18-642 Recitation #1

Aug 31, 2018
Updates

- Homeworks #1, #2, #3 due Wed. Sept 5
  - Note that HW #1 is a Canvas quiz
  - Wed evening → 8 AM Thursday morning

- Project 1 released, due tonight!
  - Introduction to ROS
  - If you get stuck:
    1. Send e-mail to course staff before deadline
    2. Get it resolved in office hours next week

- Project 2 due in one week
Today

• Project 1/ROS Demo
• Project 2 Preview
• Homeworks
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)
  – 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.\n");

    /*
    * 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 2-6 hours making style changes
• **Do not consult with other students about what good style is**
  – We’re taking a baseline of student experience
• 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
  – **Do NOT** make repository public

• 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!
Hints for Success

• Read the *project grading rubric*
  – Use it as a hand-in checklist
  – Read grading notes

• Check your work
  – Un-zip your hand-in to check it’s all there

• Don’t wait until the last minute!
  – Make a weekly plan; start early
  – Don’t expect TAs to monitor e-mail 24x7

• Come to office hours
  – Often the most important points aren’t the ones you came to ask about
Homework Overview

• Hand in slide format
  – Powerpoint (preferred) or Acrobat
• Include:
  – Your name and Andrew ID
  – Question number
  – Brief rephrase of question for context
  – Your answer
• Be ready to talk about your homework slides in lecture and/or recitation
Homework Format

• Don’t use a title slide
• Make it legible:
  – Fonts 16 points or larger (14 absolute minimum)
  – Put references to material, even clip-art
    • URL shortener is OK
  – Avoid the fancy template; use all the room on the slide to make things bigger
  – Use high contrast
    • No dark gray text on light gray background!
• Similar rules for in-class exercises
Homework questions?
Questions?

• Projects 1,2
• Homeworks
• Course structure