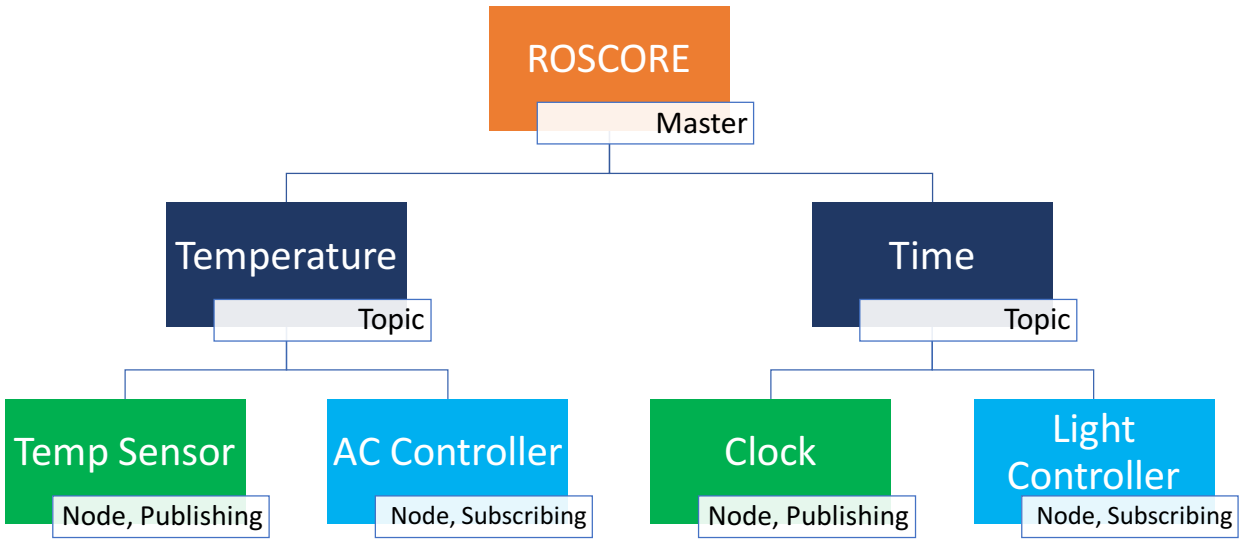
DARWIN ROS GAZEBO WORK AROUND MANUAL

What is ROS?

ROS – Robot Operating System is a good compilation of various tools/libraries used in Robotics. ROS comes in very handy when the scale of your project demands extensive processing and data handling. ROS has one master called ROSCORE. There are multiple 'Topics' that connect to the ROSCORE. Our programs/nodes can publish data to one Topic and any number of nodes/programs can subscribe to a Topic to receive specific data. This simple example to make it clear.

Components:
A temperature sensor, A LCD screen, An Air conditioner on/off controller, Clock and a Light Controller

Task:
Air conditioner is turned on and off to maintain a constant temperature in the room based on the measurements from the temperature sensor. The light is switched on/off every 6 hours. LCD screen displays both time and temperature.



Take a guess as to where the LCD screen Node would be placed in this graph above.
Answer: LCD screen node would subscribe to both the Topics as it need to display both temperature and time.

Darwin is definitely complicated with more than 100 topics.

Objective

Get Darwin to walk/run/squat and record the joint angles, velocities and torques from ROS Simulation.

Installing ROS

Please follow the instruction as in the ROS website

```
http://wiki.ros.org/kinetic/Installation/Ubuntu
```

To start ROS

Open new terminal and type

```
roscore
```

Minimise this terminal and let it stay until you shutdown. Use ctrl+c to stop ROS.

Installing Gazebo and other dependencies

Gazebo is one of the Graphical Visualizers for ROS.

Replace all hydro with your version of ROS and enter it in your terminal.

```
sudo apt-get install git ros-hydro-desktop-full gazebo ros-hydro-gazebo-plugi
ns ros-hydro-gazebo-ros ros-hydro-gazebo-ros-control ros-hydro-hector-gazebo
ros-hydro-hector-gazebo-plugins ros-hydro-effort-controllers ros-hydro-joint-
state-controller ros-hydro-joint-state-publisher ros-hydro-turtlebot-teleop
```

Create your own workspace

Workspace is more like a folder that contains all the files and codes for a project. As beginners it is best to create individual workspace for each project in ROS.
Note: Replace hydro with your version of ROS in the below snippet

```
cd ~
mkdir -p ros-darwin/src
cd ros-darwin/src
source /opt/ros/hydro/setup.bash
catkin_init_workspace
git clone https://github.com/HumaRobotics/darwin_description.git
git clone https://github.com/HumaRobotics/darwin_control.git
```

```
git clone https://github.com/HumaRobotics/darwin_gazebo.git
cd ..
catkin_make
source devel/setup.bash
```

Launch Gazebo

Open a new terminal

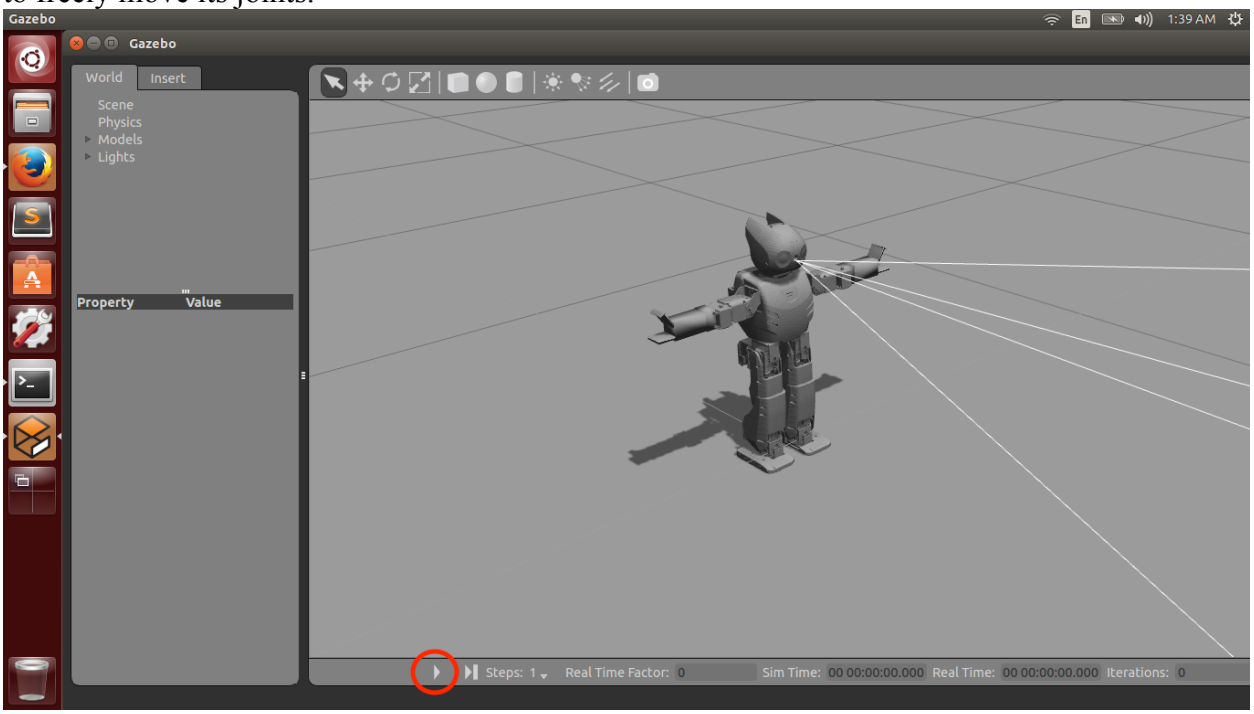
```
source ros-darwin/devel/setup.bash
roslaunch darwin_gazebo darwin_gazebo.launch
```

Please Note: From now on every new terminal should be sourced with the following command. This will not be mentioned every time in this manual. As a hint, you will see this "$ourced"

```
source ros-darwin/devel/setup.bash
```

You should now see Darwin standing still in the simulation. This might take a while depending on your computer. Look at the terminal and wait for it to stabilize.

In the Gazebo environment, click the play icon in the bottom tool bar. This allows your Darwin to freely move its joints.



Walking Demo - Just to make sure you got the above steps right

New Terminal - $ourced

```
rosrun darwin_gazebo walker_demo.py
```

If you see Darwin walk you are good. Else check if you have pressed play button the bottom tool bar. If it still does not work recheck your installation.

Nodes and Data

Open New Terminal

```
rostopic list
```

Prints all the topics that contain data published from Darwin. Feel free to look into each of it.

To observe data from each Topic, use this syntax

```
rostopic echo topic_name
rostopic echo /darwin/imu
rostopic echo /darwin/joint_states
```

For any open-cv based applications use the following command to get the camera feed

```
rosrun image_view image_view image:=/darwin/camera/image_raw
```

Controlling each joint

List of all the joints and their corresponding namespaces

```
/darwin/j_ankle1_l_position_controller/command
/darwin/j_ankle1_r_position_controller/command
/darwin/j_ankle2_l_position_controller/command
/darwin/j_ankle2_r_position_controller/command
/darwin/j_gripper_l_position_controller/command
/darwin/j_gripper_r_position_controller/command
/darwin/j_high_arm_l_position_controller/command
/darwin/j_high_arm_r_position_controller/command
/darwin/j_low_arm_l_position_controller/command
```

```
/darwin/j_low_arm_r_position_controller/command
/darwin/j_pan_position_controller/command
/darwin/j_pelvis_l_position_controller/command
/darwin/j_pelvis_r_position_controller/command
/darwin/j_shoulder_l_position_controller/command
/darwin/j_shoulder_r_position_controller/command
/darwin/j_thigh2_l_position_controller/command
/darwin/j_thigh2_r_position_controller/command
/darwin/j_thigh2_l_position_controller/command
/darwin/j_thigh2_r_position_controller/command
/darwin/j_tibia_l_position_controller/command
/darwin/j_tibia_r_position_controller/command
/darwin/j_tilt_position_controller/command
/darwin/j_wrist_l_position_controller/command
/darwin/j_wrist_r_position_controller/command
```

To turn head

```
rostopic pub /darwin/j_pan_position_controller/command std_msgs/Float64 – 1
rostopic pub /darwin/j_pan_position_controller/command std_msgs/Float64 –– –1
```

To walk with forward velocity of 1m/s

```
rostopic pub /darwin/cmd_vel geometry_msgs/Twist '[1,0,0]' '[0,0,0]'
```

To stop Darwin

```
rostopic pub /darwin/cmd_vel geometry_msgs/Twist '[0,0,0]' '[0,0,0]'
```

Play with numbers in '[]' section in the command

Note: If Darwin falls down, select *reset robot* and *reset world* from the top toolbar. In the worst case, restart gazebo.

Rosbag

Rosbag is a tool that lets you store the data published to a Topic and reuse it for testing and simulation. We use rosbag to record joints position, torques and velocity while Darwin is made to walk or stand in a specific pose. Our topic of interest here is `/darwin/joint_states`.

Recording

Once Darwin is in the desired state either walking/standing in a specific pose, we record the data published in '`/darwin/joint_states`' Topic into a rosbag and convert this rosbag to a csv file.

ROS publishes and subscribes at 10Hz. A 10 second recording will create 100 entries each consisting of joint angle, torque and velocity for each of the 24 joints. That leaves us with 3x2,400 numbers to organize and study for a recording duration of 10 seconds.

Get Darwin to walk; New Terminal - $ourced

```
rostopic pub /darwin/cmd_vel geometry_msgs/Twist '[1,0,0]' '[0,0,0]'
```

Start Recording; New Terminal - $ourced

```
rosbag record –O <filename>.bag /darwin/joint_states
```

Replace "<filename>" to anything you like. Example …–O jointsData.bag /darwin…

Keep recording for as much time you want and to stop recording press ctrl+c

Check the local directory to see if filename.bag is created. If not redo the recording process.

Replaying our Rosbag

Stop Darwin by pressing the stop button in the bottom toolbar.

Open new terminal - $ourced

```
rostopic echo /darwin/joint_states
```

This will not print any data now. Keep this terminal open and visible. Let us call this Terminal-1

Open another new terminal - $ourced
Make sure you are in the same directory as your rosbag

```
rosbag play filename.bag
```

Terminal 1 will display the stream of recorded data from our bag

If this is successful you are done with the recording part. Congratulations !

Rosbag to CSV

Download the bag_to_csv.py file from this link to the same folder that holds your bag file.
https://github.com/CPsridharCP/DarwinTempo/blob/master/bag_to_csv.py

This python script will convert any .bag file to a .csv file.
New terminal; navigate to the bag directory; substitute your file name in the following command

```
python bag_to_csv.py filename.bag
```

Executing this script will create a new folder in same directory containing a copy of out bag file and the newly created csv file.

CSV to .mat

The main objective is to push recorded data into matlab for further analysis. By the end of this tutorial we will create five .mat files named as mentioned below containing data stacked up in rows, each row corresponding to one time instance.

```
Position_Darwin.mat
Velocity_Darwin.mat
Torque_Darwin.mat
TimeStamp_Darwin.mat
JointNames_Darwin.mat
```

As we cannot create .mat file from python script we open matlab and create five empty files and copy it out local working directory. To reduce the workload you are provided with these empty files.

Download the five files from this link https://github.com/CPsridharCP/DarwinTempo

Also download the python script named Darwin_xlsx_to_matlab_program.py from the same link mentioned above to the directory holding the .bag and .csv file.

Open the CSV file we created in the previous step and save it as a .xlsx file. (use MS Excel)

There are six edits to the Darwin_xlsx_to_matlab_program.py script you need to complete.

Open Darwin_xlsx_to_matlab_program.py and update the .xlsx file name on the 9th line to the you .xlsx filename. Example,

```
9   wb = openpyxl.load_workbook(filename = 'darwin_joint_states.xlsx')
```

Update the empty .mat file path location in lines 87 to 91 to the file path in you computer. Example,

```
scipy.io.savemat('/Users/cplabs/Documents/Projects/C_Mummolo/Output_Matlab/Position_Darwin.mat', mdict={'arr': total_position}
scipy.io.savemat('/Users/cplabs/Documents/Projects/C_Mummolo/Output_Matlab/Velocity_Darwin.mat', mdict={'arr': total_velocity}
scipy.io.savemat('/Users/cplabs/Documents/Projects/C_Mummolo/Output_Matlab/Torque_Darwin.mat', mdict={'arr': total_torque})
scipy.io.savemat('/Users/cplabs/Documents/Projects/C_Mummolo/Output_Matlab/TimeStamp_Darwin.mat', mdict={'arr': total_time})
scipy.io.savemat('/Users/cplabs/Documents/Projects/C_Mummolo/Output_Matlab/JointNames_Darwin.mat', mdict={'arr': joint_names})
```

Save the file and execute it.

```
python Darwin_xnsx_to_matlab_program.py
```

After successful execution the empty .mat files will be rewritten with required date from Darwin.

Each .mat file is typically one multidimensional vector.

Squat instead of Walk

We can repeat the same for different postures of Darwin.

Instead of making the Darwin walk we can make the Darwin stand in a squat pose and redo the recording.

To make the Darwin Squat use this python script slow_squart.py
https://github.com/CPsridharCP/DarwinTempo/blob/master/slow_squart.py