CSE598: PROJECT 3a

Team Members :

- ZAKK GIACOMETTI – REPORTER
- VIVEK VERMA – TECH LEAD
- VENKATESHWARAN SRINIVASAN - SUPPORTER

CONSTRUCTION OF THE WORLD

A floor plan was laid out using the building editor in gazebo, where .png file of the plan was uploaded to get the *correct scale* and *appropriate scale(px/m)*. Walls were laid out as per the plan by selecting it from *create walls*. Slight modifications from the original plans were along the way for a reaching a better lay out for the robot to traverse in. The layout consists of Diner, 2 rooms, living room, hallways and objects (randomly placed). Gaps between walls were left at places, for entry and exit into these rooms.

The world was then saved in the .sdf format. **Screenshots have been attached at the end of the report**.

The saved .sdf file was loaded into gazebo using the *Gazebo *filename** command. Object placing was done via *Insert* tab and then selecting the objects to put on the floor. Some objects included were Tables, Bookshelves, Cylinders.

INSTRUCTION ON RUNNING THE PROGRAM :

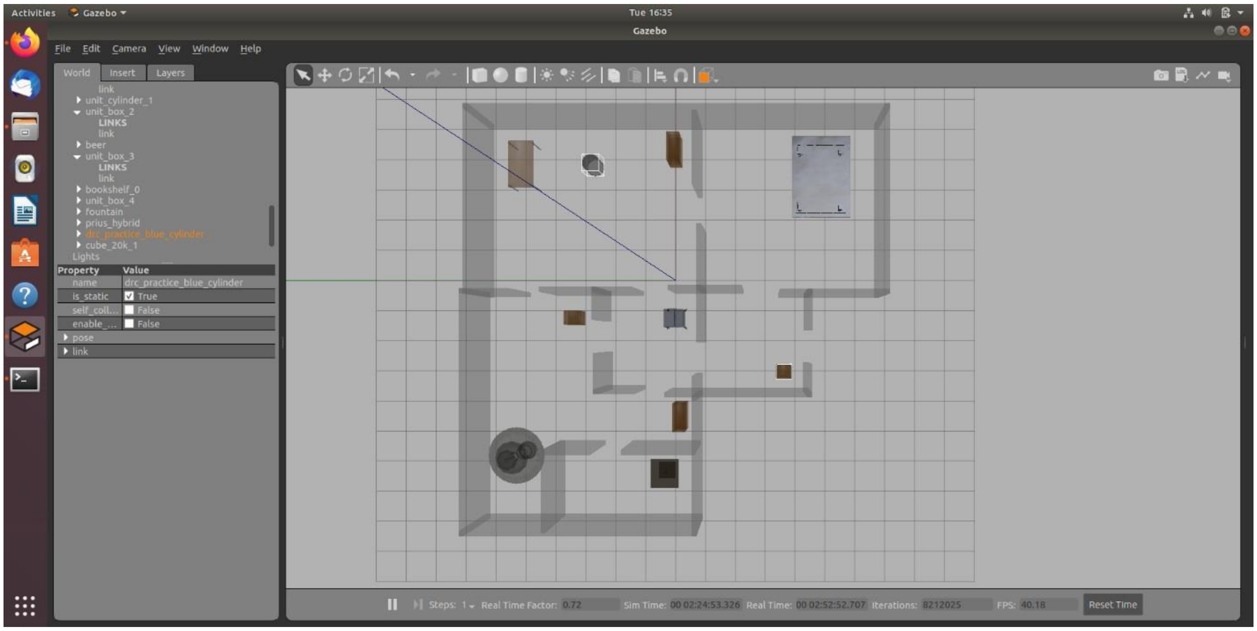
NOTE : If you've not already sourced bash files, you can do it by "*source devel/setup.bash"* before typing in any of the below commands.

- For running the node, first we must start the *roscore*.
- Now in a separate terminal we'll launch the world with the commands : "*roslaunch pursuitevasion pursuitevasion_world.launch"* . This'll spawn the world with the turtlebot in it.
- Now for the turtlebot to run, use the following command : "*rosrun pursuitevasion navigate".*

ABOUT THE NODE :

We've created a node called "*turtlebot_navigate"* written in python. It has a function navigate which takes the parameters(topic, commands), through which we're passing inputs to the robots in the form of (Linear x velocity, Linear z velocity, Time). A list of commands has been hard coded into the cmd_nav and the code in the function takes them one at a time. These messages are extracted into *clx , cwz , cdt* then published by the twist messages. Commands aren't executed until ROS publishes a valid time(non-zero).

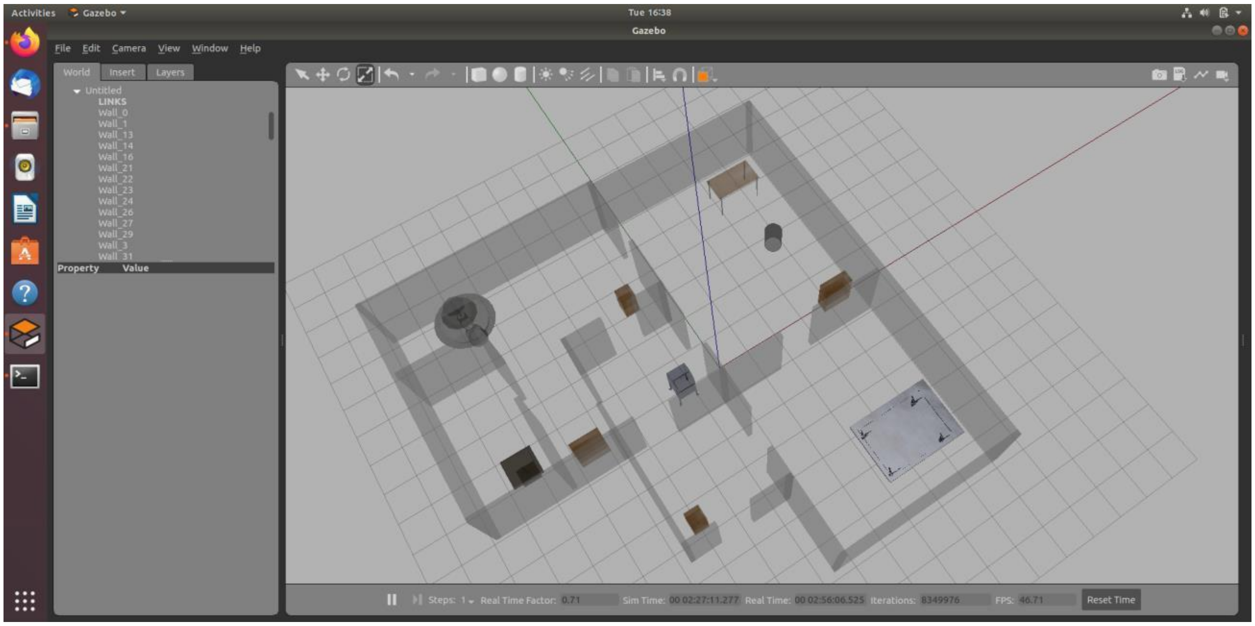
NOTE : The zip files are packages and should be unzipped in the *src* folder of the catkin workspace.



```
        link
    ▸ unit_cylinder_1
    ▾ unit_box_2
        LINKS
        link
    ▸ beer
    ▾ unit_box_3
        LINKS
        link
    ▸ bookshelf_0
    ▸ unit_box_4
    ▸ fountain
    ▸ prius_hybrid
    ▸ drc_practice_blue_cylinder
    ▸ cube_20k_1
  Lights
```

Property	Value
name	drc_practice_blue_cylinder
is_static	☑ True
self_coll...	□ False
enable_...	□ False
▸ pose	
▸ link	

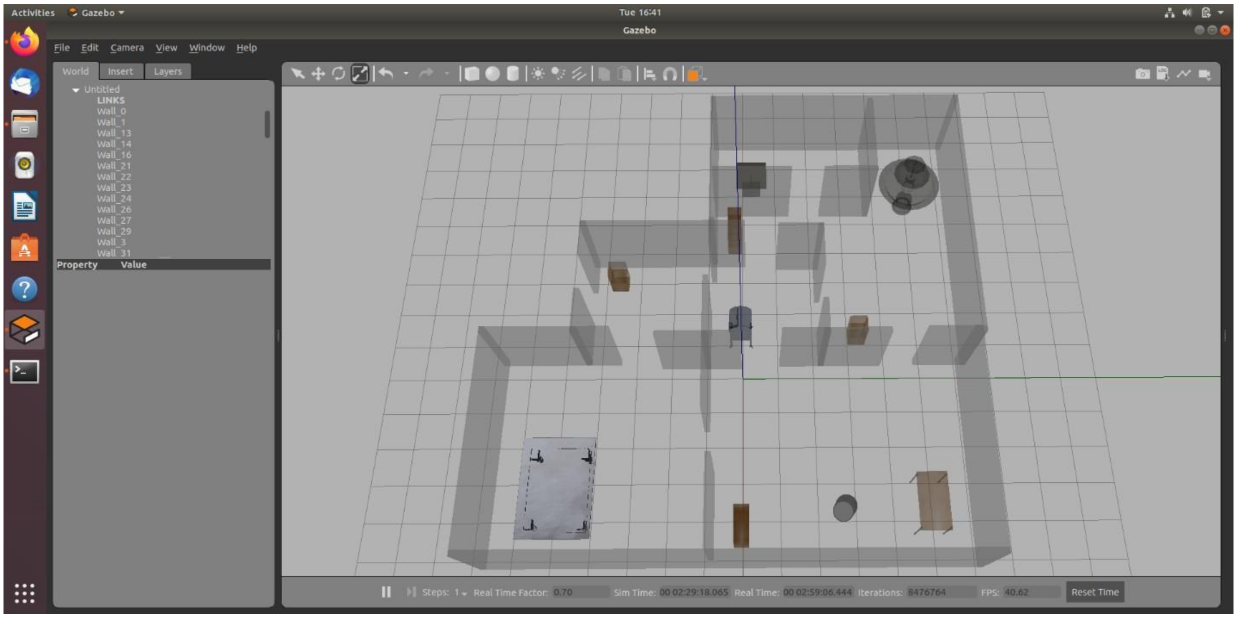
```
Steps: 1   Real Time Factor: 0.72   Sim Time: 00 02:24:53.326   Real Time: 00 02:52:52.707   Iterations: 8212025   FPS: 40.18   Reset Time
```



```
    ▾ Untitled
        LINKS
        Wall_0
        Wall_1
        Wall_13
        Wall_14
        Wall_16
        Wall_21
        Wall_22
        Wall_23
        Wall_24
        Wall_26
        Wall_27
        Wall_29
        Wall_3
        Wall_31
```

Property	Value

```
Steps: 1   Real Time Factor: 0.71   Sim Time: 00 02:27:11.277   Real Time: 00 02:56:06.525   Iterations: 8349976   FPS: 46.71   Reset Time
```



Untitled
LINKS
Wall_0
Wall_1
Wall_13
Wall_14
Wall_16
Wall_21
Wall_22
Wall_23
Wall_24
Wall_26
Wall_27
Wall_29
Wall_3
Wall_31

Property Value

Steps: 1 Real Time Factor: 0.70 Sim Time: 00 02:29:18.065 Real Time: 00 02:59:06.444 Iterations: 8476764 FPS: 40.62