1. Installing the Simulator

- Download a zipped copy of the source codes from the following Github repository: https://github.com/legorovers/pirover_simulator/tree/extended_interface
    - Ensure Python is installed on your machine (Python 2.7 or later). To see if Python is installed on your machine enter `python --version` on your command prompt or terminal.
    - To install Python, see the following guide: https://wiki.python.org/moin/BeginnersGuide/Download
    - It is helpful to use some environment manager to manage python environments. A manager is Anaconda: https://www.anaconda.com/what-is-anaconda
- Unzip the downloaded source into a location of your choice. In this document `simulator_folder` refers to the location that directly contains the Python file pysim.py and pysimosx.py

2. Running the Simulator

- In the command line prompt, navigate to the `simulator_folder`.
    - For Mac machines run `python pysimosx.py`
    - For Windows machines run `python pysim.py`
- Running pysimosx.py (or pysim.py) loads the following landing window.

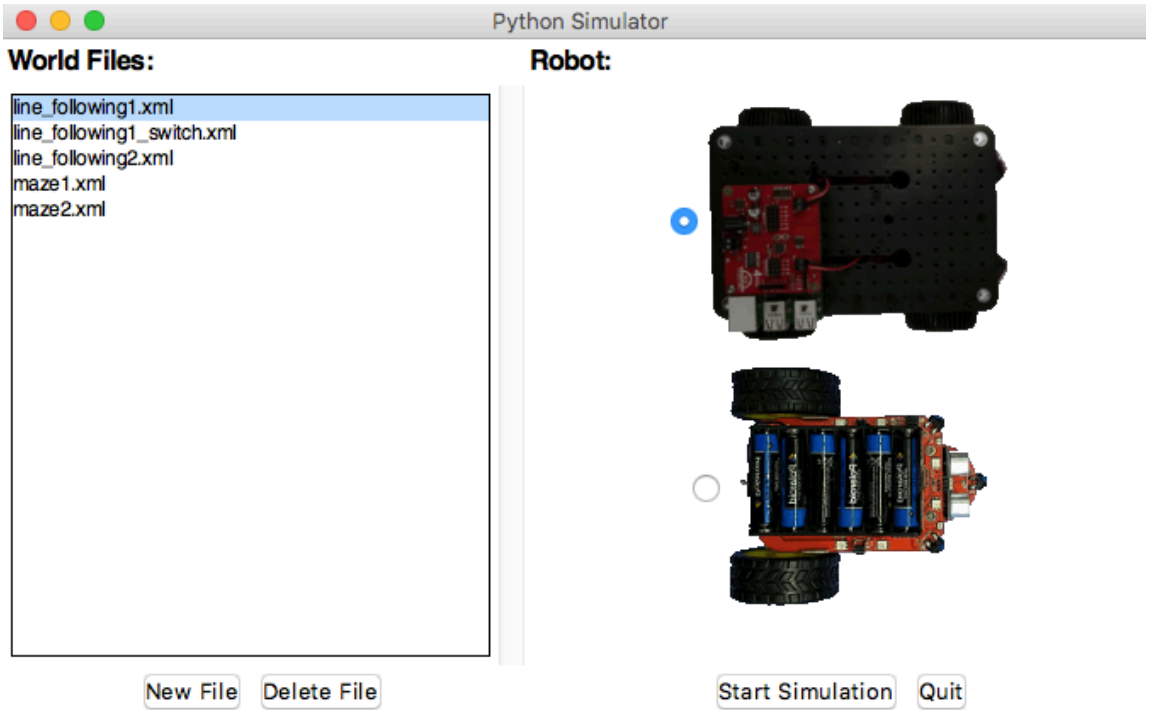

Figure 1: Landing Window for the Simulator

3. Loading a World

A world in the simulator is the environment in which the robot lives and moves. It consists of a background (terrain) and objects.

To see some sample worlds provided with the simulator, upon starting the simulator, select one of the worlds from the "World Files" list in the landing window (shown **Figure 1**), and press the "Start Simulation" button.

4. Loading a Custom World

A custom world is a world you have defined yourself. You can define a custom world by creating one in the `simulator_folder/worlds` folder of your simulator installation. You can start by copying and modifying one of the provided sample world files in `simulator_folder/worlds` folder. For example, you can add or remove more obstacles or line loops by adding/removing the corresponding tags for the desired objects in the world file. The provided sample files contain examples of all the tags that can be used to describe a world.

For setting the position of the objects in the tags, refer to the orientation of the Cartesian coordinates system with reference to the simulator window, shown in **Figure 2**.

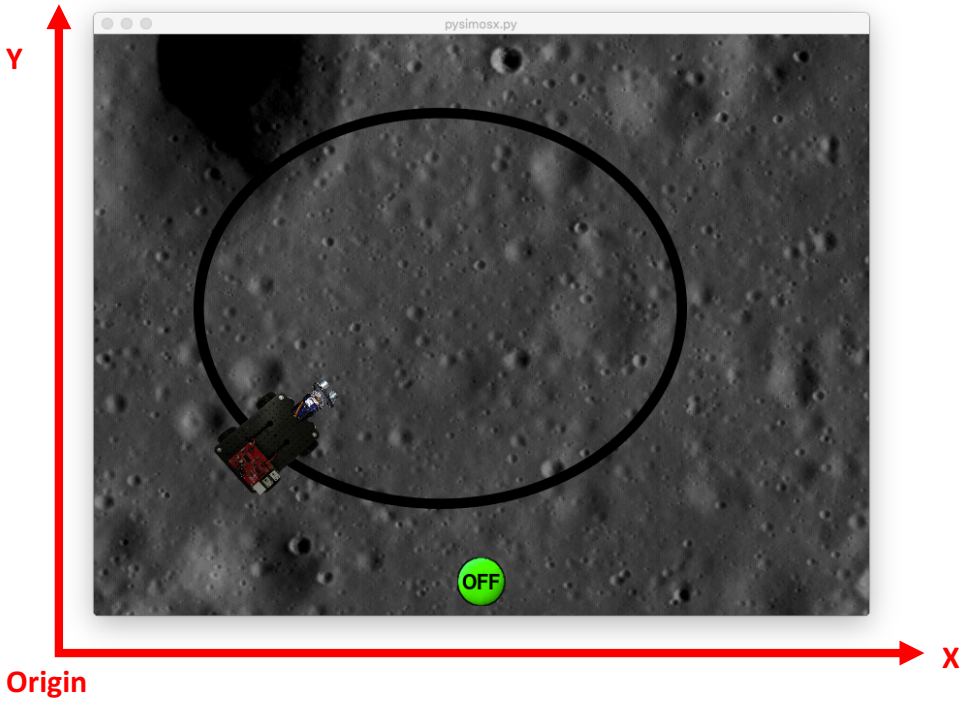

Figure 2: Showing Orientation of the Cartesian Coordinates System for the Main Simulator Window.

After creating a custom world file, ensure that you place the new worlds file in the `simulator_folder/worlds` folder. Then, your created world should appear as one of the "World Files" options on the landing window (see **Figure 1**) when you launch the simulator.

CAUTION: On the landing window, there is a "Delete File" button which not only removes the selected world file from the list of available world files, but also DELETES the actual world file from the `simulator_folder/worlds` folder.

5. Choosing a Robot

At the landing window (see **Figure 1**) notice that there is a choice of which robot to load.

- Initio
- Pi2Go

The features of Initio and Pi2Go correspond to those of the actual Initio and Pi2Go physical robots, as described in the following documentation: http://cgi.csc.liv.ac.uk/~lad/pirover

Quickly, one difference to note between the two robots is that Initio has no light emitting diodes (LEDs), whereas Pi2Go has 8 LEDs, just like the physical Pi2Go robots (see documentation at http://cgi.csc.liv.ac.uk/~lad/pirover/WS3.pdf).

6. World Modification at Simulator Runtime

While the Simulator is running, you can place or remove objects in the world by entering/enabling the **Edit Mode**.

To enter Edit Mode, press the letter 'e' on the keyboard when the simulator is running. At this point the Objects Pane will open right next to the main simulator window. See **Figure 5-9** for a description of the Objects Pane.

To exit the Edit Mode, press the letter 'e' again.

To place an obstacle in the world, first enable the Edit Mode. Then left-click on the desired obstacle in the Objects Pane – doing so highlights the selected obstacle. Then, right-click on the location you wish to place the obstacle within the main simulator window – the selected obstacle will be placed there.

All other objects are placed in the world in the same way an obstacle is placed. To place a new background (terrain) in the world, select the background you wish to place from the Objects Pane, then right-click anywhere within the main simulator window to replace the current background.

Objects placed in the world (except Backgrounds) can be dragged around with or without being in Edit Mode. A typical example use of this feature is while moving the light source around the world.

Apart from moving the light source itself, the ray/light beam from the light source can also be dragged to change its direction of focus around the light source object.

To remove an object from the world, first ensure that the Edit Mode is enabled. Then select (left-click) the 'delete' icon (see **Figure 9**) from the Objects Pane. Next, in the main simulator window, right-click on the object you wish to delete from the world – this will delete the object from the world.

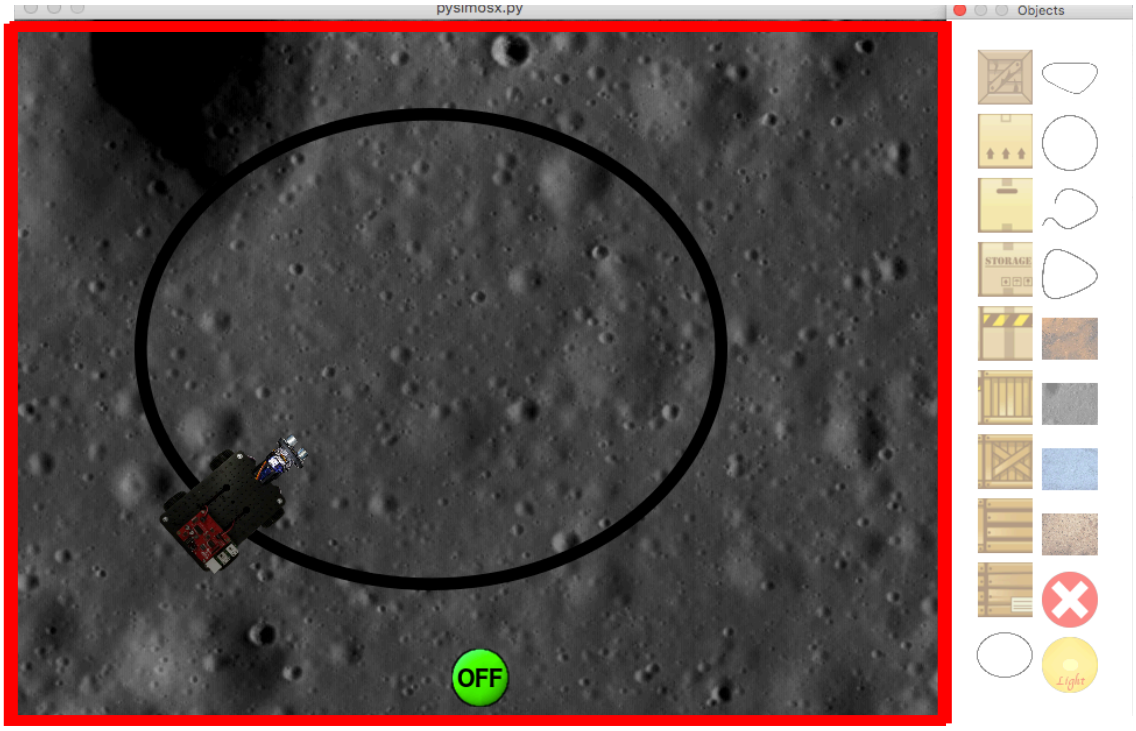

Figure 3: The Main Simulator Window is Highlighted in Red Colour.

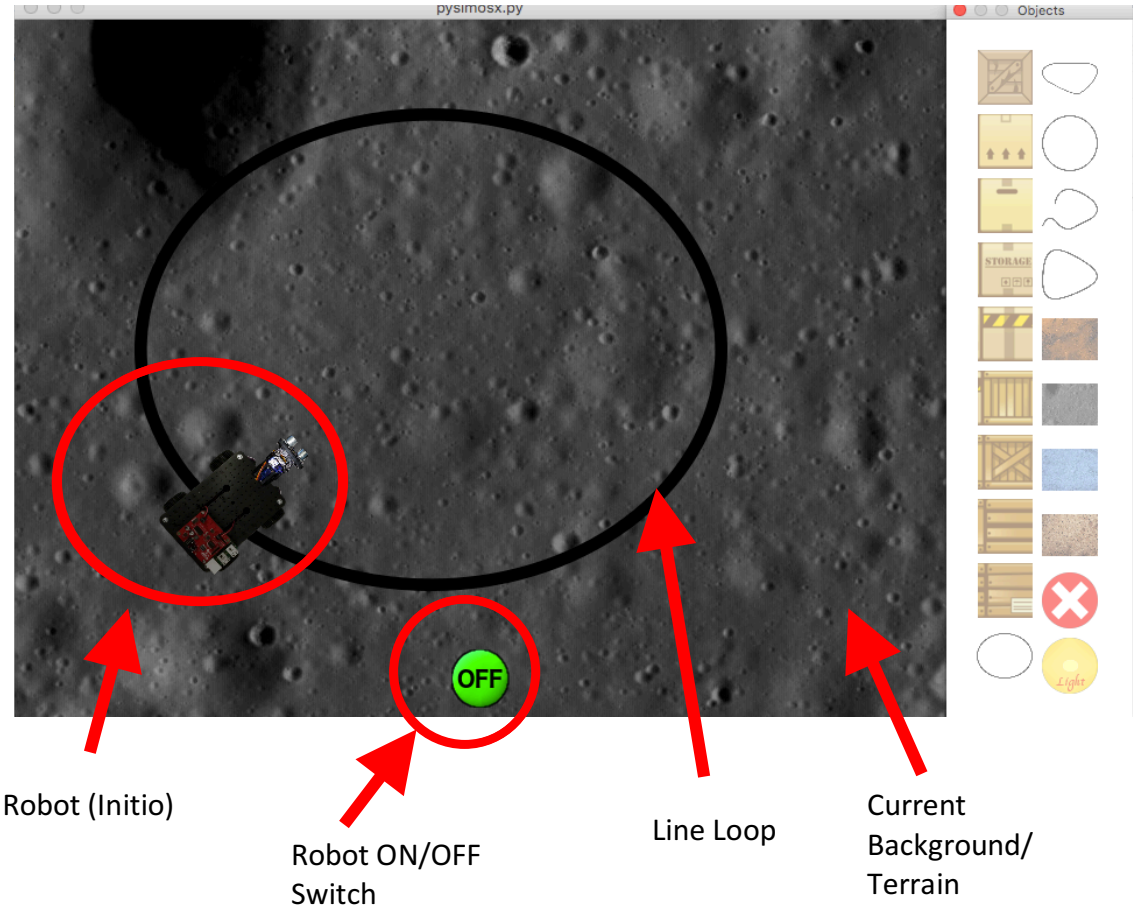

Figure 4: Highlighting Objects and Controls in the Main Simulator Window.

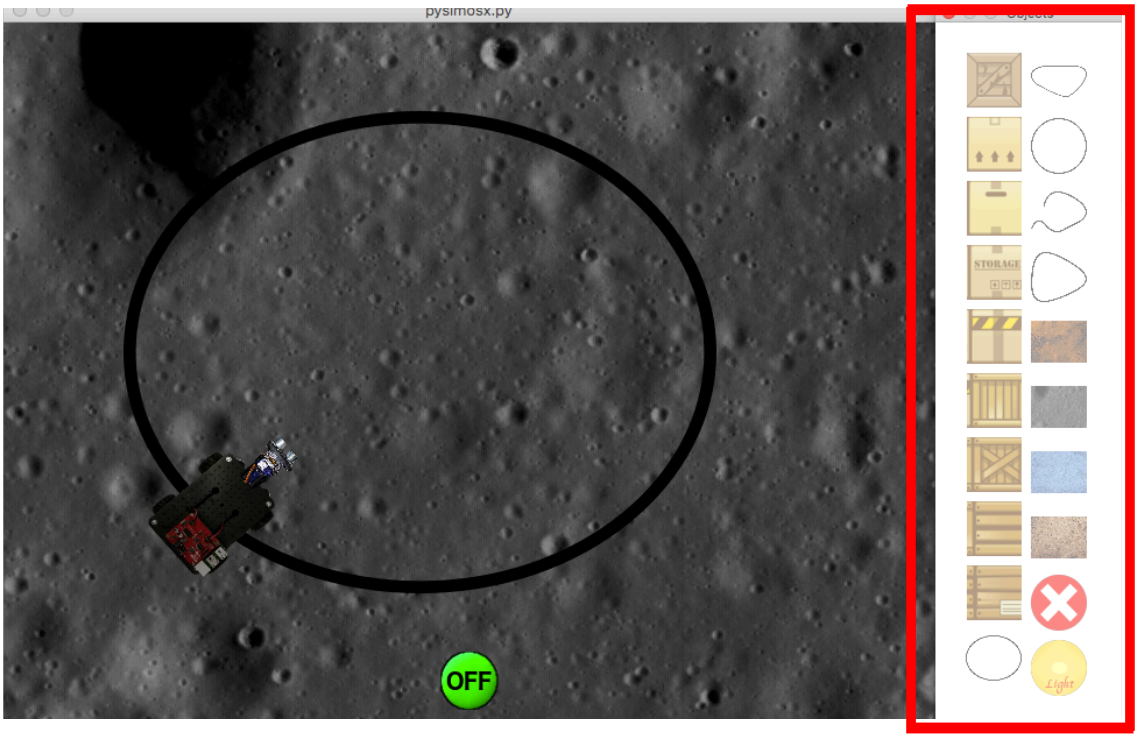

Figure 5: The Objects Pane is Highlighted in Red Colour.

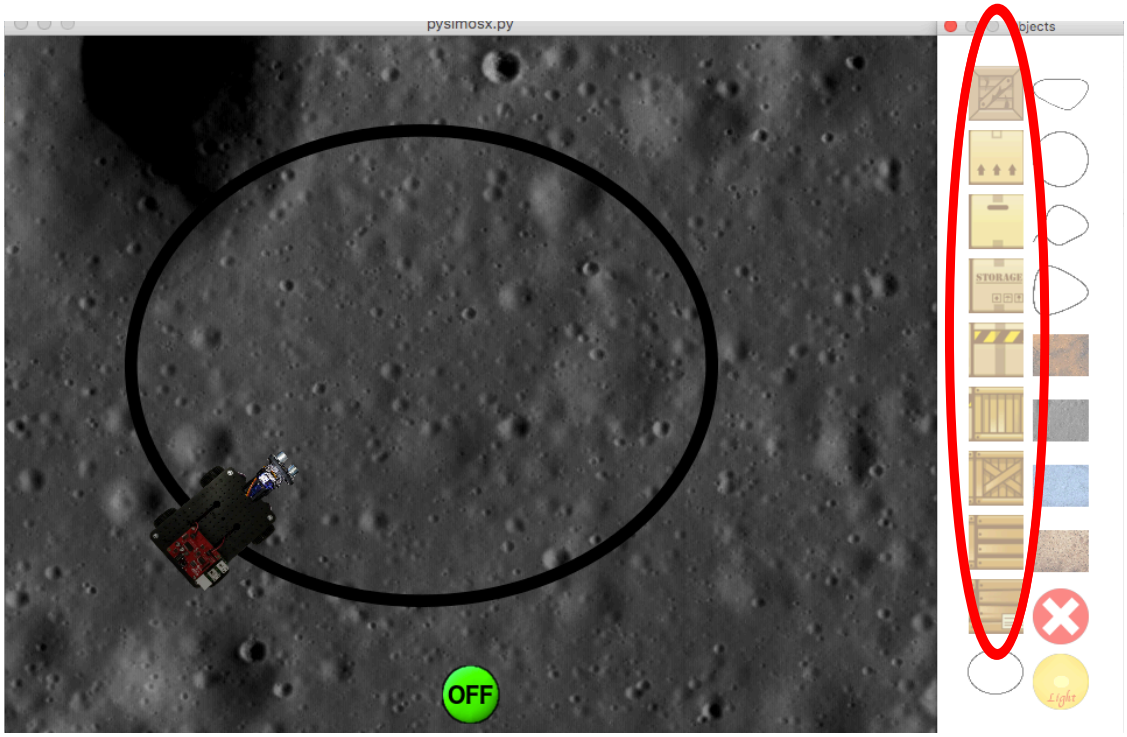

Figure 6: 'Obstacle' Objects are Highlighted in the Objects Pane.

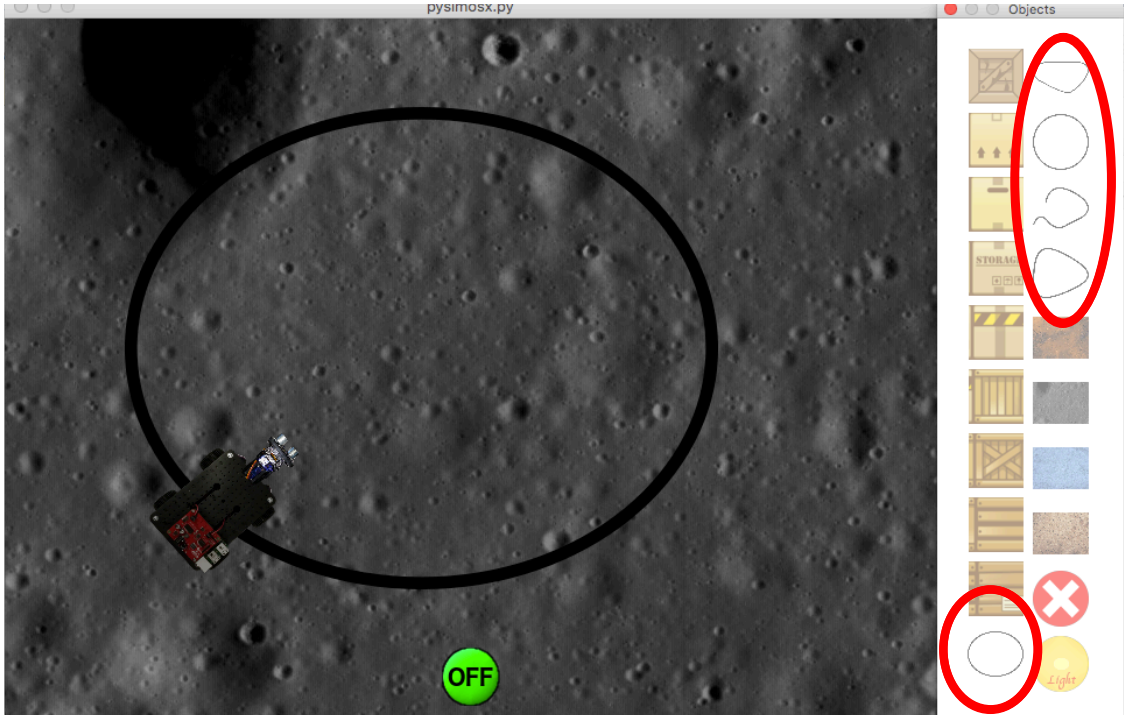

Figure 7: Line Loop Objects are Highlighted in the Obstacles Pane.

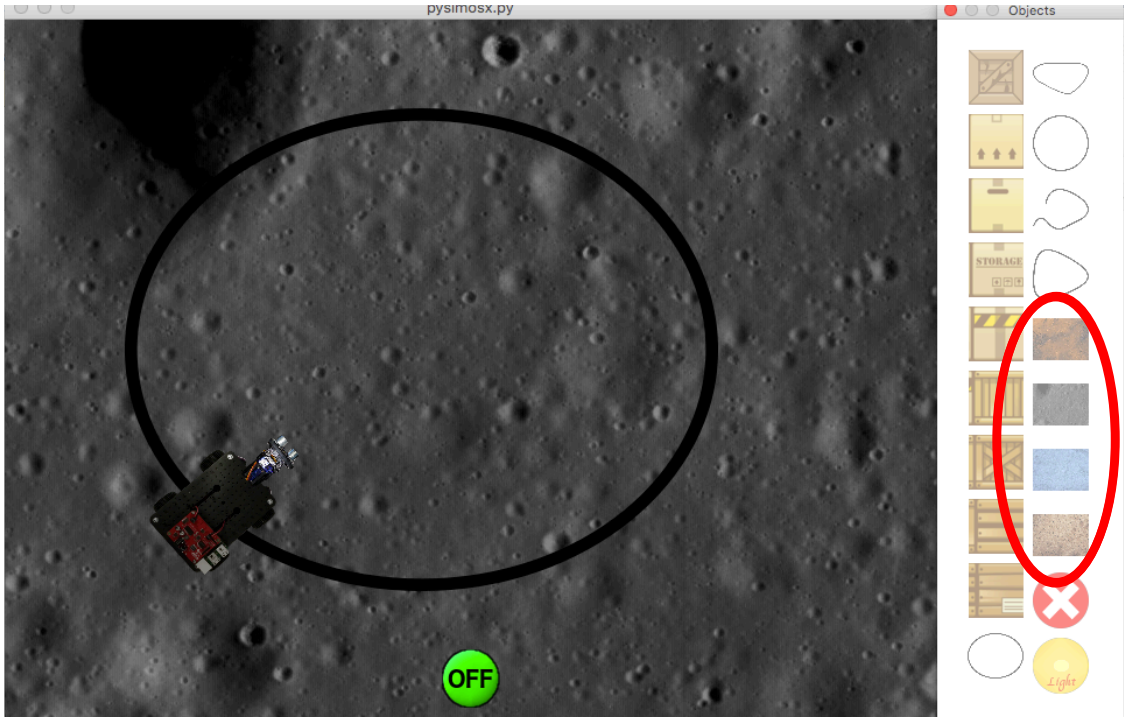

Figure 8: Background Objects are Highlighted in the Objects Pane.

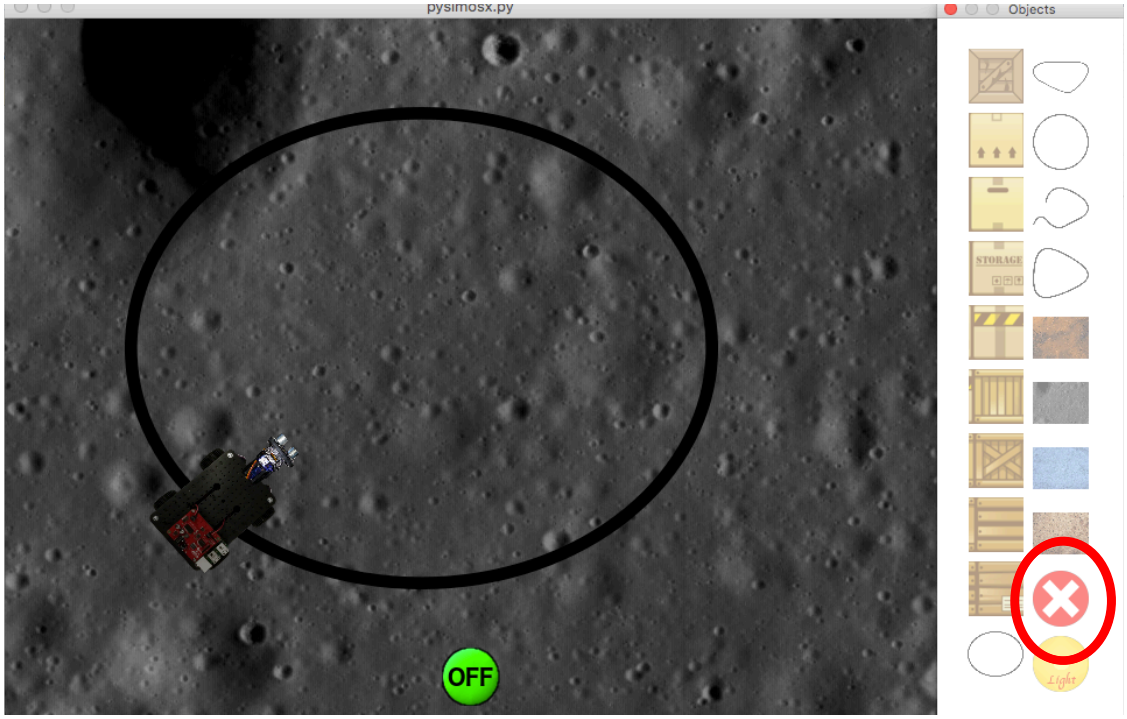

Figure 9: 'Delete' Control Icon is Highlighted in the Objects Pane.

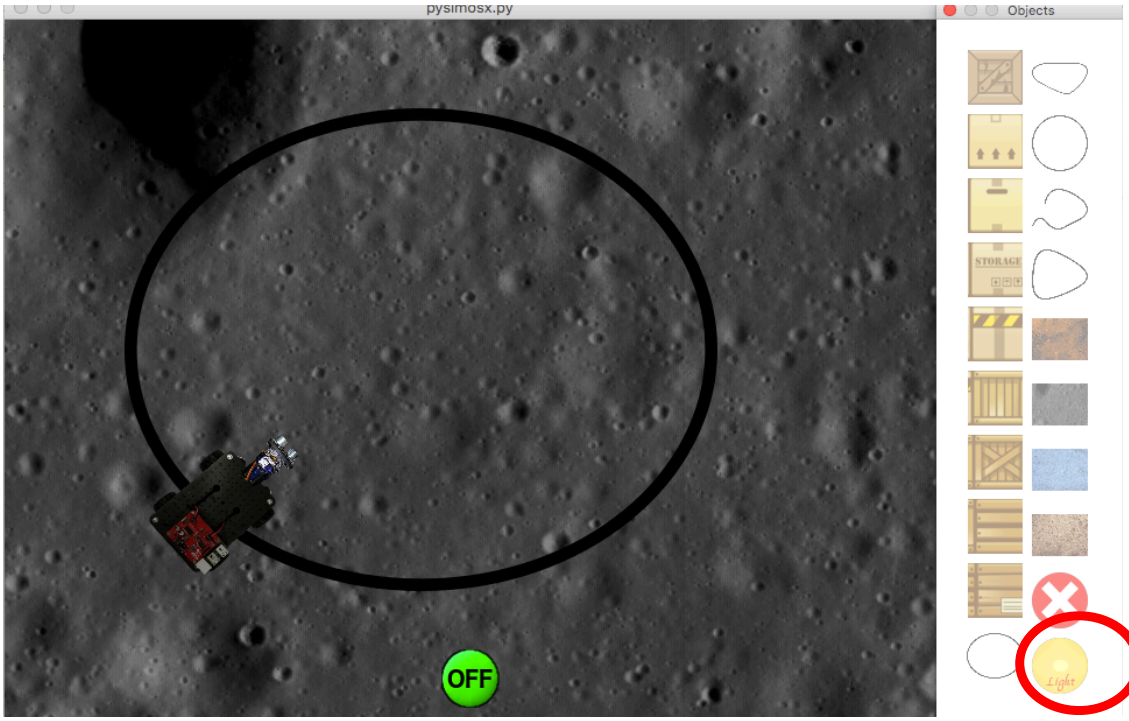

Figure 10: Light Source Control Icon is Highlighted in the Objects Pane.

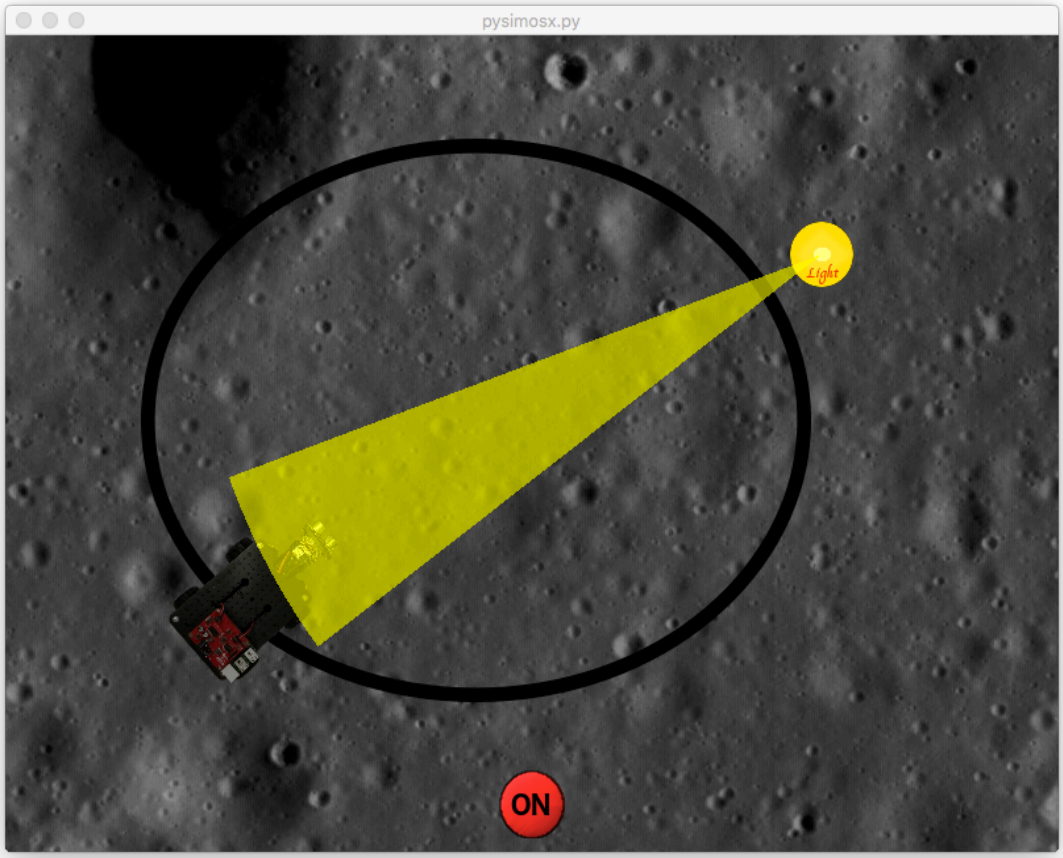

Figure 11: Observe the Light Source and Ray/Beam from the Light Source.

7. Running a Sample Code on the Simulator

The sample codes can be found at `simulator_folder/examples/` folder

- Ensure the that the path to the `simclient` folder is in the PYTHONPATH environmental variable.
- Run the simulator, loading a world and a robot as described in Sections 3, 4 and 5.
- Now open a different terminal or command window and navigate to the `simulator_folder`.
    - For example, to run the provided LineFollower.py sample, type the following at the terminal and press enter.
        - `python examples/LineFollower.py`
- Now go back to the main simulator window and click the switch button to turn it ON.
    - Clicking the switch button starts or resumes execution of the sample code on the robot (the button toggles to ON from the OFF state).
    - The sample code execution on the simulator can be paused by clicking the switch button again – at this point the switch button toggles back to the OFF state.

8. Creating and Running a New Sample Code for the Simulator

- You may begin by modifying one of the pre-packaged sample codes – that will at least indicate the essential module to import – the simrobot.py module, which contains the core programming interface functions needed to write sample programmes for the simulated robots (Initio and Pi2Go)
- Ensure that the path to the `simclient` folder is in the PYTHONPATH environmental variable.
- Using the functions in simrobot.py, and other ancillary modules of your choice, implement your desired code.
- Run your implemented code in the same way as the sample codes, as described in Section 7 (above) "Running a Sample Code on the Simulator".

9. Exiting the Simulator

Press Ctrl+C keys on the terminal/command line prompt to quit the simulator.

Alternatively, close all the windows (main simulator window and the Objects Pane, if open) using the Close Window button (**X**) at the top right corner of each of the windows. Once both the main simulator window and the Objects Pane are closed, the simulator also shuts down.