Homework 4: Gazebo and ROS

24-775 Robot Design & Experimentation

Name: ________________________________

This homework will introduce Gazebo, a robotics simulation package, and the Robot Operating System (ROS). These instructions involve downloading a virtual machine that contains these packages already installed, although if you have a Linux machine already you may install and use Gazebo and ROS directly. The versions used in the virtual machine are: Ubuntu 14.04, Gazebo 7, and ROS Indigo, though other versions may work as well.

You will also be communicating using Matlab and the Matlab Robotics System Toolkit. These are available through CMU: https://www.cmu.edu/computing/software/all/matlab/index.html

1. To begin, install VirtualBox: https://www.virtualbox.org/

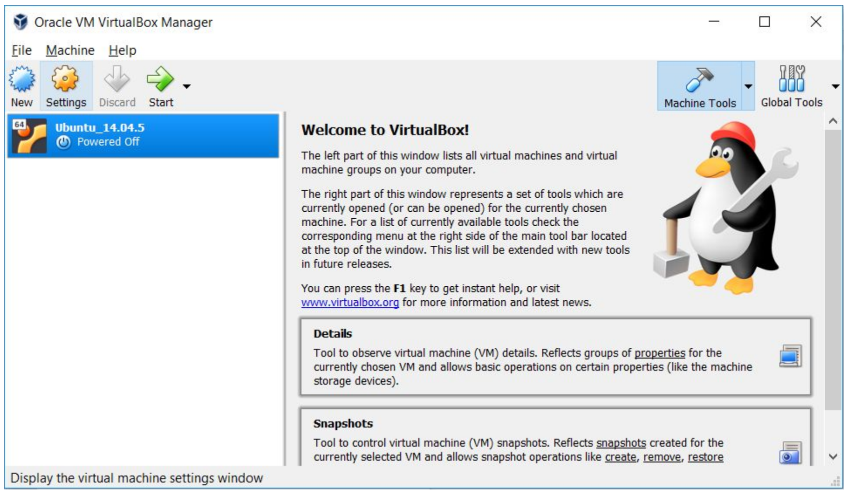
Then, download the virtual machine file:

https://cmu.box.com/s/3b7i1aymjy4jm0zt6vikn8b6a7p19s1b

This is a large file (3.1GB), and will require additional space (7.98GB) on your computer once it is imported. You may delete the downloaded file once it is imported to regain that space.

The username and password on the virtual machine is: `robot`

2. Next, import the virtual machine. (File > Import Appliance > next). Once it is imported, you should see a screen that looks like this:



Ubuntu_14.04.5
Powered Off

Welcome to VirtualBox!

The left part of this window lists all virtual machines and virtual machine groups on your computer.

The right part of this window represents a set of tools which are currently opened (or can be opened) for the currently chosen machine. For a list of currently available tools check the corresponding menu at the right side of the main tool bar located at the top of the window. This list will be extended with new tools in future releases.

You can press the **F1** key to get instant help, or visit www.virtualbox.org for more information and latest news.

Details
Tool to observe virtual machine (VM) details. Reflects groups of properties for the currently chosen VM and allows basic operations on certain properties (like the machine storage devices).

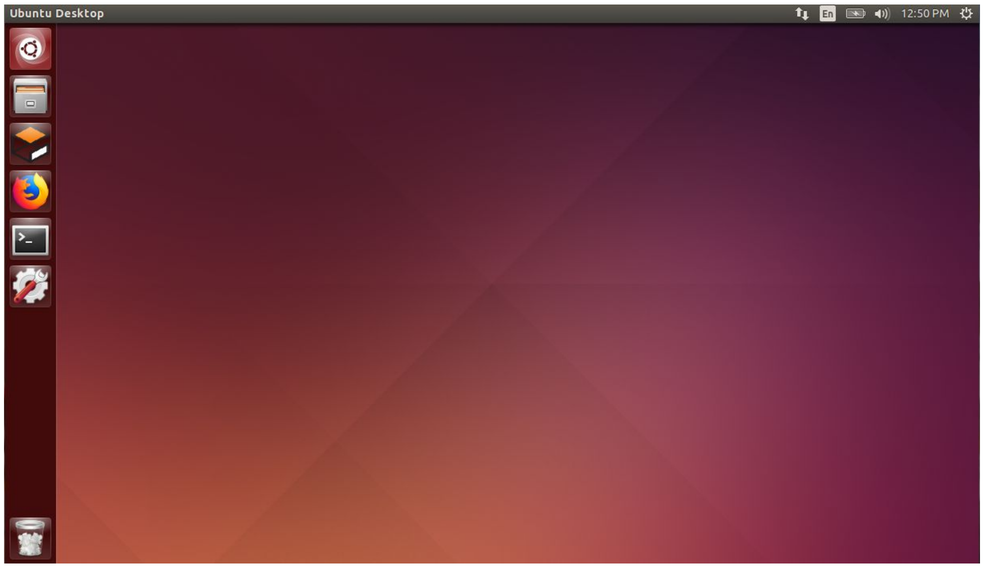
Snapshots
Tool to control virtual machine (VM) snapshots. Reflects snapshots created for the currently selected VM and allows snapshot operations like create, remove, restore

Display the virtual machine settings window

Allocate appropriate (more if you want faster VM) processors/memory to the VM.

(Settings>System> change Base Memory and Processor). When you click “Start” you should eventually see a desktop that looks like this:

Run programs→
Open Files →
Gazebo →
Firefox →
Terminal →
Settings →



Everything you should need is in that left hand launcher.

TIP: If you are having trouble getting the VirtualMachine to load with an error message like "No host-only network adapter is currently selected" or "...the following physical network interfaces were not found" you may need to setup your VirtualBox network. From the main VirtualBox Manager, click on "Global Tools"->"Host Network Manager" and make sure there is a Host-Only Ethernet Adapter added in this list. Then go back to your machine settings and connect the first network adapter to that Host-Only Ethernet Adapter.

3. Begin by following the beginner Gazebo tutorials. You can run gazebo by clicking the Gazebo icon on the left.

“Understanding the GUI” : http://gazebosim.org/tutorials?cat=guided_b&tut=guided_b2

“Model Editor”: http://gazebosim.org/tutorials?cat=guided_b&tut=guided_b3

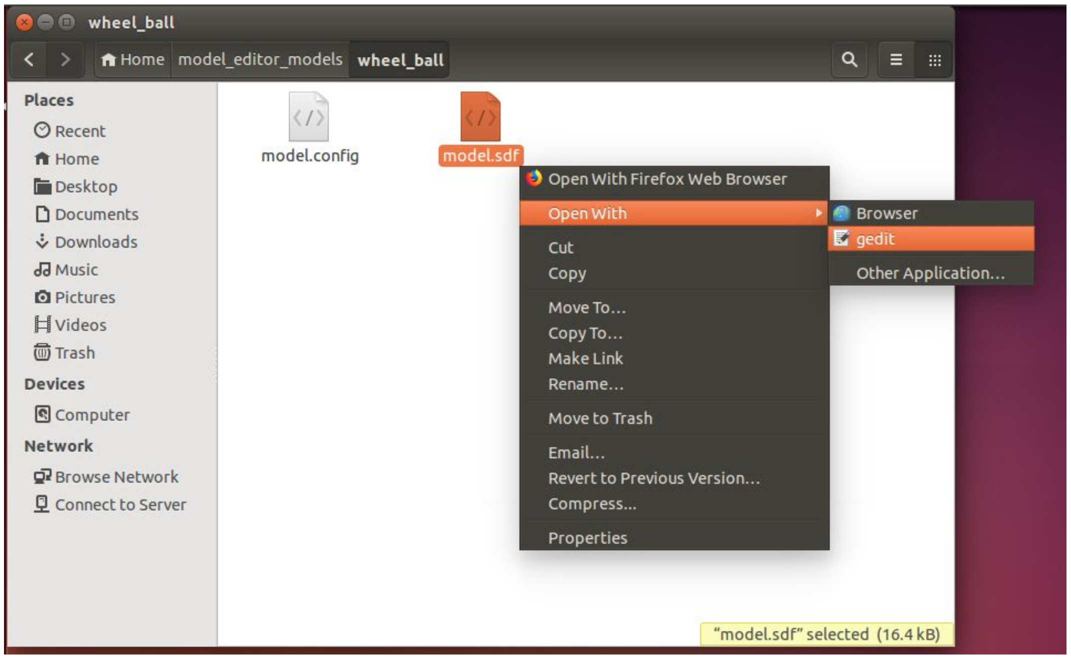
You can find all the Gazebo tutorials here: http://gazebosim.org/tutorials

TIP: When you aren’t running a simulation, you can hit the pause button at the bottom of the Gazebo window to lighten the load on your computer.

4. The model editor is one way to create a robot in Gazebo, but you can also work with the SDF robot description filetype file directly. You can find the full documentation on how this file is setup, and what the available commands are, here: http://sdformat.org/spec

For this assignment, you will modify the robot you created in the “Model Editor” tutorial. Close

Gazebo, and open the file folder by clicking on the Files icon on the left, navigate to: `model_editor_models/my_vehicle`
(Or wherever you saved your model). Right click on `model.sdf`, then click on `Open With -> gedit`



The SDF file has many nested structures, for example almost everything falls between the `<model>` and `</model>` tags, indicating that they are part of that model. Find the different `<link>` tags, representing the different parts of your model. If you didn't change their names, they will be called `'link_0'` (the base link), `'link_1'` (the first wheel), `'link_1_clone'` (the second wheel that you cloned), etc. In each link section, you can find all of the properties such as the mass and inertia. The `<visual>` tag tells Gazebo how to display the link. Changing the dimensions here will not affect the physics but just how it is displayed.

You will now make each link of your robot a different color. For each link, find the `<ambient>` tag which indicates the red, green, blue, and transparency values for Gazebo to display, between 0.0 and 1.0. Pick your favorite colors and set each wheel and the base to a different color.

Save the file, open Gazebo, and insert your robot into the new scene.

<u>ASSIGNMENT:</u> Take a screenshot of your creation and include it in your submission. Then close Gazebo again.

5. We will now see how to talk to Gazebo from Matlab. To do this we need to know the IP address of the virtual machine. Open a Terminal by clicking the icon on the left side of the screen, and type: `hostname -I`

You should see two sets of four numbers, such as: `192.168.56.101 10.0.3.15`
The first set, `192.168.56.101`, is how the virtual machine talks to your host computer. (The second set is how your virtual machine talks to the internet). Your IP address may be slightly different, for example `192.168.56.102`

Now we will start Gazebo again, but this time we want it to start with a connection to ROS, the Robot Operating System. To do this, in your open terminal type:

```
roslaunch gazebo_ros empty_world.launch
```

Then add in your robot from before to the environment. Back on your host computer, open Matlab. To tell Matlab where to find the ROS environment, set the IP address to the value you found earlier:

```
ipaddress='192.168.56.101';
rosinit(ipaddress)
```

The output should look like:

```
Initializing global node /matlab_global_node_28242 with NodeURI
http://192.168.56.1:49862/
```

Now lets see what ROS can provide for us. ROS uses “topics” to share data between different parts of the robot. For example, one topic might come from your camera, and another might be where you output control commands. To see what topics are available, type:

```
rostopic list
```

You should see something like this:

```
/clock
/gazebo/link_states
/gazebo/model_states
/gazebo/parameter_descriptions
/gazebo/parameter_updates
/gazebo/set_link_state
/gazebo/set_model_state
/rosout
/rosout_agg
```

You can see directly what data is being transferred by typing:

```
rostopic echo /clock
```

You should see the clock updates running by. Press the pause button back in Gazebo and the updates should stop. You can exit from `rostopic echo` by pressing Ctrl+C.

Note: Any time you are done working for a while, it is good practice to clear the workspace of publishers, subscribers, and other ROS-related objects when you are finished with them.

And it is recommended to shut down the global node and disconnect from Gazebo.

```
clear
rosshutdown
```

6. Now let us interact with Gazebo using some special Matlab scripts. These scripts use `rostopics` to get information and present them in a format that is easier to use in Matlab. These instructions roughly follow the tutorials from Matlab, starting here: https://www.mathworks.com/help/robotics/examples/read-model-and-simulation-properties-from-gazebo.html

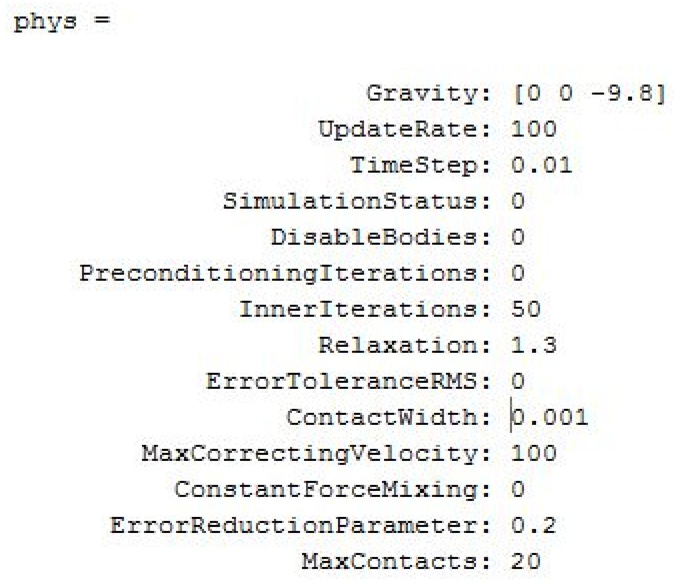
Set up a connection to the Gazebo world through the `ExampleHelperGazeboCommunicator` class, which makes interaction with Gazebo easier. In Matlab:

```
gazebo = ExampleHelperGazeboCommunicator();
```

Using the that object, extract the physics properties of the simulation.

```
phys = readPhysics(gazebo)
```

The output looks similar to this output:



```
phys =

                     Gravity: [0 0 -9.8]
                  UpdateRate: 100
                    TimeStep: 0.01
            SimulationStatus: 0
               DisableBodies: 0
   PreconditioningIterations: 0
             InnerIterations: 50
                  Relaxation: 1.3
           ErrorToleranceRMS: 0
                ContactWidth: 0.001
        MaxCorrectingVelocity: 100
          ConstantForceMixing: 0
      ErrorReductionParameter: 0.2
                  MaxContacts: 20
```

Gazebo uses SI units. To explore, change the gravity of the simulation so that it goes in the opposite direction in z. Reduce the value, too. Reset the simulation for the physics to affect all objects.

```
phys.Gravity = [0 0 0.1];       % Units are m/s^2
setPhysics(gazebo,phys);
resetSim(gazebo);
resumeSim(gazebo);
```

```
pause(3);                               % Let gravity take effect for 3 seconds
pauseSim(gazebo);
```

ASSIGNMENT: Take a screenshot of your robot floating in the air.

Now we will run the simulation at a reduced speed by changing the update rate on the physics. You can see how it looks running at 1/8 speed.

```
phys.UpdateRate = phys.UpdateRate/8;
phys.Gravity = [0 0 -9.8];   % Set gravity back to normal value (m/s^2)
setPhysics(gazebo,phys);
resumeSim(gazebo);
pause(15);
pauseSim(gazebo);
```

When you resume the simulation the robot falls back to the ground, but in slow motion. Note that the `pause(15)` function is in Matlab, so that time is not slowed down.

Switch the update rate back to normal so the simulation runs in real time

```
phys.UpdateRate = phys.UpdateRate*8;   % Set update rate back to normal value
setPhysics(gazebo,phys);
resetSim(gazebo);
```

7. The `gazebo` object allows you to find all models in the Gazebo world:

```
models = getSpawnedModels(gazebo)
```

Retrieve specific information about a model by creating an object for it with the `ExampleHelperGazeboSpawnedModel` class:

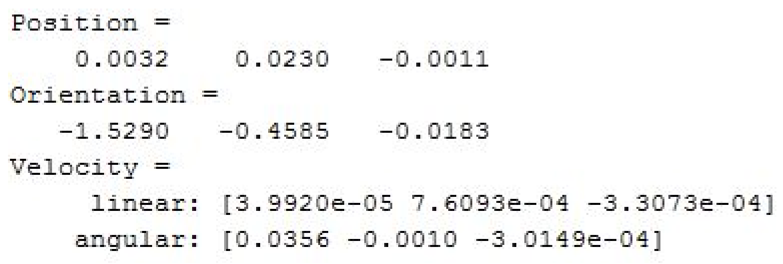
```
robot = ExampleHelperGazeboSpawnedModel('my_vehicle',gazebo)
[robotLinks, robotJoints] = getComponents(robot)
```

If your robot has a different name (as indicated from `getSpawnedModels`), replace text `'my_vehicle'` with that name.

Using the robot object you just created, obtain the position, orientation, and velocity by using the `getState` function.

```
[position, orientation, velocity] = getState(robot)
```

The output from the state commands looks like this output:



```
Position =
    0.0032    0.0230   -0.0011
Orientation =
   -1.5290   -0.4585   -0.0183
Velocity =
     linear: [3.9920e-05 7.6093e-04 -3.3073e-04]
    angular: [0.0356 -0.0010 -3.0149e-04]
```

8. You can also add new models to the Gazebo environment from Matlab. Use this function to display all Gazebo models available:

```
builtInModels = exampleHelperGazeboListModels()
```

To create a model, use the `ExampleHelperGazeboModel` class. In the following code, a ball is created. Define properties and spawn the ball:

```
ball = ExampleHelperGazeboModel('Ball')
addLink(ball,'sphere',0.2,'color',[0.3 0.7 0.7 1.0],'bounce',[1 10]);
spawnModel(gazebo,ball,[2.5,0,1]);
```

All units for Gazebo commands are specified in SI units. Depending on your view, you might have to zoom out to see the ball, because it is placed at [2.5, 0, 1]. If start the simulation you should see it bounce.

9. Finally, lets apply a torque to the robot's wheels and hit the ball. To do this, we will use the `robotJoints` variable we created earlier:

```
resetSim(gazebo);
stopTime = 3; % Seconds
effortLeft = 3.0; % Newton-meters
effortRight = 3.0; % Newton-meters
jointTorque(robot, robotJoints{1}, stopTime, effortLeft);
jointTorque(robot, robotJoints{2}, stopTime, effortRight);
resumeSim(gazebo);
```

Depending on exactly how your robot is set up, you may need to change with `robotJoints` you use, how much torque you apply, how big the ball is, etc. You can keep using `resetSim` to try again.

ASSIGNMENT: Get your robot to drive into the ball. Take a screenshot of your robot after it has kicked the ball forward.

As noted above, when you are finished run:

```
clear
rosshutdown
```

Then close Gazebo and shut down the virtual machine.

TURN IN: The screenshots from steps 4, 6, and 9.