2 sample
• Homework #3 overview
• Project 1/ROS Demo
• Project 2 Preview
Homework #2 Sample
Homework #3 Overview

• Based on Toyota lecture
• Explore Audi 5000 case
• Give some opinions
  – No right answers – want to see you thinking through some scenarios
• Be ready to talk through questions in recitation next week!
Homework questions?
Project 1

• Work through ROS tutorials
• Write your own simple ROS code
ROS Demo

- Robot operating system
- Lots of useful tutorials ([http://wiki.ros.org/ROS/Tutorials](http://wiki.ros.org/ROS/Tutorials))
  - You’ll do some of them for the project
ROS code preview

```cpp
#include "ros/ros.h"
#include "geometry_msgs/Twist.h" // access using geometry_msgs::Twist
#include "turtlesim/Pose.h" // access using turtlesim::Pose

// any state variables or callback/computation functions go here

int main(int argc, char **argv) {
    ros::init(argc, argv, "turtle_publisher_18642");
    ros::NodeHandle nh;

    ros::Publisher vel_pub = nh.advertise<geometry_msgs::Twist>("/turtle1/cmd_vel", 1000);
    ROS_INFO("Starting up turtle_publisher_18642.");

    /*
     * The following is code to command the turtle in a straight line, in 5 steps.
     * You should replace it with your own code to draw a figure 8.
     */
    ros::Rate r(.5); // .5Hz (run every 2 seconds)

    int num_steps = 5;

    while (num_steps > 0) {
        geometry_msgs::Twist vel_msg;
        vel_msg.linear.x = 1.0;
        vel_pub.publish(vel_msg);
        num_steps--;
        ros::spinOnce();
        r.sleep();
    }
    return 0;
}
```
Project 1 Questions?
Project #2

• Focus on working with messy existing code
• Full points even if you don’t succeed implementing the solution
• Graded on documentation of process
Maze follower

• Current implementation is left-hand rule:
  – Keep left hand constantly on wall
  – Can think of robot constantly trying to turn left

Source: https://en.wikipedia.org/wiki/Maze_solving_algorithm
Maze follower

• Your task is to switch to right-hand rule
  – Read the existing code and modify it

• You get full points for a well-documented effort

Source: https://en.wikipedia.org/wiki/Maze_solving_algorithm
Project #2 Structure

1. Download, compile, and run code
2. Attempt to implement solution
3. Clean up code
4. Attempt to implement again
Demo
Clean up code

- On a working version of the code
- Spend 2-5 hours making style changes
- Do not consult with other students about what good style is
- Keep a log in intervals to nearest 0.1 hr:
  - 0.2 hours: Found and used a tool ([tool name]) that automatically applies a coding style.
  - 0.3 hours: converted turtle direction to enum.
  - 0.1 hours: ...
Implementation of solution

• **Only modify the student.cpp file!**
• Please obey the time limits!
• You are only graded on your explanation of why you did/did not succeed
• If you succeeded on the first try:
  – Still clean up code (as it stands)
  – Automatically get full points for second try
Version control

• Start versioning as soon as you can compile and run the files we gave you
• Google how to turn existing directory into a (git/svn/mercurial/etc) repository
  – You can host a repository on your afs space
• Use version control to revert to working version before code cleanup if your implementation doesn’t work
Project #2 Summary

• Focus on process
• Writeup is the most important part of the grade
  – Turn in student.cpp to show evidence of work
• You do not have to succeed in your implementation!
Questions?

• Homeworks
• Projects 1,2
• Course structure