NAO Lab Assignment #2

Assignment to understand API (noted below) due before you come to lab.
Programming occurs in-class (2:10PM), either Tuesday, Oct. 18, or Thursday, Oct. 20, and either Tues., Oct. 25, or Thurs., Oct. 27, according to your group assignment below

This in-class exercise is for you to program the NAO robot to walk around randomly (and carefully), use sonar and foot bump sensors to detect obstacles, and change walking direction when obstacles are detected. Note that this is a graded exercise. Each team will receive a grade on how well the robot accomplishes its task. Individual student grades will be a function of your team's grade, peer-review from your teammates, and possibly the TA's observation of your individual contributions to your team.

NAO API:

- The NAO robots run a service called NaoQi. Just as choreographe can communicate with that service, you can also connect with the naoqi python library.
- Before the in-class lab, familiarize yourself with the NaoQi API here: http://users.aldebaran-robotics.com/docs/site_en/bluedoc/naoqi.html. You will need to reference it when working on this lab.
- If you are not familiar with Python, you should also try to learn as much as you can about it before the labs, so that you can better contribute to your team. It isn't hard to learn.
- Some notes about the NaoQi API:
    - Note that there are modules that represent higher-level features, like ALFaceTracker, ALSonar, etc.
    - Also note that much information about the state of the NAO can be accessed via the ALMemory getData function (http://users.aldebaran-robotics.com/docs/site_en/bluedoc/ALMemory.html#getData). This includes the state of any of the simple sensors (e.g., bump sensors on the robot's feet). Available data "names" can be found with the getDataListName() function. You can also subscribe to sensors via the ALSensors module.
    - In choreographe, you can save any program as a "behavior" locally on the robot. Then, using the ALBehaviorManager module (http://users.aldebaran-robotics.com/docs/site_en/bluedoc/ALBehaviorManager.html), you can call any behavior that you have saved on the robot. This allows us to package difficult or common behaviors. "StandUp" and "SitDown" are the names of the Stand Up and Sit Down behaviors that have been loaded onto your NAOs for you.

TASK:

- Your task for this lab is to create a simple program that makes the NAO walk around carefully, uses sonar and foot bump sensors to detect obstacles, and changes direction when obstacles are detected.
- Your program will run on your NAO's laptop and connect to your NAO. (Python and the naoqi library are already installed on the lab's laptops).
- This behavior should continue until a sensory input is given, such as a contact depress (like on the head) or a voice command.
- Sample python code has been provided below that illustrates how to use some of the more important API pieces.
- As always, don't let your robot fall!!

Sample Python Code for key parts of the API:

```
#!/usr/bin/python
from naoqi import *
IP = '10.26.210.XX' #set your robot's IP here
PORT = 9559

speech = ALProxy("ALTextToSpeech", IP, PORT)
mp = ALProxy("ALMotion", IP, PORT)
b = ALProxy("ALBehaviorManager", IP, PORT)
mp.setStiffnesses('Body', 1)
sonar = ALProxy("ALSonar", IP, PORT)
sonar.subscribe("SonarTest", 500, 1.0)
mem=ALProxy('ALMemory', IP, PORT)

b.stopAllBehaviors()
speech.post.say("Standing Up")
b.runBehavior('standUp')

mp.setWalkArmsEnable(True,True)
sonarLeftDetected = mem.getData('SonarLeftDetected')
sonarRightDetected = mem.getData('SonarRightDetected')
headMiddle =
mem.getData('Device/SubDeviceList/Head/Touch/Middle/Sensor/Value', 0)
leftBumper  = mem.getData('LeftBumperPressed')
rightBumper = mem.getData('RightBumperPressed')
mp.setWalkTargetVelocity(-1,0,0,MAXSPEED) #back up
mp.setWalkTargetVelocity(1,0,0,MAXSPEED) #move forward
mp.setWalkTargetVelocity(0,0,1,MAXSPEED) #move left
mp.setWalkTargetVelocity(0,0,-1,MAXSPEED) #move right
mp.setWalkTargetVelocity(0,0,0,MAXSPEED) #stop
speech.post.say("Sitting Down")
b.runBehavior('sitDown')
mp.setStiffnesses('Body',0)
```

Here's the plan: Per the assignments below:

Tuesday, Oct. 18 and Tuesday, Oct. 25:

- Groups 1-4 meet in Claxton 105 (Hydra Lab) for in-class NAO exercise
- Groups 5-8 meet in Claxton 205 (regular classroom) for lecture on obstacle avoidance

Thursday, Oct. 20 and Thursday, Oct. 27:

- Groups 1-4 meet in Claxton 205 (regular classroom) for lecture on obstacle avoidance
- Groups 5-8 meet in Claxton 105 (Hydra Lab) for in-class NAO exercise

You are assigned to the following groups for in-class lab on either Tuesday, Oct. 18 or Thursday, Oct. 20. The name beside the group number is the name of the NAO robot assigned to your group. Each team has at least 2 members who said they are comfortable programming in Python. Undergrads and grads are in separate groups, to facilitate grading on different curves.

In-Class Lab (in Claxton 105 – Hydra Lab) on Tuesday, Oct. 18 and Tuesday, Oct. 25:

Group 1: Bumblebee Chad Armstrong Cody Boyles Trushna Patel Doug Slater	***Group 2: Ironhide*** John Blood Matthew Burnett Aaron Mishtal Jungi Jeong	
Group 3: Jazz Colin Campbell Chris Maddux Alex Saites	***Group 4: Ratchet*** Mike Franklin Nick Lineback Ryhan Pathan Lonnie Yu	***Optimus*: Backup robot**

In-Class Lab (in Claxton 105 – Hydra Lab) on Thursday, Oct. 20, and Thurs. Oct. 27:

Group 5: Bumblebee Tyler Burkle Ian Harmon Curtis Taylor	***Group 6: Ironhide*** Phil Andreason Michael Craze Eric Martin Peter Pham	
Group 7: Jazz Peter Gaultney Dan Henderson Bob Lowe Joshua Strange	***Group 8: Ratchet*** Blake Haugen Zahra Mahoor Jacob Robertson Chi Zhang	***Optimus*: Backup robot**

If it isn't your turn in the lab, remember that you should be in the regular classroom for a lecture on obstacle avoidance.

Grading:

During the 2nd week of the lab, you must demonstrate your robot's performance to the TA, who will grade your exercise. This grade will be based on the ability of the robot to move around safely, to detect obstacles, and to turn away from the obstacles. The more area that your robot can explore safely, the better your robot's performance will be graded. Points will be deducted if your robot falls (even though it is caught by the safety harness).

You will also conduct peer reviews of your teammates, to help ensure that everyone contributes to the assignment. Note that not everyone can contribute in the same way, since there will typically be just one programmer at a time. But, all can work toward making the assignment a success.

To allow for this time to grade during the 2nd week, your team's programming must be complete by early in the 2nd lab. Thus, you should work efficiently during the first lab to try to complete as much of the assignment as possible. The TA will begin grading in order of when the teams complete the assignment. If certain teams are not complete, but time is running out, the TA will grade the assignment based on the robot performance achieved by the time of the grading during the 2nd lab. If this situation occurs, graduate student groups will be graded before undergraduate student groups (thus giving the undergraduate groups more time, if needed).

If your team is very productive, and completes the lab quickly, then you should use the remainder of the lab time to explore other functionalities of the robot, and to (safely) program the NAO to achieve new capabilities.