NAZARBAYEV UNIVERSITY
SCHOOL OF SCIENCE AND TECHNOLOGY

Project Assignment #4

CSCI502 – HARDWARE/SOFTWARE CO-DESIGN

Spring Semester 2019

INTRODUCTION TO PTHREAD SYNCHRONIZATION AND ROBOT OPERATING SYSTEM (ROS)

DUE TIME AND DATE

- **Class presentation and report uploaded to Moodle on Wednesday 4 April**

LEVEL OF COLLABORATION ALLOWED

You can work in assigned groups of students

REFERENCES

Effective Robotics Programming with ROS. Third edition, 2017 (available in Moodle)

This assignment is adopted from https://tams.informatik.uni-hamburg.de/lectures/2017ws/praktikum/mobileRob/assignments/Assignment_01.pdf

INTRODUCTION

In this assignment you will learn basics of Robot Operating System (ROS) that is a trending robot application development platform that provides various features such as message passing, distributed computing, code reusing, and so on. Please refer to the software website www.ros.org , course textbook, introductory lecture slides and other information from Internet for more information about the system.

You will work on ROS Kinetic Kame version running on Ubuntu 16.04 OS and will follow instructions from the Effective Robotics Programming with ROS textbook (available in Moodle) and ROS wiki tutorials http://wiki.ros.org/

ASSIGNMENT DETAILS

TASK 1. POSIX THREADS PROGRAMMING AND SYNCHRONIZATION (30% OF THE TOTAL GRADE)

1. Implement the programming project below and demonstrate in class. Write a report with a detailed description of the project code and its operation including screenshots in the report. You can refer to class exercises and the POSIX threads programming tutorial with exercises https://computing.llnl.gov/tutorials/pthreads

@Almas Shintemirov email: ashintemirov@nu.edu.kz

The Sleeping TA programming project

A university Computer Science department has a teaching assistant (TA) who helps students with their programming assignments during regular office hours. The TA's office is rather small and has room for only one desk with a chair and computer. There are three chairs in the hallway outside the office where students can sit and wait if the TA is currently helping another student. When there are no students who need help during office hours, the TA sits at the desk and takes a nap. If a student arrives during office hours and finds the TA sleeping, the student must awaken the TA to ask for help. If a student arrives and finds the TA currently helping another student, the student sits on one of the chairs in the hallway and waits. If no chairs are available, the student will come back at a later time.

Using POSIX threads, mutex locks, and semaphores, implement a solution that coordinates the activities of the TA and the students. Details for this assignment are provided below.

Using Pthreads, begin by creating n students. Each will run as a separate thread. The TA will run as a separate thread as well. Student threads will alternate between programming for a period of time and seeking help from the TA. If the TA is available, they will obtain help. Otherwise, they will either sit in a chair in the hallway or, if no chairs are available, will resume programming and will seek help at a later time. If a student arrives and notices that the TA is sleeping, the student must notify the TA using a semaphore. When the TA finishes helping a student, the TA must check to see if there are students waiting for help in the hallway. If so, the TA must help each of these students in turn. If no students are present, the TA may return to napping.

Perhaps the best option for simulating students programming—as well as the TA providing help to a student—is to have the appropriate threads sleep for a random period of time.

TASK 2: ROS TUTORIAL PRACTICE (10%)

2. Install desktop-full ROS Kinetic version to your PC with Ubuntu 16.04 following instructions from http://wiki.ros.org/kinetic/Installation/Ubuntu

   You can use an Ubuntu virtual machine for this assignment - follow instructions from Chapter 1 of the textbook.

3. Practice the Core ROS Tutorials (Beginner Level) from http://wiki.ros.org/ROS/Tutorials and refer to Chapter 2 of the textbook for more information. In particular, practice Turtlesim tutorials for teleoperating a virtual turtle in order to learn how to start ROS and ROS nodes and get to know basic ROS commands.

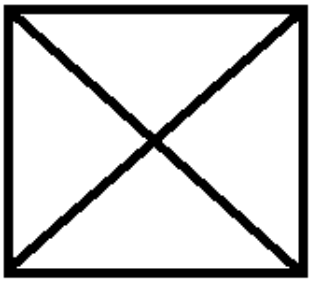
   Use several command terminals for running roscore, rosrun turtlesim turtlesim_node, etc, rosnode list, rostopic list and other commands.

4. Please document your work in the report and explain the following ROS elements with regards to your ROS tutorial activities (they will be also asked in the project demo sessions):
    - Nodes;
    - Communication: messages, topics, services;
    - ROScore.

TASK 3: ROS NODE AND PACKAGE CREATION PRACTICE (25%)

Create a ROS package with your group's name in your workspace. For this task, look up this tutorial: http://wiki.ros.org/ROS/Tutorials/CreatingPackage

Your program will depend on the 'turtlesim' and the 'geometry msgs' packages.



1. Using the turtlesim tutorial create your first ROS node: Write a node that moves the turtle around, such that it draws a square with diagonals as shown in the figure on a side or other figure of similar or more complexity.
2. Look up the ROS tutorial for a simple publisher. There is one for C/C++ and one for Python. You have to make sure you look at the tutorial for ROS Kinetic and language of your choice.
3. Now **write a new node** (either in C++ or Python) that publishes messages that make the turtle draw the figure. In this tutorial you can look up which topic the turtle uses: http://wiki.ros.org/ROS/Tutorials/UnderstandingTopics
4. Every package for ROS has to be build at least once using **catkin_make** in order to make it known to the workspace. This has to be done even if you implement the node within the package in Python. Afterwards you can start your node with the **rosrun** command (just as you started the nodes in the previous task): **rosrun <package> <program>**
5. Read up on the Services available by Turtlesim. Try calling some of turtlesim's Services through the command line.

TASK 4: ROS APPLICATION REVIEW AND CLASS DEMO (25%)

1. Explore the ROS textbook, introductory video at http://www.ros.org/about-ros/ about ROS capabilities.
2. Use of the provided in Moodle example papers to get inspired and search IEEEXplore research database (http://ieeexplore.ieee.org/) (full paper access available from computers/laptops connected to Internet on NU campus except Student Hall blocks) and find most relevant paper describing ROS application to various robotics applications such as robot vision and perception, mobile robotics, human-robot interaction, intelligent robotics, etc.

3. Make a short presentation describing the ROS implementation (nodes, topics, services, etc.) based on the paper of your choice for class demo and discussion with other students.

GRADING CRITERIA

Demonstration and grading of your presentation including running Turtlebot simulation will be done during the class lab session on Wednesday 3 April. The total grade for all implemented assignments as described will constitute 90%. Bonus grade will be awarded to 2 groups showing best presentations explaining ROS application with implementation details, video and demonstrating clear understanding of the material.

Please submit the presentation with your reference paper, detailed project report and program code (zip-file of your package in your workspace source folder, e.g. catkin_ws/src folder) to the project folder in Moodle by the end of Thursday 15 February.

This project evaluation will be done using individual grading depending on the level of participation and understanding of the project assignments.

Late submission penalty – 10% per day