# Assignment 3

The objective of this assignment is for you to become familiar with the Turtlebot 2 robot.

## 1. Ensure that your assigned robot is functional.

Look over the tutorials available in:
http://wiki.ros.org/Robots/TurtleBot
Your robot is using the kobuki base.
Follow the following articles:
http://spectrum.ieee.org/automaton/robotics/diy/new-turtlebot-tutorials
http://spectrum.ieee.org/automaton/robotics/diy/turtlebot-2-unboxing
http://www.cse.sc.edu/~jokane/teaching/574/notes-turtlebot.pdf

Follow the tutorials, in particular the depth image to laser scan:
http://wiki.ros.org/depthimage_to_laserscan

## 2. Random Walk 60%

Modify the code from Assignment 1 to perform random walk with the actual robot. Your robot should never hit an obstacle. Always ensure you have adequate clearance.

## 3. (Individual) Grid Mapping (start) 40%

The primary goal of this part of the assignment is to work on the problem of mapping, we will use an occupancy grid as the underlying map structure. **This part should be done individually!**

A change from the last assignment, please create a directory under **catkin_ws/src** using the following naming convention **LastnameInitial** for example for the instructor the directory should be:

```
>catkin_ws/src/RekleitisI/
```

Move your random_walk node under that directory and also unzip the grid_mapper under the same directory:

```
>catkin_ws/src/RekleitisI/random_walk
>catkin_ws/src/RekleitisI/grid_mapper
```

Compile your code using catkin_make and ensure to run the command from the ROS tutorial:

```
>. ~/catkin_ws/devel/setup.bash
```

to ensure ROS can find your executables.
Create a program that would implement Occupancy Grid Mapping in the long term. For now you are going to only plot the pose of the robot. Use the random walk implemented in assignment 1 to have a single robot wander around the environment. Use the provided code as a guideline on how to store and display an occupancy grid in the form of an image. Display the path of the robot. Remember to consider *Occupied*, *Unoccupied* and *Unknown* Cells.

Please find the complete package under grid_mapper.zip. Compile the code using catkin_make. The grid_mapper code can be run as:

```
>rosrun grid_mapper grid_mapper 60 60
```

The last two parameters indicate the dimensions of the world.

**4. Bonus question 20%**
Consider negative obstacles such as steps or drop offs. Monitor the floor plane and anything that drops lower than a few cm consider it an obstacle. Test you code in the lab, ensure that you are close enough to the robot to catch it if it miscalculates. Stay out of the field of view.

## Note:

There is a difference between image coordinates and Stage coordinates. See sample code and implement accordingly.
You need to have opencv 2.x and some boost packages installed. For the sample code:

- press SPACEBAR to save snapshot of occupancy grid canvas into folder where user opened the executable in.
- press X or x to quit.
- read over the code, and pay attention to the 2 TODO comments especially

Other new things in the project, i.e. why did I have to include a zip of the entire project:

- CMakeLists.txt has some lines needed to link project with Boost and OpenCV libraries
- package.xml has ros_dep line that asks project to depend on EXTERNALLY INSTALLED OpenCV library (recommended by ROS, see http://www.ros.org/wiki/opencv2)

For all questions prepare a written report with a screen (images saved in your program; see attached code) that document the progress of the mapping.

**Evaluation:**

Prepare a report presenting screenshots of your work, provide a brief overview on how the program works and also discuss the choices made. Place the report in the base directory with your name and zip everything into a single file. Submit in the dropbox. I will arrange with every team to see a demo of the random walk behaviour.