**PA1** - EECE 5554 Robotics Sensing and Navigation

**Deadline**: 11:59 PM on Sunday, Jan 19, 2020

**Readings**

### 1. Real-Time Operating Systems

### [**https://en.wikipedia.org/wiki/Real-time\_operating\_system**](https://en.wikipedia.org/wiki/Real-time_operating_system)

### Big question to think about – why we might need RTOS and why me might not?

### If you look at the issues they address, keep thinking over the next few weeks about whether they are addressed in ROS and if so to what extent?

### 2. Read the following article.

### [**http://www.naturalnavigator.com/find-your-way-using/stars**](http://www.naturalnavigator.com/find-your-way-using/stars)

### Before we get into GPS satellites I want you to focus for a millisecond on celestial navigation. We all know our latitude in Boston (or it is trivial to look up). Estimate your latitude using the methodology in this 1 page article and compare it with the real result. Be ready to give a truthful answer in class.

### Latitude is relatively easy. What about Longitude? A good reference is the book by Dava Sobel called simply “Longitude”.

**NOTE**: This and all our ROS homeworks are based on the format used by Prof Louis Whitcomb for the class he teaches at Johns Hopkins University.

3.ROS

* [ROS Introduction](http://wiki.ros.org/ROS/Introduction)
* [ROS Get Started Guide](http://wiki.ros.org/ROS/StartGuide) Introduction, Concepts, Higher-Level Concepts,Client Libraries, Technical Overview

**Useful Resources**

* [ROS developer's guide](http://wiki.ros.org/DevelopersGuide)
* [ROS C++ Style Guide](http://wiki.ros.org/CppStyleGuide)
* [ROS Python Style Guide](http://wiki.ros.org/PyStyleGuide)
* Useful cheat-sheets:
  + [ROS cheat-sheet for Indigo and Catkin](http://www.ros.org/news/2015/05/ros-cheatsheet-updated-for-indigo-igloo.html)
* Documentation on command line tools
  + [rosnode](http://wiki.ros.org/rosservice)
  + [rostopic](http://wiki.ros.org/rostopic)
  + [rosparam](http://wiki.ros.org/rosparam)
  + [rosmsg](http://wiki.ros.org/rosmsg)

**Install ROS**

**Variant : Desktop-Full Install: (Recommended)**

Install ROS Melodic (In case of ubuntu 18.04) or ROS Kinetic (in case of ubuntu 16.04)

ROS Melodic <https://wiki.ros.org/melodic/Installation/Ubuntu>

ROS Kinetic <https://wiki.ros.org/kinetic/Installation/Ubuntu>

**Tutorials**

**ROS** Tutorials on official ROS wiki <https://wiki.ros.org/ROS/Tutorials>

**TF** tutorials on official ROS wiki <https://wiki.ros.org/tf/Tutorials>

Complete the following tutorials from above ROS tutorials link

1. [Navigating the ROS Filesystem](https://wiki.ros.org/ROS/Tutorials/NavigatingTheFilesystem)This tutorial introduces ROS filesystem concepts, and covers using the roscd, rosls, and [rospack](https://wiki.ros.org/rospack) commandline tools.
2. [Creating a ROS Package](https://wiki.ros.org/ROS/Tutorials/CreatingPackage)This tutorial covers using [roscreate-pkg](https://wiki.ros.org/roscreate) or [catkin](https://wiki.ros.org/catkin) to create a new package, and [rospack](https://wiki.ros.org/rospack) to list package dependencies.
3. [Building a ROS Package](https://wiki.ros.org/ROS/Tutorials/BuildingPackages)This tutorial covers the toolchain to build a package.
4. [Understanding ROS Nodes](https://wiki.ros.org/ROS/Tutorials/UnderstandingNodes)This tutorial introduces ROS graph concepts and discusses the use of [roscore](https://wiki.ros.org/roscore), [rosnode](https://wiki.ros.org/rosnode), and [rosrun](https://wiki.ros.org/rosrun) commandline tools.
5. [Understanding ROS Topics](https://wiki.ros.org/ROS/Tutorials/UnderstandingTopics)This tutorial introduces ROS topics as well as using the [rostopic](https://wiki.ros.org/rostopic) and [rqt\_plot](https://wiki.ros.org/rqt_plot) commandline tools.

**Readings for ros parameters**

1. [Documentation on parameter command-line tool (rosparam) and python client libraries](http://wiki.ros.org/Parameter%20Server)
2. [ROS Parameters](http://wiki.ros.org/rospy_tutorials/Tutorials/Parameters)
3. [Understanding ROS Services and Parameters](https://wiki.ros.org/ROS/Tutorials/UnderstandingServicesParams)This tutorial introduces ROS services, and parameters as well as using the [rosservice](https://wiki.ros.org/rosservice) and [rosparam](https://wiki.ros.org/rosparam) commandline tools.
4. [Using rqt\_console and roslaunch](https://wiki.ros.org/ROS/Tutorials/UsingRqtconsoleRoslaunch)This tutorial introduces ROS using [rqt\_console](https://wiki.ros.org/rqt_console) and [rqt\_logger\_level](https://wiki.ros.org/rqt_logger_level) for debugging and [roslaunch](https://wiki.ros.org/roslaunch) for starting many nodes at once. If you use ROS fuerte or ealier distros where [rqt](https://wiki.ros.org/rqt) isn't fully available, please see this page with [this page](https://wiki.ros.org/ROS/Tutorials/UsingRxconsoleRoslaunch) that uses old rx based tools.
5. [Using rosed to edit files in ROS](https://wiki.ros.org/ROS/Tutorials/UsingRosEd)This tutorial shows how to use [rosed](https://wiki.ros.org/rosbash) to make editing easier.
6. [Creating a ROS msg and srv](https://wiki.ros.org/ROS/Tutorials/CreatingMsgAndSrv)This tutorial covers how to create and build msg and srv files as well as the [rosmsg](https://wiki.ros.org/rosmsg), rossrv and roscp commandline tools.
7. [Writing a Simple Publisher and Subscriber (Python)](https://wiki.ros.org/ROS/Tutorials/WritingPublisherSubscriber%28python%29)This tutorial covers how to write a publisher and subscriber node in python.
8. [Examining the Simple Publisher and Subscriber](https://wiki.ros.org/ROS/Tutorials/ExaminingPublisherSubscriber)This tutorial examines running the simple publisher and subscriber.
9. [Writing a Simple Service and Client (Python)](https://wiki.ros.org/ROS/Tutorials/WritingServiceClient%28python%29)This tutorial covers how to write a service and client node in python.
10. [Examining the Simple Service and Client](https://wiki.ros.org/ROS/Tutorials/ExaminingServiceClient)This tutorial examines running the simple service and client.

**Readings for rosbag**

[Documentation on rosbag](http://wiki.ros.org/rosbag/Commandline)

1. [Recording and playing back data](https://wiki.ros.org/ROS/Tutorials/Recording%20and%20playing%20back%20data)This tutorial will teach you how to record data from a running ROS system into a .bag file, and then to play back the data to produce similar behavior in a running system.
2. [Getting started with roswtf](https://wiki.ros.org/ROS/Tutorials/Getting%20started%20with%20roswtf)Basic introduction to the [roswtf](https://wiki.ros.org/roswtf) tool.

**TF tutorials**

**Readings for TF**

[Documentation on tf command-line tools and python client libraries](http://wiki.ros.org/tf)

1. [Writing a tf broadcaster (Python)](https://wiki.ros.org/tf/Tutorials/Writing%20a%20tf%20broadcaster%20%28Python%29)This tutorial teaches you how to broadcast the state of a robot to tf.
2. [Writing a tf listener (Python)](https://wiki.ros.org/tf/Tutorials/Writing%20a%20tf%20listener%20%28Python%29)This tutorial teaches you how to use tf to get access to frame transformations.
3. [Adding a frame (Python)](https://wiki.ros.org/tf/Tutorials/Adding%20a%20frame%20%28Python%29)This tutorial teaches you how to add an extra fixed frame to tf.
4. [Learning about tf and time (Python)](https://wiki.ros.org/tf/Tutorials/tf%20and%20Time%20%28Python%29)This tutorial teaches you to use the waitForTransform function to wait for a transform to be available on the tf tree.
5. [Time travel with tf (Python)](https://wiki.ros.org/tf/Tutorials/Time%20travel%20with%20tf%20%28Python%29)This tutorial teaches you about advanced time travel features of tf

**What to Submit for PA1 ?**

Write and test the following:

* + Write and test a publisher node that publishes a TOPIC and subscriber node that subscribes to that TOPIC.
  + Write and test a server node that provides a SERVICE and client node that calls this SERVICE
  + ROS Parameters
  + A simple python ROS node that reads some parameters on startup and uses them
  + A roslaunch file that sets the parameters and runs the python node
  + Package containing your tutorial code and launch files for
* tf broadcaster node
* tf listener node
* adding a moving tf frame
* adding a tf time-travel frame

**How to Submit PA1 ?**

1. In your class repo ‘EECE5554\_RoboticsSensing’, create a directory called PA1
2. Inside PA1, create sub-directory structure ‘catkin\_ws/src’.
3. Copy your catkin\_package used for assignment under src.

Your repo structure should look similar to ‘<Path\_to\_repo>/EECE5554\_RoboticsSensing/PA1/catkin\_ws/src/<your\_catkin\_pkg\_dir>’

1. Push your local commits to (remote) gitlab server. You can verify this by visiting gitlab.com and making sure you can see the commit there.
2. Upload a .ppt with your gitlab link and the screen shots of the output of your working code to Blackboard