SIMULTANEOUS LOCALIZATION AND MAPPING USING LIDAR

Presentation by: **Mr. YONRITH Phayuth**

Advisor: **Dr. SRANG Sarot**

8th July 2020

Contents

1. Introduction
2. Occupancy Grid Map
3. Robotic Operating System (ROS)
4. Wheel Mobile Robot (WMR)
5. Light Detection and Ranging (Lidar)
6. Coordinate Frame
7. Simulation
8. Result and Discussion
9. Conclusion and Recommendation

Introduction

Background

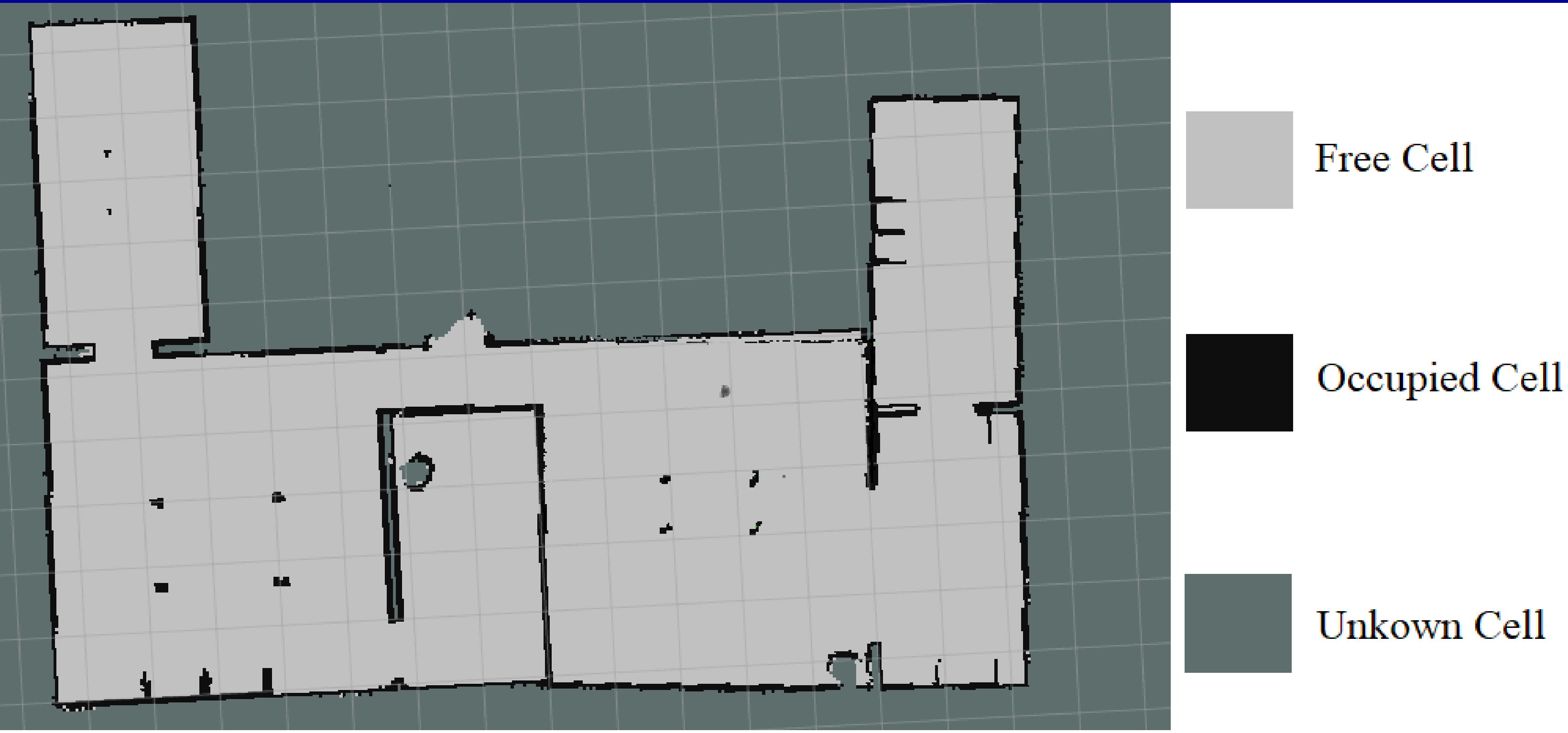
- Simultaneous Localization and Mapping (SLAM) is use for
  - localize the position of Robot in Global Frame
  - creating the map out of that environment.
- Lidar sensor have been used for SLAM
- The map that acquired from SLAM could be further use for Path Planning and Autonomous Navigation purpose.

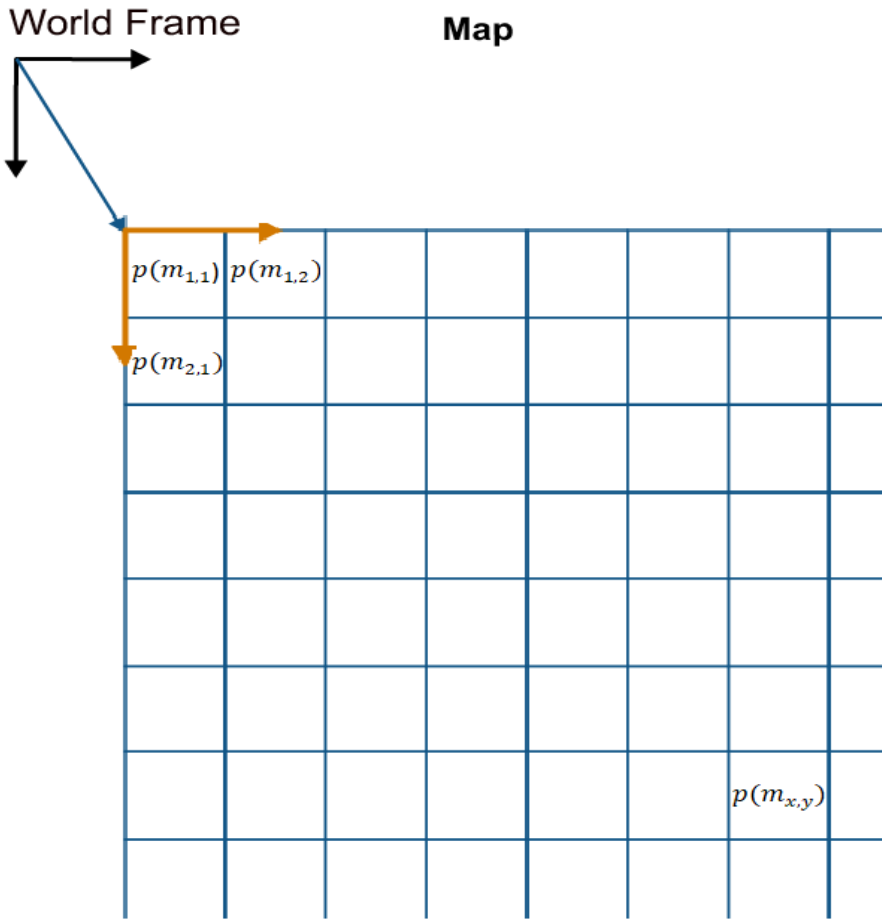
Introduction

Objective

- To build the **Occupancy Grid Map** of surrounding simulated environment using **Lidar**
- Implementing **Hector SLAM**, **Gmapping SLAM** and **MATLAB SLAM**.
- Using **Gazebo** 3D Simulation Software to simulate **surrounding environment, sensor** and **WMR**

Occupancy Grid Map





- The cell is either occupied or free,
- each probability value contain in each cell are independent,
- The surrounding environment is static.

$$m_{x,y} = \{free, occupied\} = \{0,1\}$$

Occupancy Grid Map

- The Occupancy Map uses a ***log-odds*** representation of the probability values for each cell.
- This representation efficiently updates probability values with the fewest operations. Thus, integrate sensor data into the map can be calculate quickly.
- Each probability value is converted to a corresponding *log-odds* value for internal storage.
- The value is converted back to probability when accessed.

$$odd(x\ event) = \frac{probability\ of\ x\ event\ happen}{probability\ of\ x\ event\ not\ happen} = \frac{p(x)}{p(-x)} = \frac{p(x)}{1-p(x)}$$

$$l(x) := \log \frac{p(x)}{1-p(x)} \qquad\qquad p(x) = 1 - \frac{1}{1 + expl(x)}$$

Occupancy Grid Map

Denoted that $p(m_i)$ is referred as the cell is occupied

We want to find the odd if the cell is occupied $l_{t,i} = log \dfrac{p(m_i \mid z_{1:t}, x_{1:t})}{1 - p(m_i \mid z_{1:t}, x_{1:t})}$

Using Bayes rule, $p(m_i \mid z_{1:t}, x_{1:t}) = \dfrac{p(m_i|z_t, x_t)\, p(z_t|x_t)\, p(m_i|z_{1:t-1}, x_{1:t-1})}{p(m_i)\, p(z_t|z_{1:t-1}, x_{1:t})}$

For $1 - p(m_i \mid z_{1:t}, x_{1:t}) = p(-m_i \mid z_{1:t}, x_{1:t})$

Thus $p(-m_i \mid z_{1:t}, x_{1:t}) = \dfrac{p(-m_i|z_t, x_t)\, p(z_t|x_t)\, p(-m_i|z_{1:t-1}, x_{1:t-1})}{p(-m_i)\, p(z_t|z_{1:t-1}, x_{1:t})}$

$$\frac{p(m_i \mid z_{1:t}, x_{1:t})}{1 - p(m_i \mid z_{1:t}, x_{1:t})} = \frac{p(m_i|z_t, x_t)}{p(-m_i|z_t, x_t)} \frac{p(m_i|z_{1:t-1}, x_{1:t-1})}{p(-m_i|z_{1:t-1}, x_{1:t-1})} \frac{p(-m_i)}{p(m_i)}$$

Occupancy Grid Map

$$log \frac{p(m_i\,|z_{1:t},x_{1:t})}{1\ -\ p(m_i\,|z_{1:t},x_{1:t})} = \log \frac{p(m_i|z_t,x_t)}{p(-m_i|z_t,x_t)} + log \frac{p(m_i|z_{1:t-1},x_{1:t-1})}{p(-m_i|z_{1:t-1},x_{1:t-1})} + log \frac{p(-m_i)}{p(m_i)}$$

$$l_{t,i} = l(m_i|z_{1:t},x_{1:t}) = l(m_i|z_t,x_t) + l(m_i|z_{1:t-1},x_{1:t-1}) - l(m_i)$$

$$l_{t,i} = inverse\_sensor\_model(m_i,x_t,z_t) + l_{t-1,i} - l_0$$

Algorithm occupancy_grid_mapping($\{l_{t-1,i}\},x_t,z_t$):

  For all cell m_i do

    If m_i in perceptual field of z_t then

      $l_{t,i} = l_{t-1,i} + inverse\_sensor\_model(m_i,x_t,z_t) - l_0$

    else

      $l_{t,i} = l_{t-1,i}$

    endif

  Endfor

return $\{l_{t,i}\}$

Occupancy Grid Map

Algorithm inverse_lidar_sensor_model(i, x_t, z_t):

Let x_i, y_i be the center of mass of m_i

$r^i = \sqrt{\left(m_x^i - x_{1,t}\right)^2 + \left(m_y^i - x_{2,t}\right)^2}$

$\phi^i = \tan^{-1}\frac{(m_x^i - x_{1,t})}{(m_y^i - x_{2,t})} - x_{3,t}$

$k = argmin_j|\phi^i - \phi_j^s\|$

If $r > \min(r_{\max}^s)$

return l_0

If $r_k^s < r_{\max}^s$

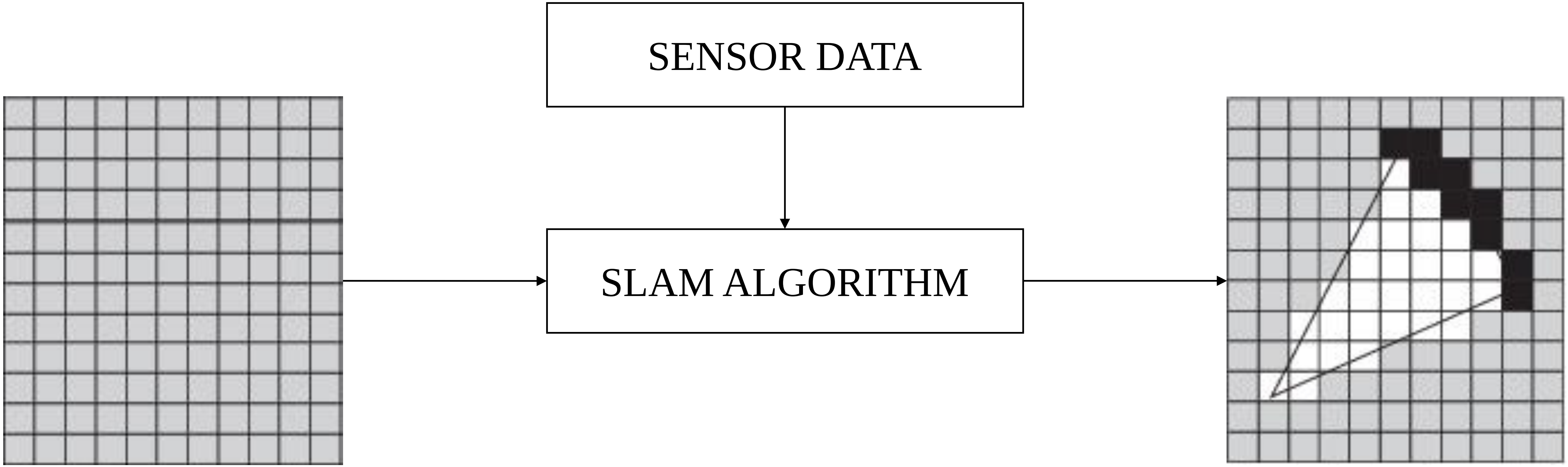
return l_{occ}

If $r^i \leq r_k^s$

return l_{free}

endif

Occupancy Grid Map



Prior Map
At time t-1

Posterior Map
At time t

Occupancy Grid Map

Scan Matching

To find the rigid transformation of robot pose that make the best lidar scan alignment, we have to find transformation $\xi = (p_x, p_y, \phi)^T$ that minimizes

$$\xi^* = argmin_\xi \sum_{i=1}^{n} [1 - M(S_i(\xi))]^2$$

Where

$S_i(\xi)$ are the world coordinates of scan endpoint $s_i = \begin{pmatrix} s_{i,x} \\ s_{i,x} \end{pmatrix}$

$S_i(\xi)$ is the function of ξ , the pose of the robot in the world coordinate.

Given some starting estimate of ξ, we want to estimate $\Delta\xi$ which optimize the error measure according to

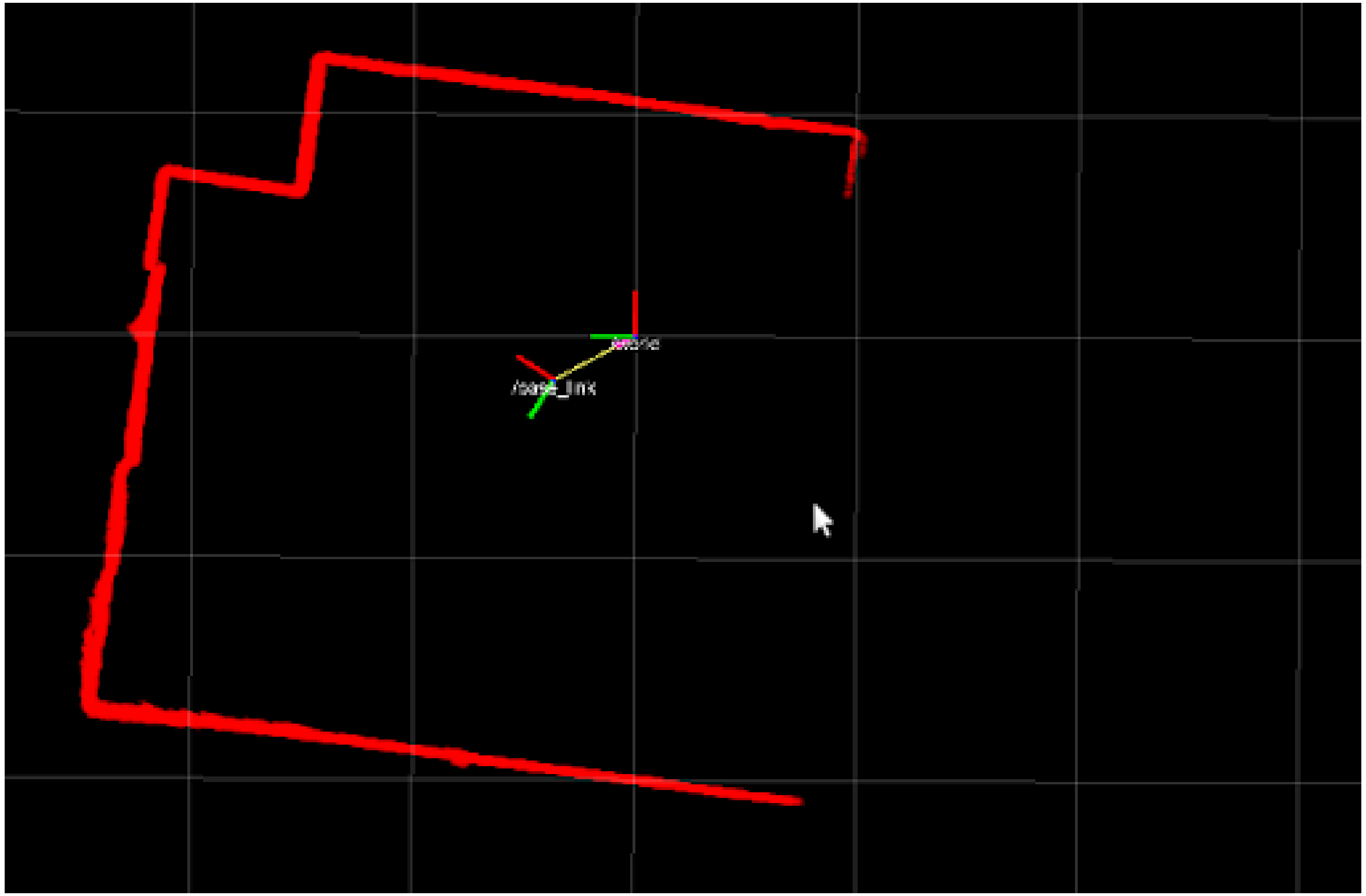
$$\sum_{i=1}^{n} [1 - M(S_i(\xi))]^2 \rightarrow 0$$

Solving for $\Delta\xi$ yields the Gauss-Newton equation for the minimization problem

$$\Delta\xi = H^{-1}\sum_{i=1}^{n}\left[\left[\nabla M\big(S_i(\xi)\big)\frac{\partial S_i(\xi)}{\partial \xi}\right]^T [1 - M(S_i(\xi))\right]$$

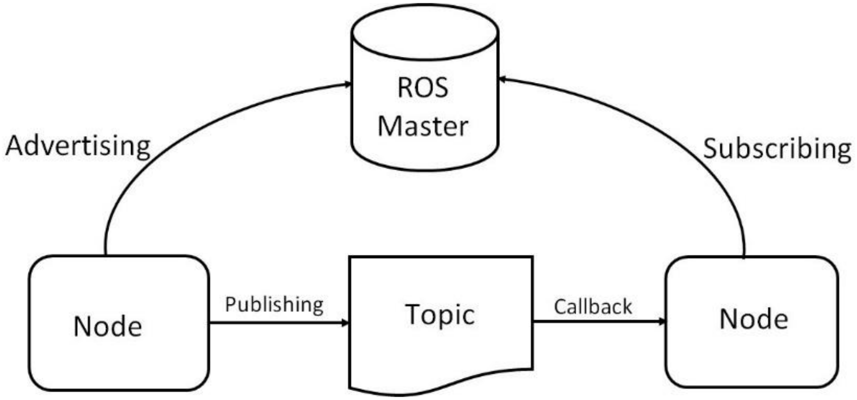
Where

$$H = \left[\nabla M\big(S_i(\xi)\big)\frac{\partial S_i(\xi)}{\partial \xi}\right]^T \left[\nabla M\big(S_i(\xi)\big)\frac{\partial S_i(\xi)}{\partial \xi}\right]$$



Visualization of Rigid transformation of robot pose from lidar scan

Robotic Operating System (ROS)



ROS Components

- 'ROS Master' is the center of ROS
- 'Node' is ROS computation
- 'Message' is ROS data
- 'Publishing' is the data output
- 'Subscribing' is the data reading
- 'Topic' is the name of buses for ROS data

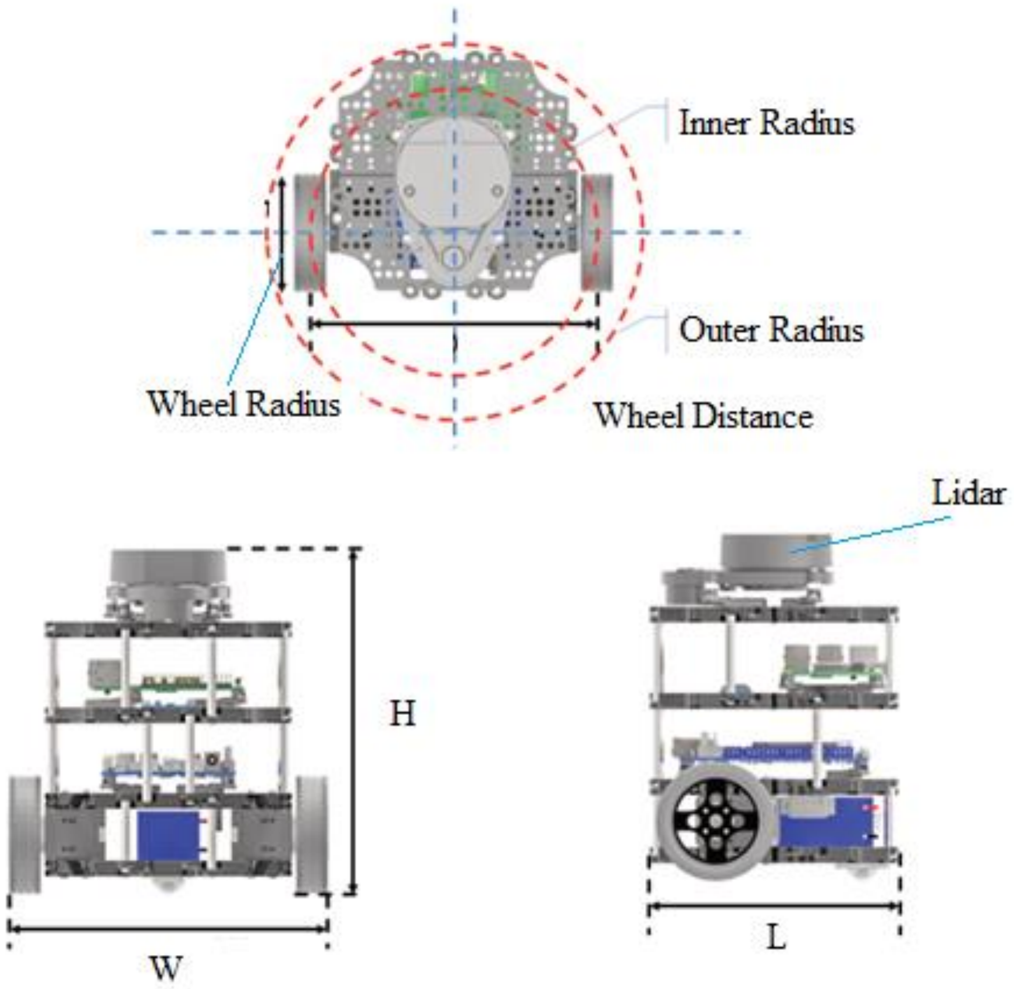
ROS usage

- Gazebo Simulation
- Tf Package
- SLAM Packages

(Hector, Gmapping and MATLAB SLAM)

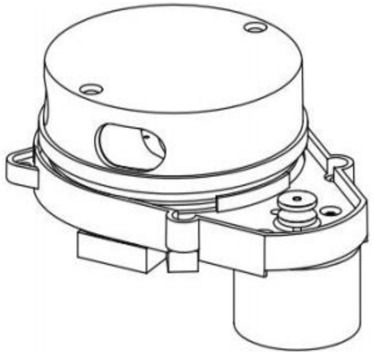
- RVIZ

Wheel Mobile Robot (WMR)



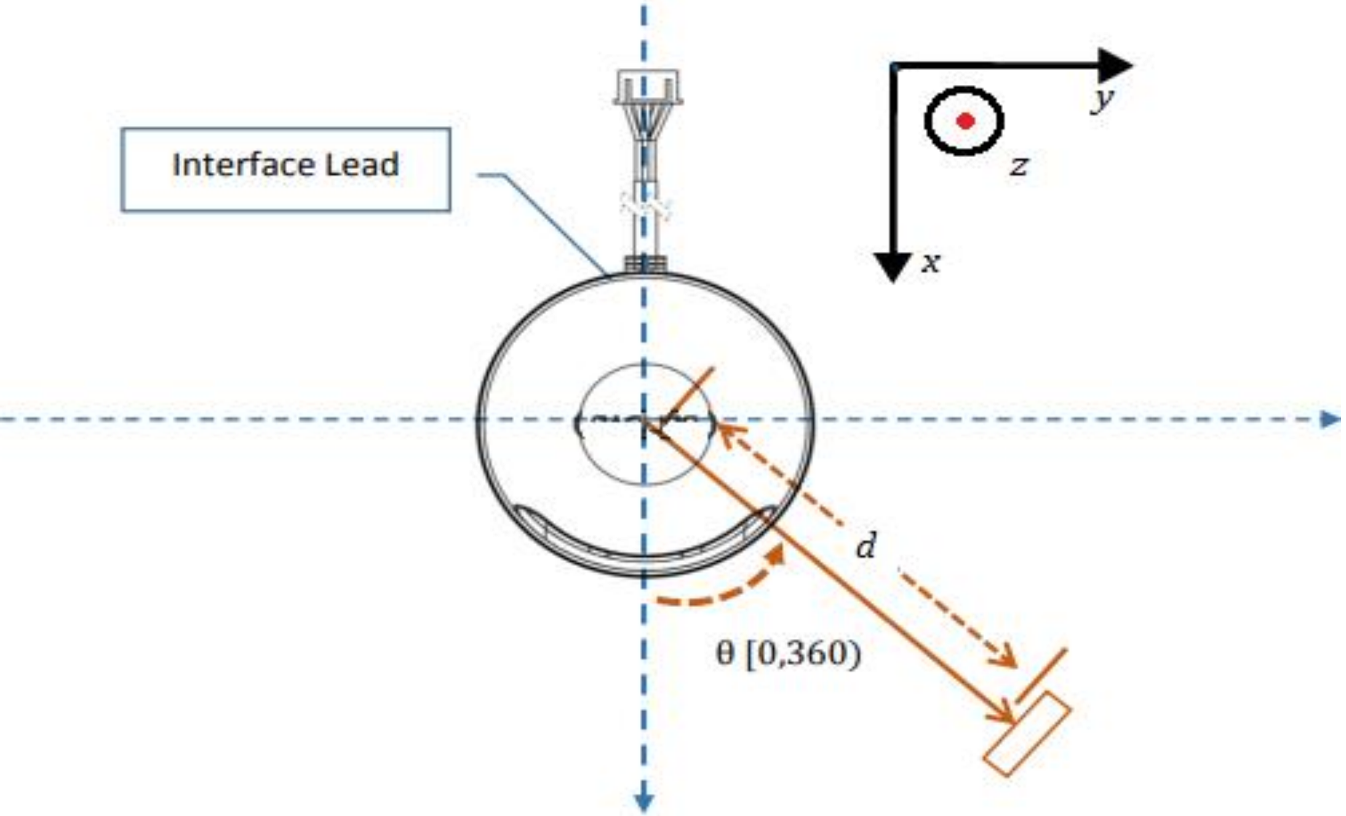
Parameters	Value	Unit
Wheel Radius	66	mm
Robot Width (W)	178	mm
Robot Length (L)	138	mm
Robot Height (H)	192	mm
Distance between wheel	160	mm
Robot Outer Radius	105	mm
Robot Inner Radius	80	mm
Maximum Translational velocity	0.22	mm/s
Maximum Rotational velocity	2.84	rad/s

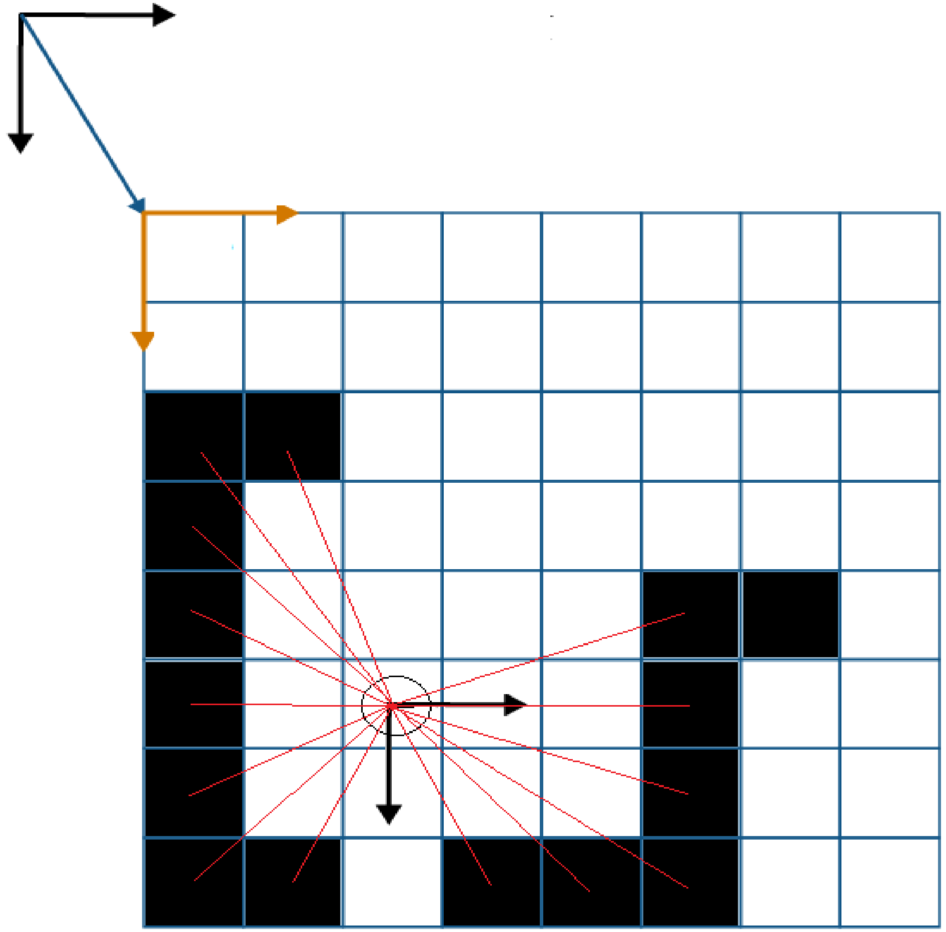
Turtlebot3 burger 2 wheels differential drive



360 degrees angle Lidar

2D coordinate(x, y) with rotation angle θ

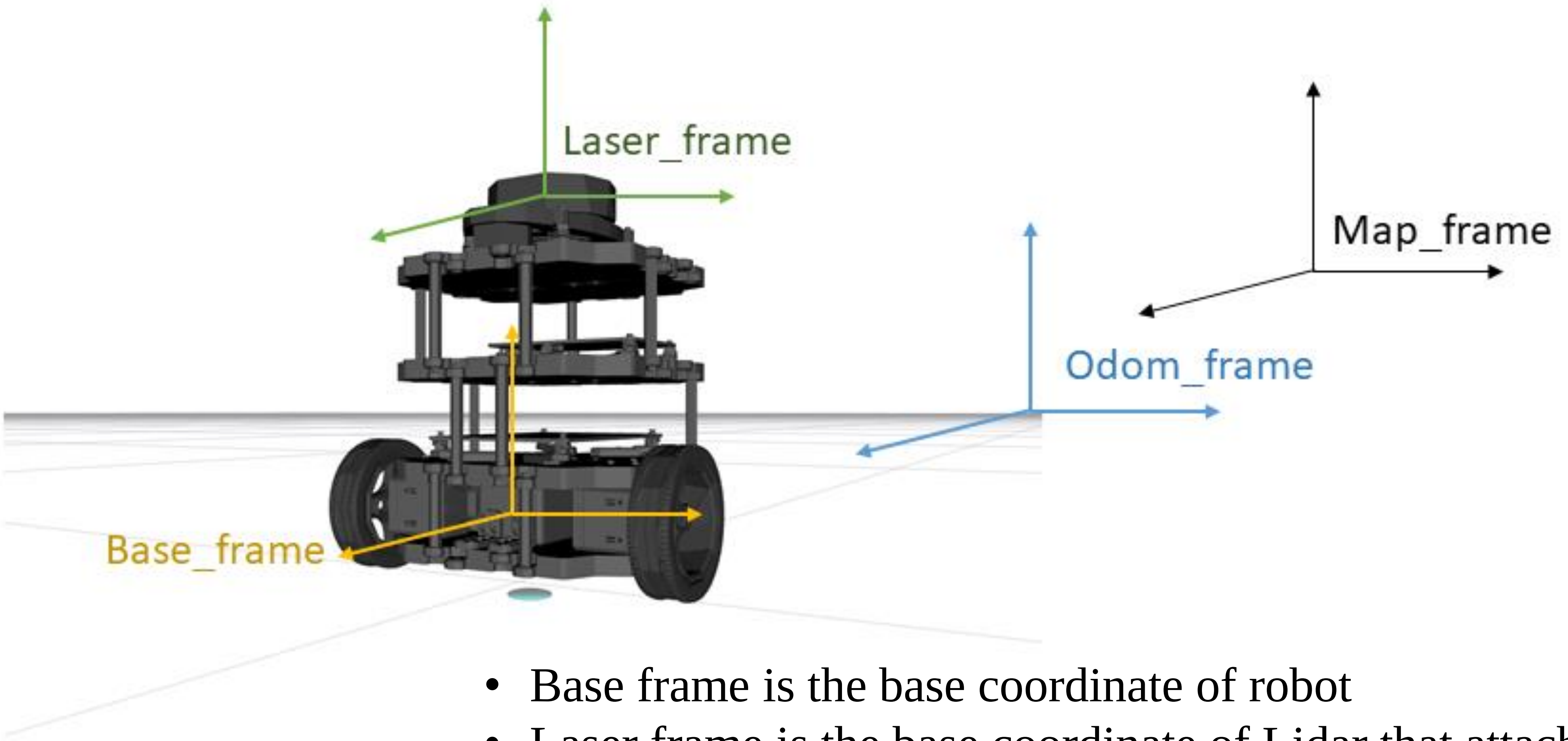




$$\theta = [\theta_1, \theta_2, \dots \dots \dots, \theta_{360}]$$

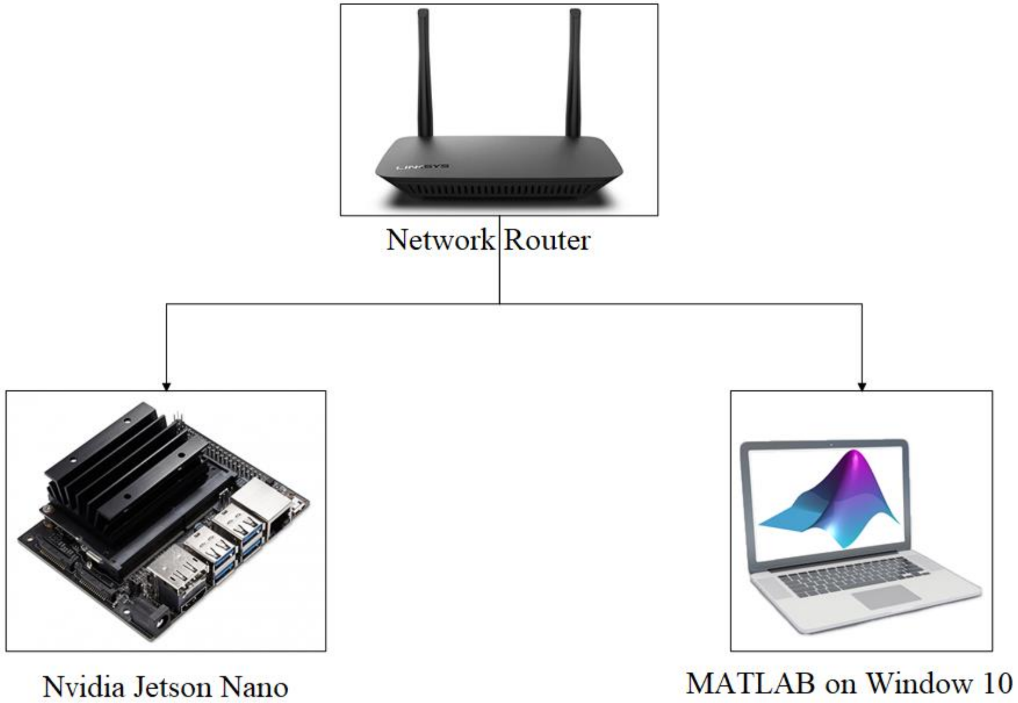
$$d = [d_1, d_2, \dots \dots \dots, d_{360}]$$

Coordinate Frame



- Base frame is the base coordinate of robot
- Laser frame is the base coordinate of Lidar that attached to base frame
- Odom frame is the path of robot
- Map frame is the Occupancy Map Coordinate

Simulation



On Nvidia Jetson Nano
Ubuntu ROS melodic

↓

Initiate ROS master
Command <roscore>

↓

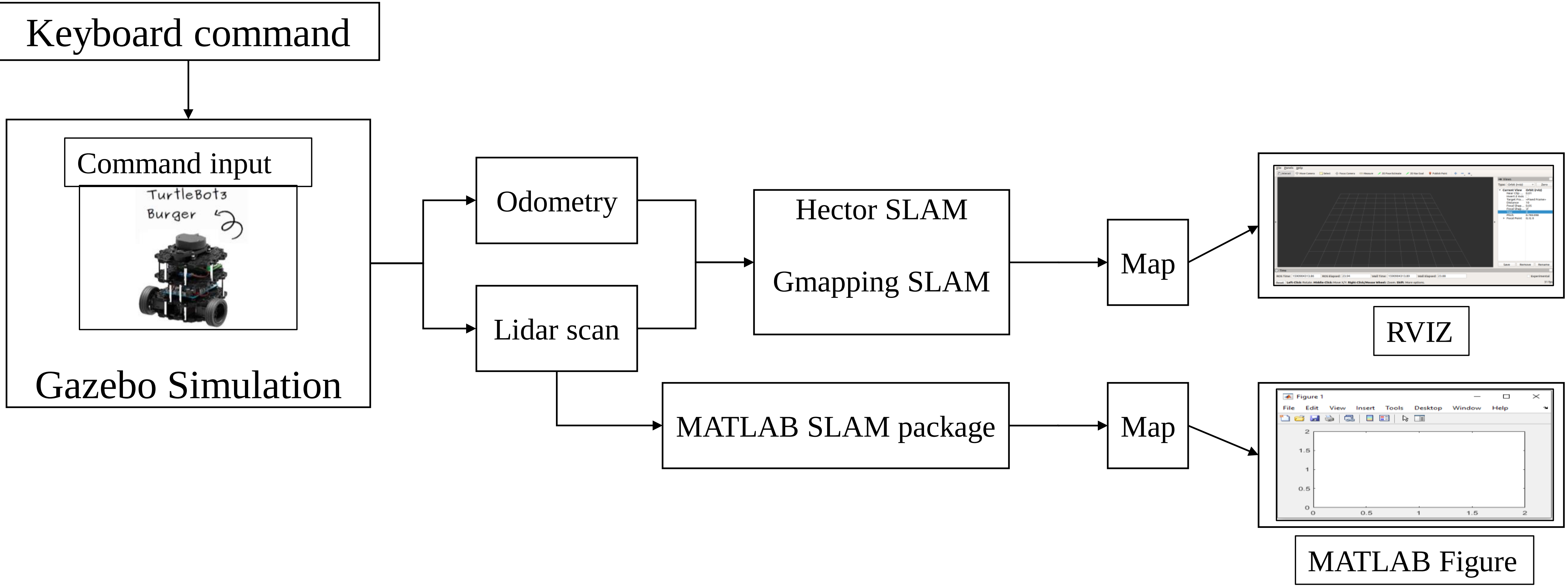
Open Gazebo simulation software
with WMR (Turtlebot3 burger) and environment

On MATLAB
Window

↓

Initiate ROS subscriber

Simulation


Keyboard command
Command input
TurtleBot3
Burger
Gazebo Simulation
Odometry
Lidar scan
Hector SLAM
Gmapping SLAM
MATLAB SLAM package
Map
Map
RVIZ
MATLAB Figure

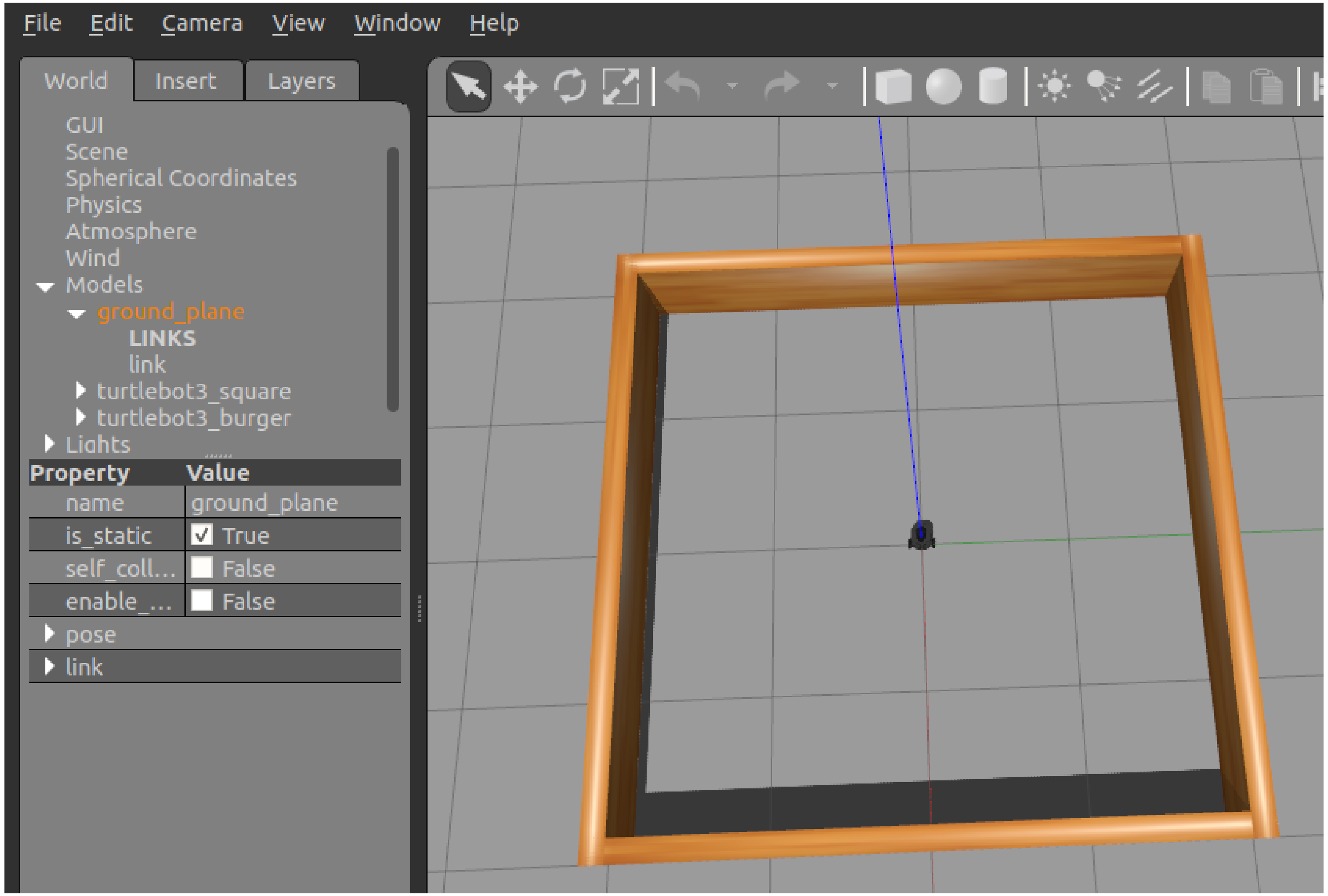
Result

Simulation is done on 5 gazebo map:

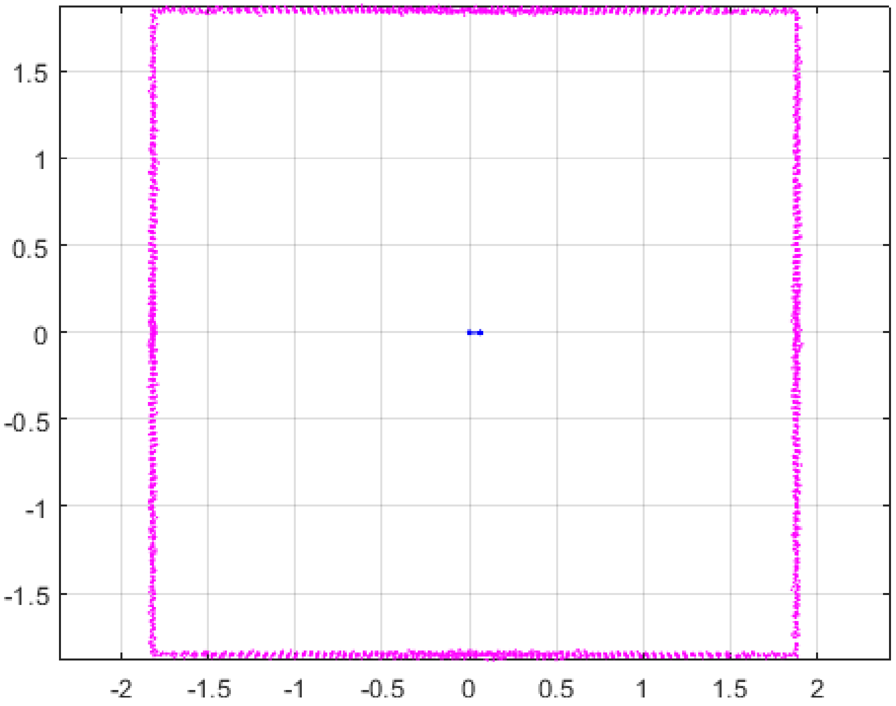
- Stage 1
- Stage 2
- Stage 3
- Gazebo world
- House

Using Hector SLAM, Gmapping SLAM, MATLAB SLAM

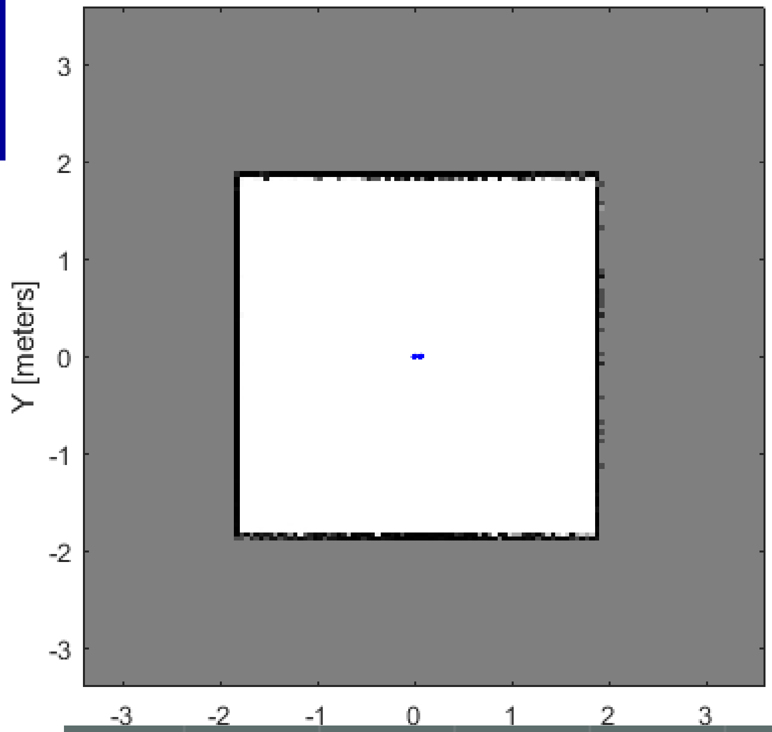
Result and Discussion



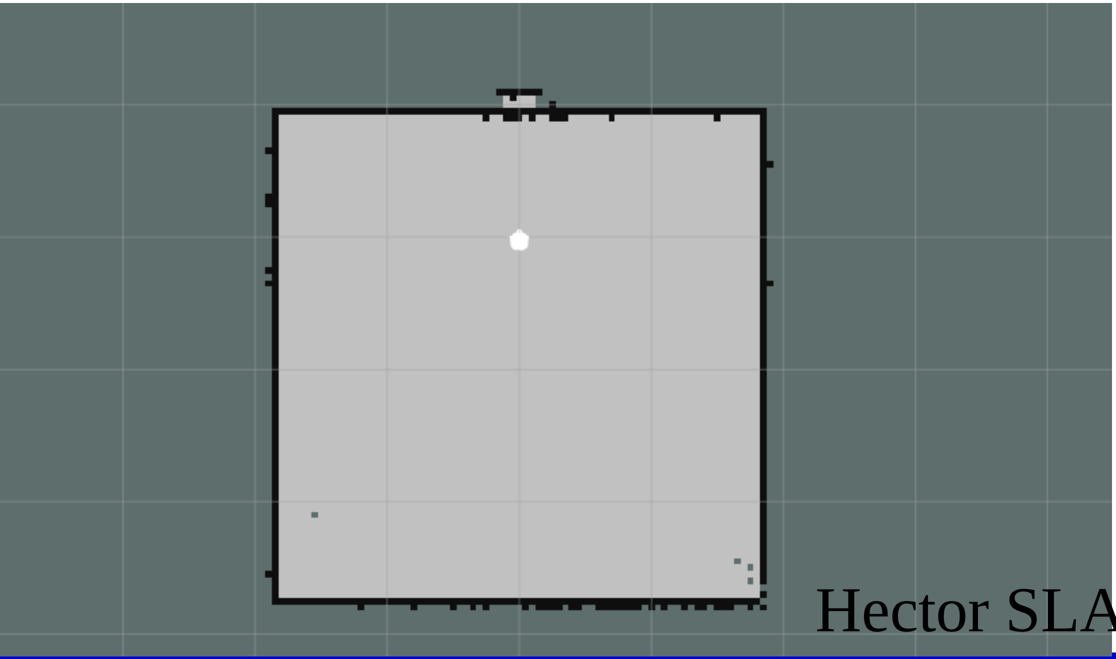
Gazebo stage 1



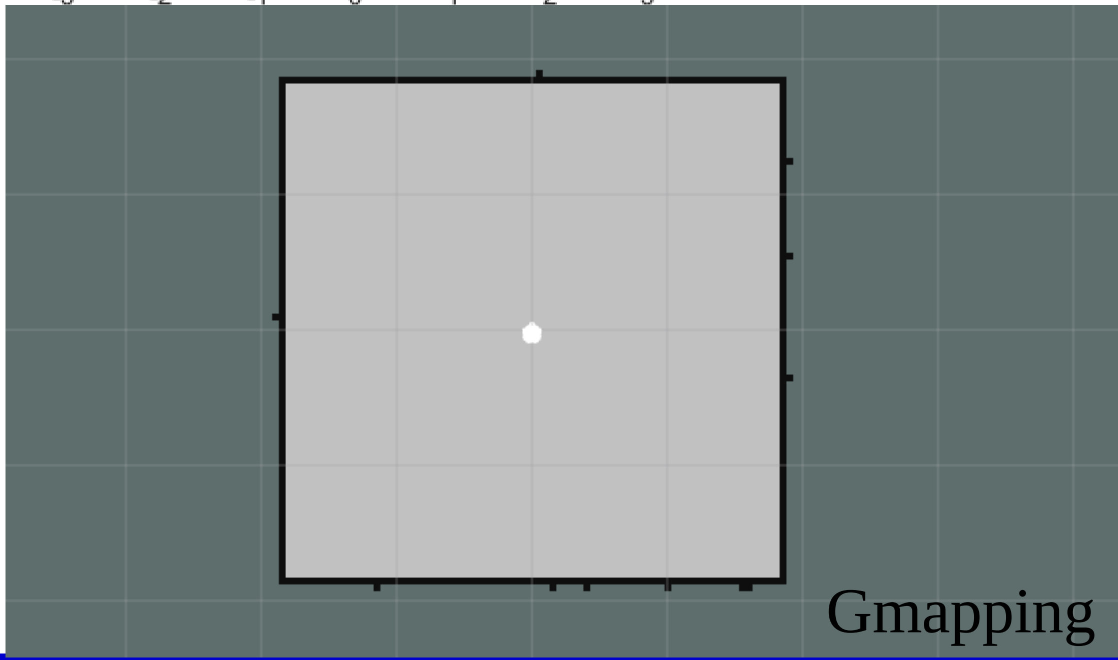
Pose Graph



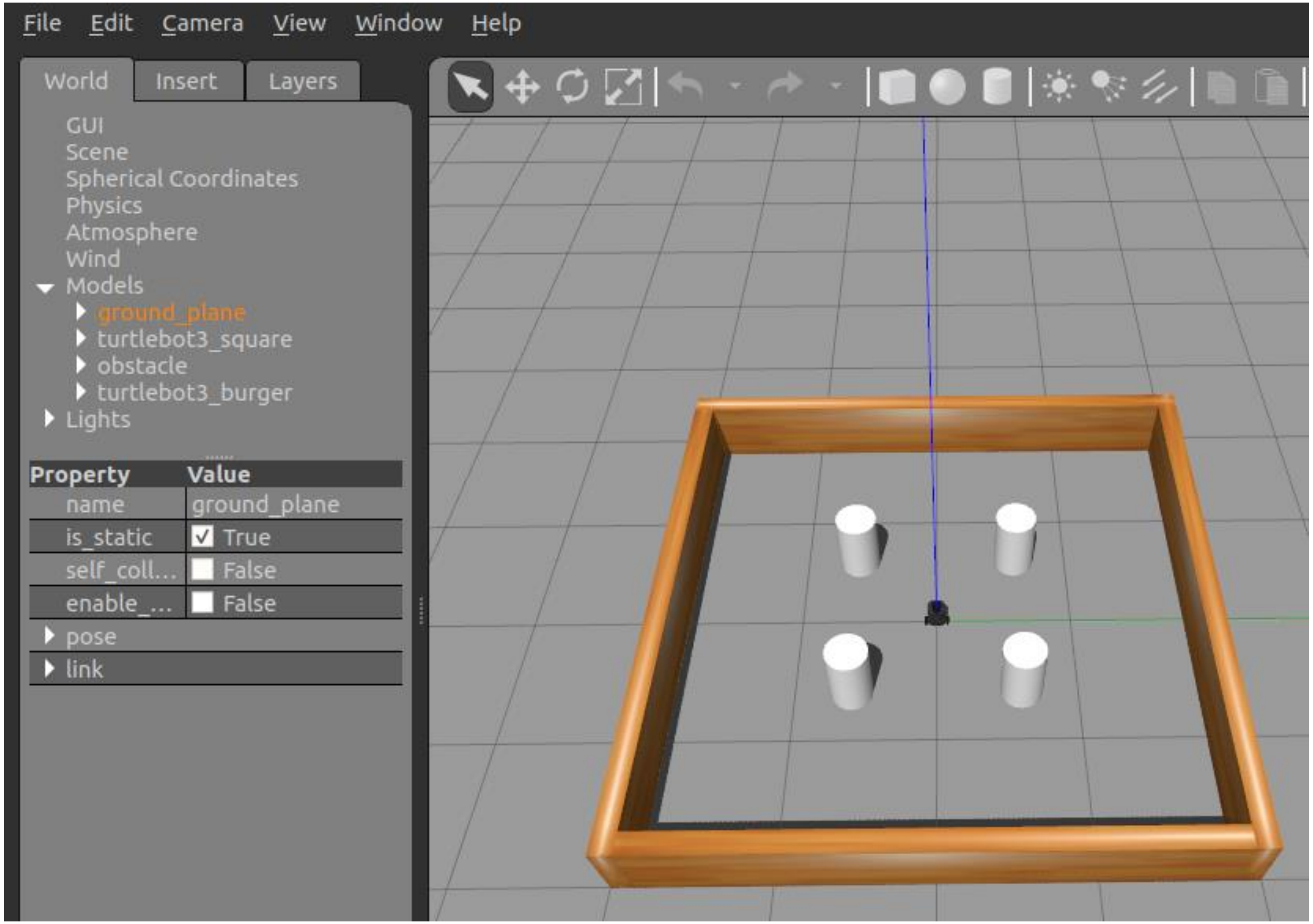
MATLAB



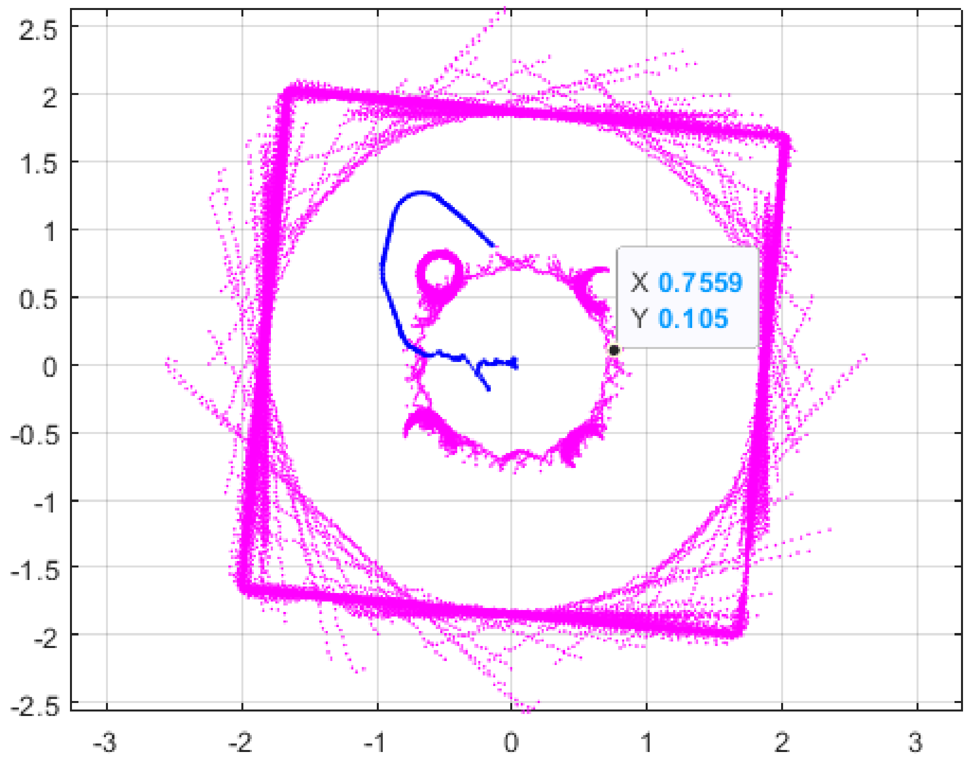
Hector SLAM



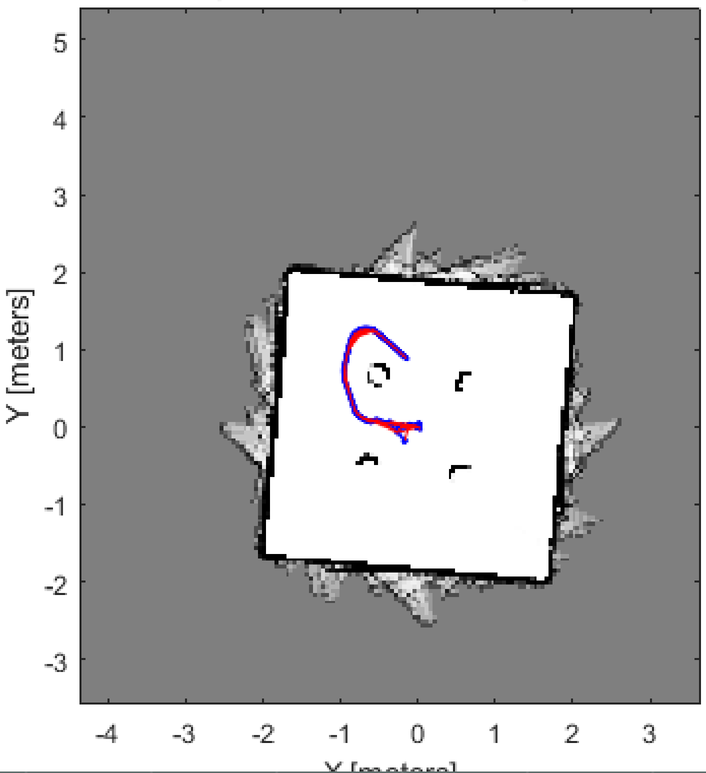
Gmapping



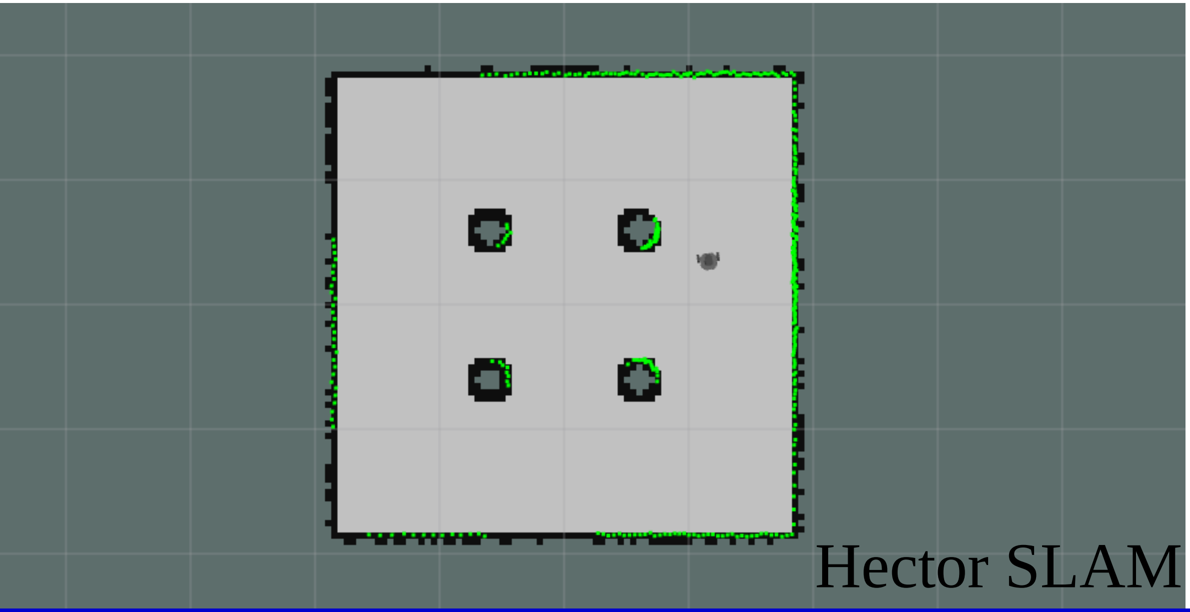
Gazebo stage 2



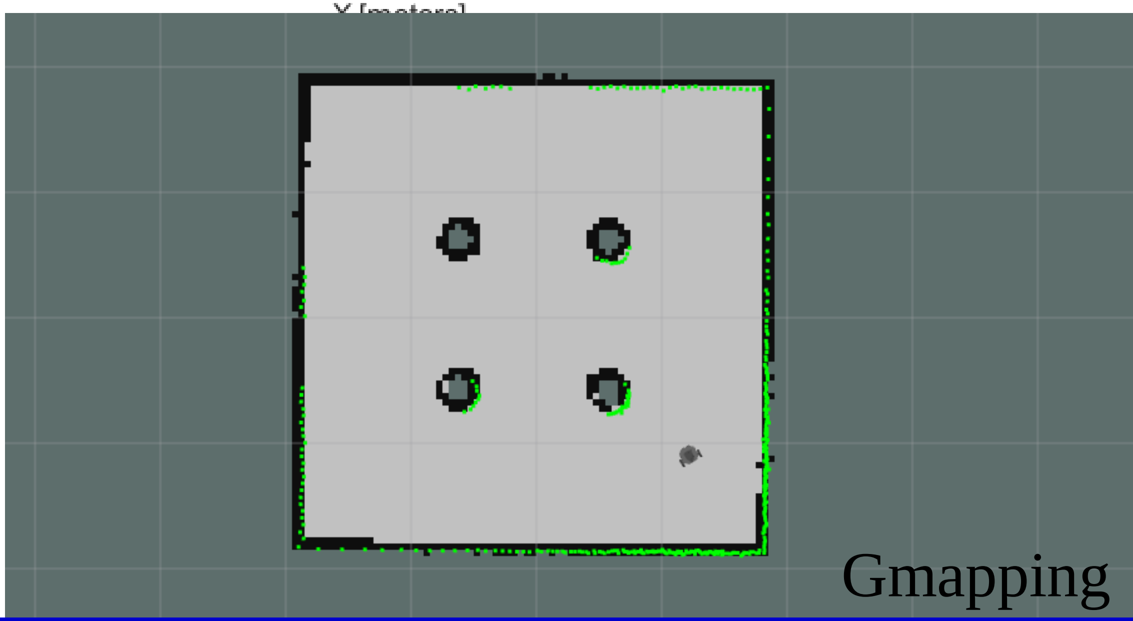
Pose Graph



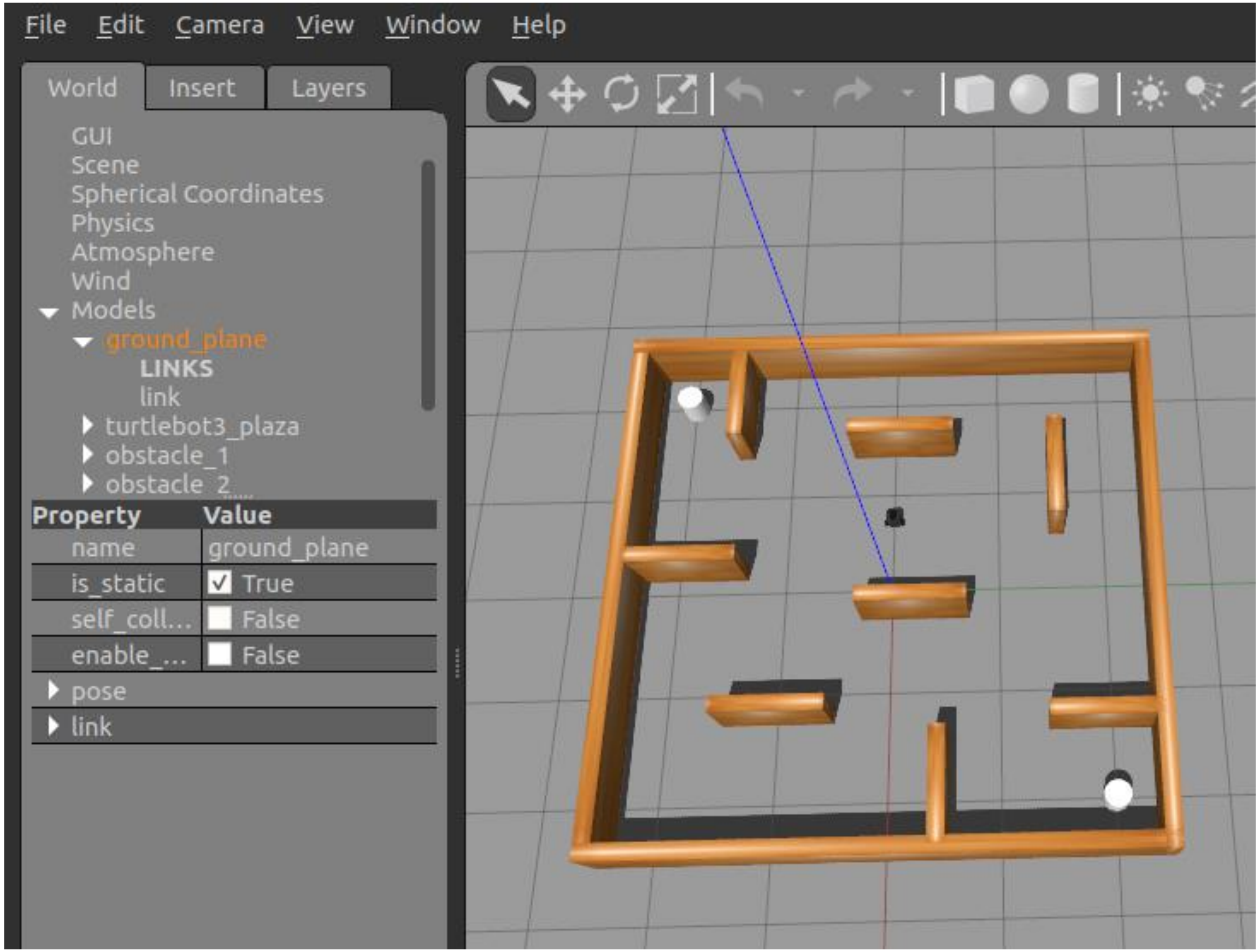
MATLAB



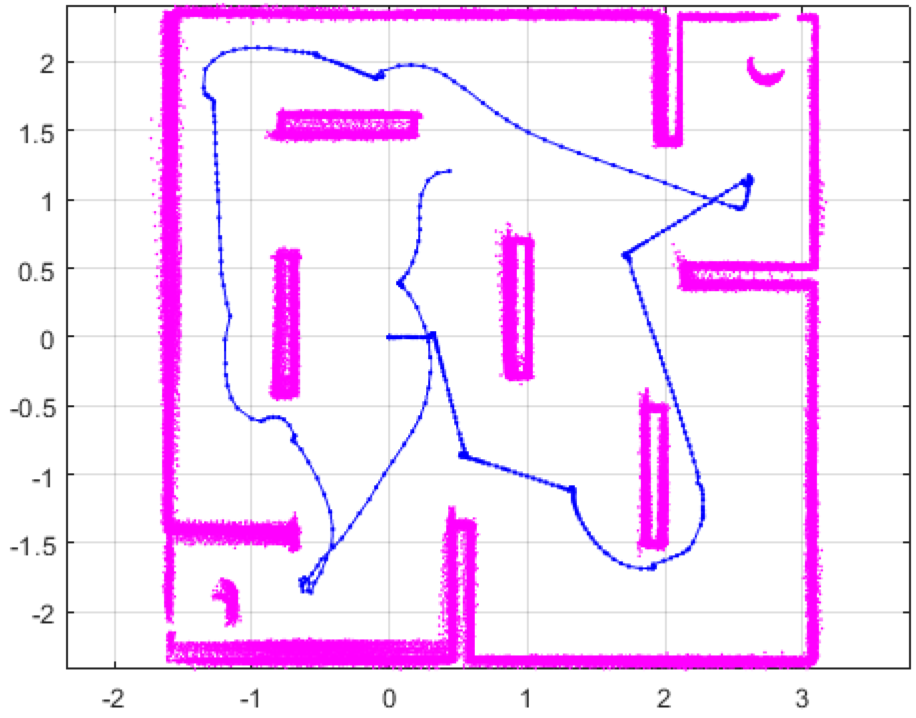
Hector SLAM



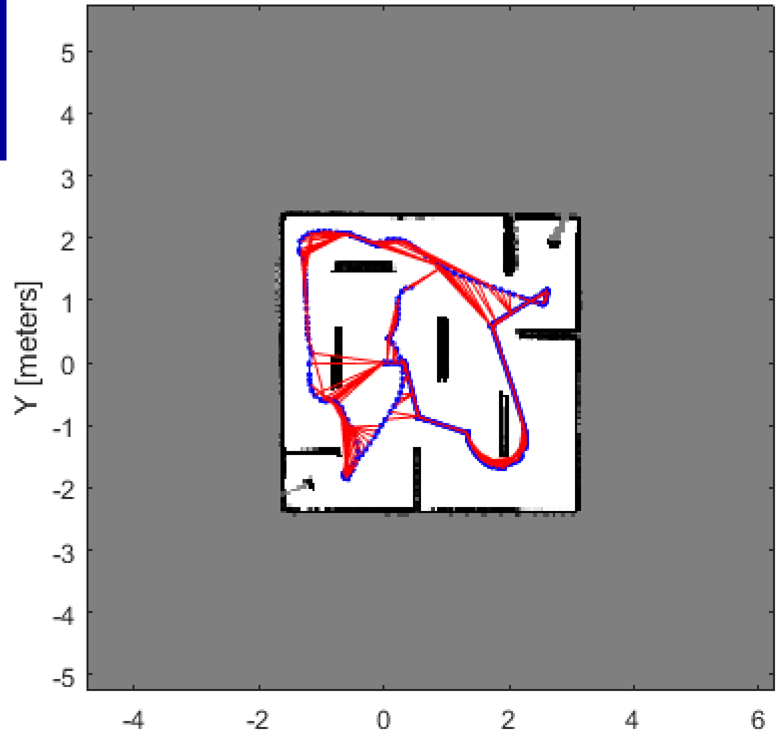
Gmapping



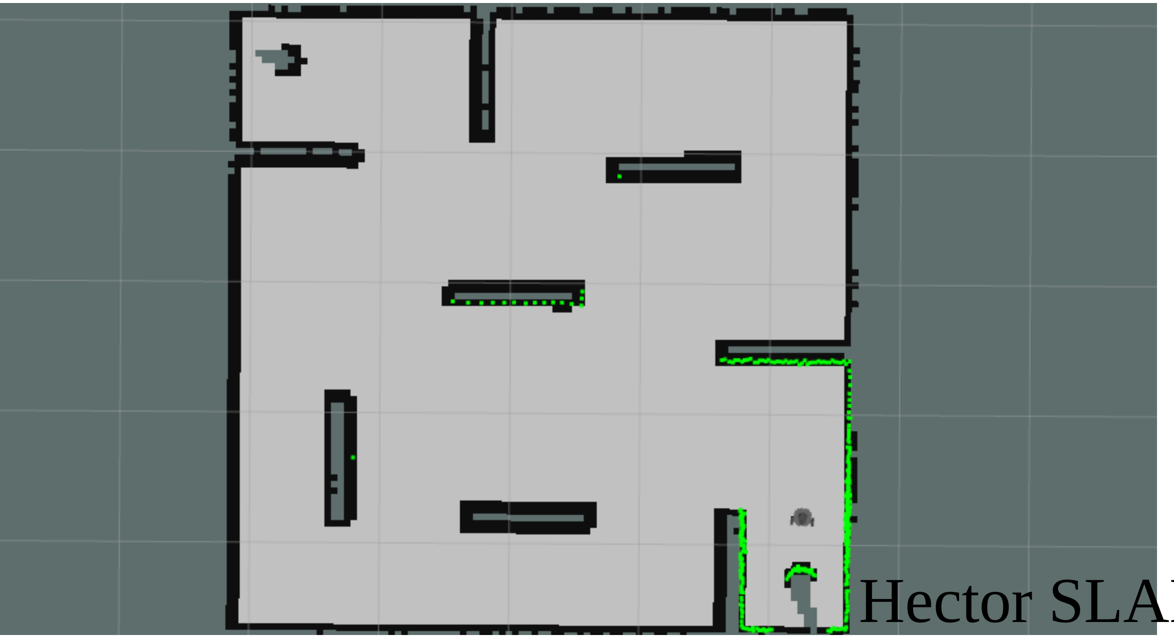
Gazebo stage 3



Pose Graph

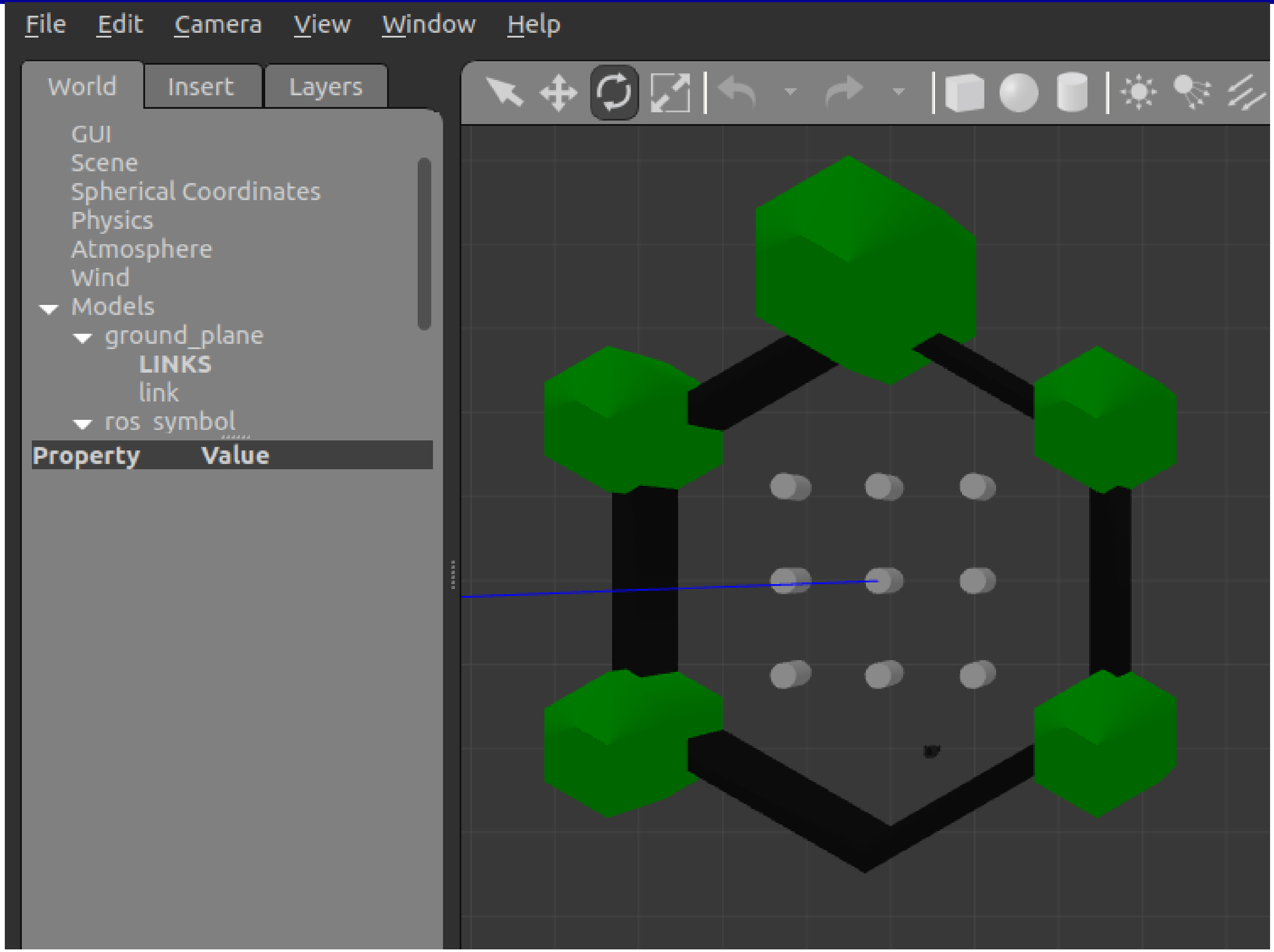


MATLAB

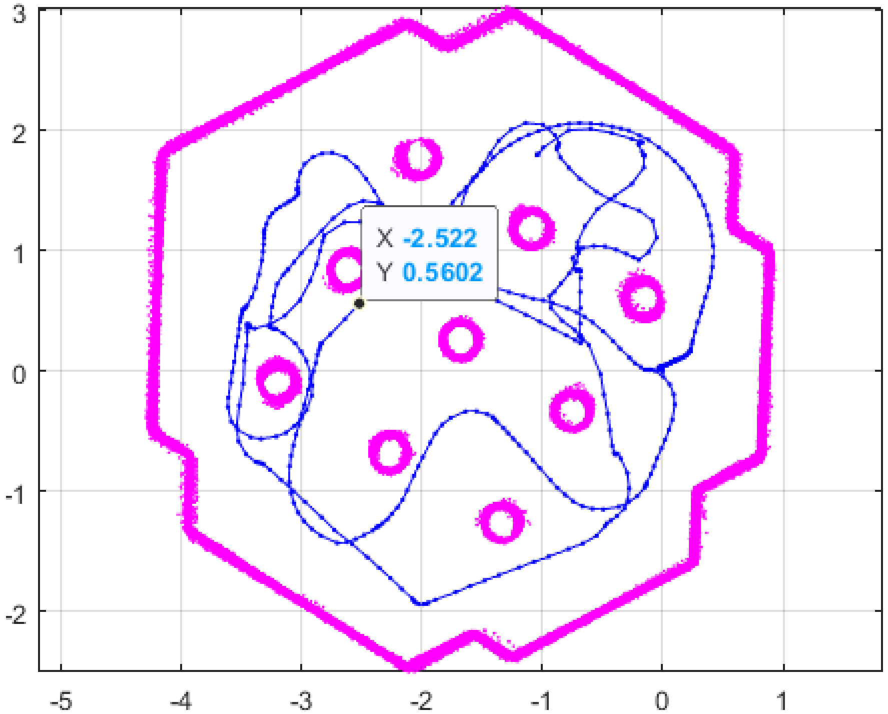


Hector SLAM

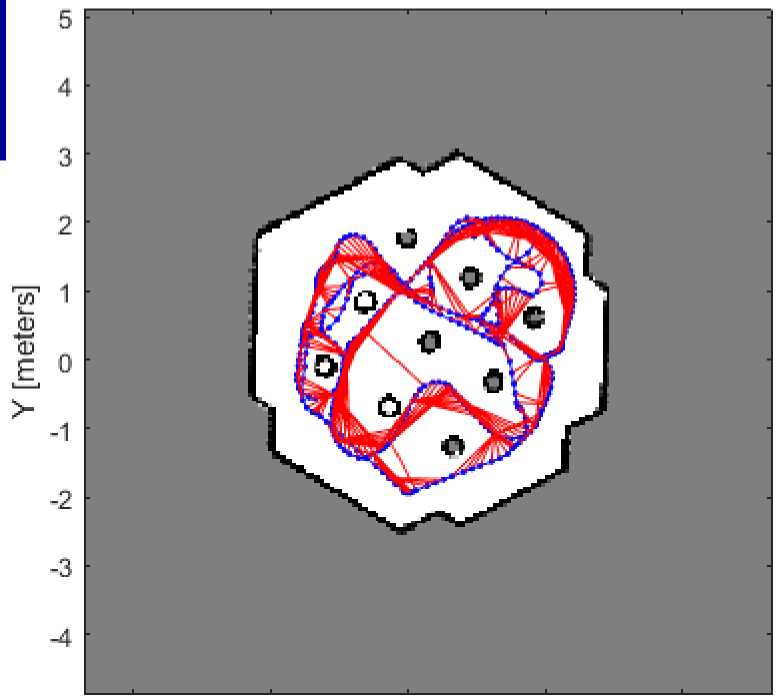
Gmapping



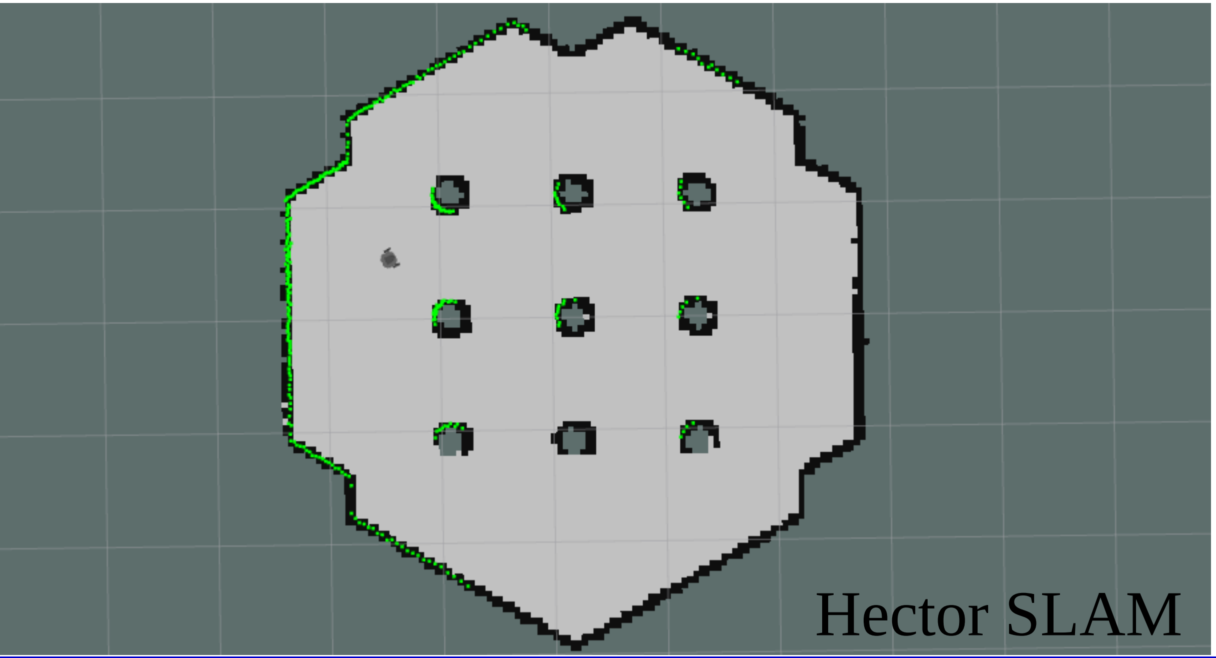
Gazebo world



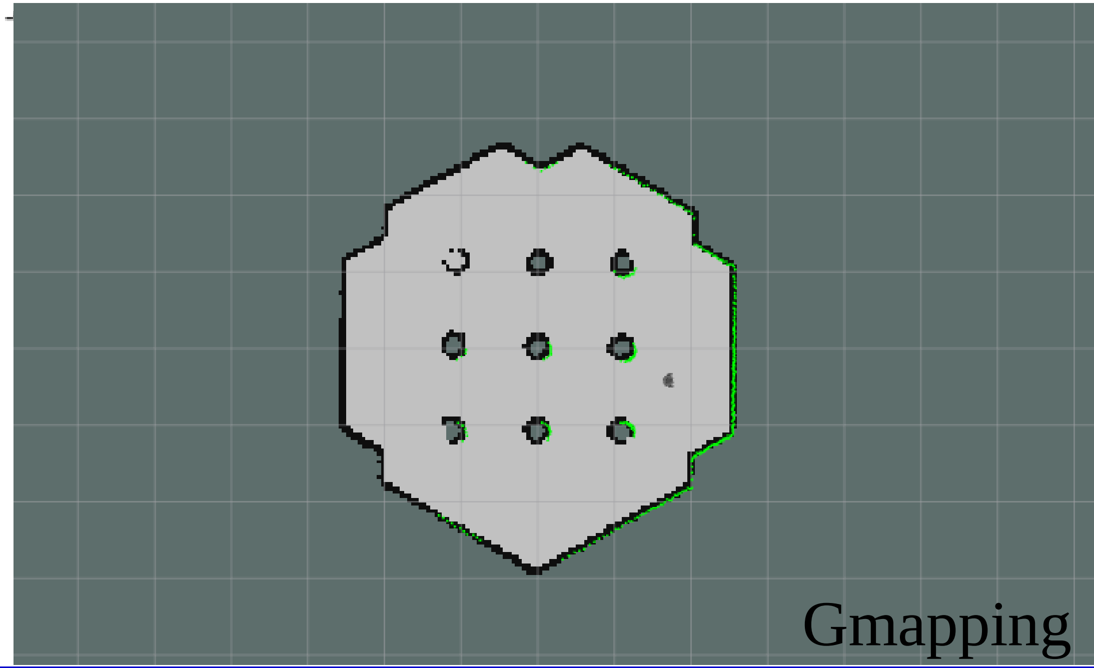
Pose Graph



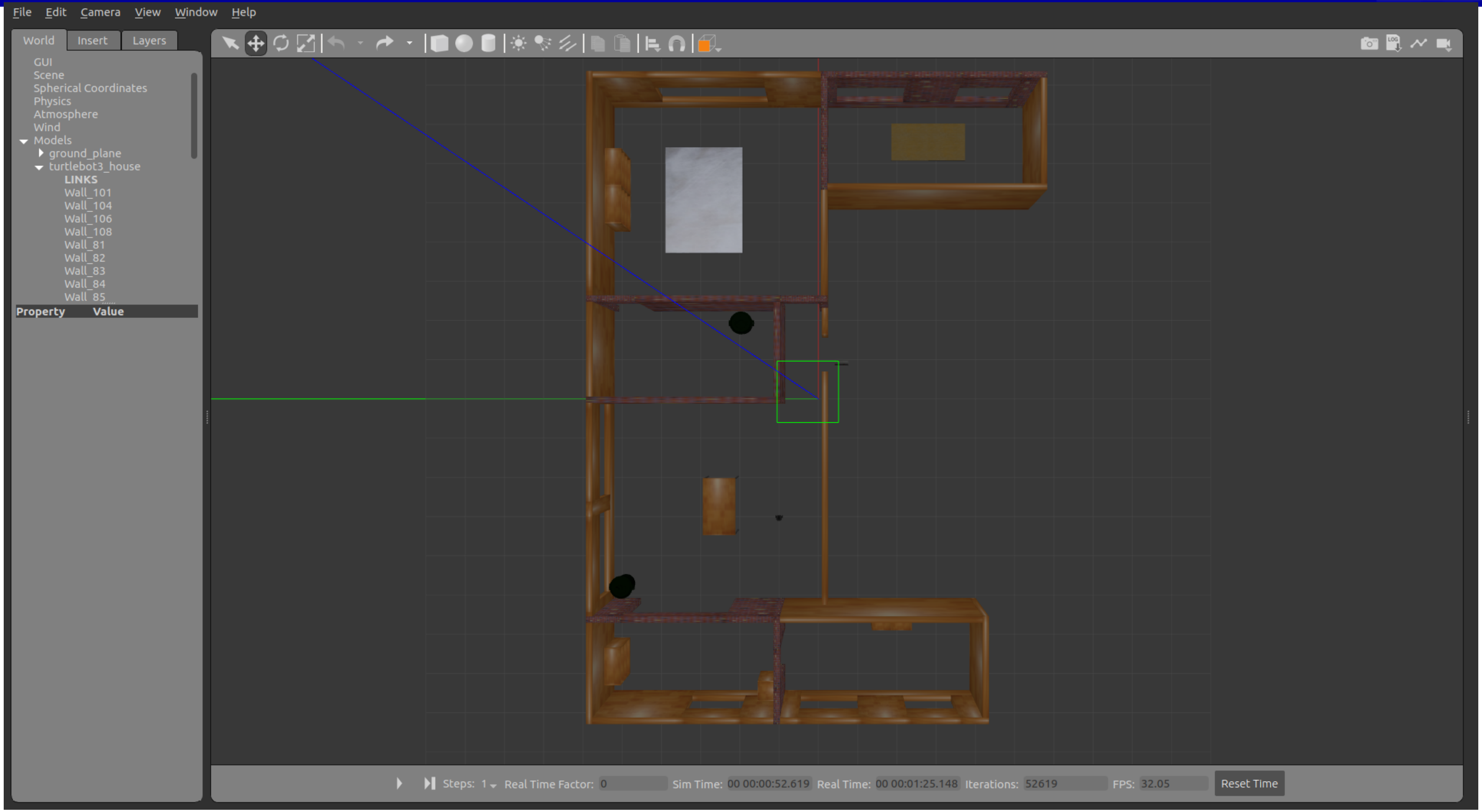
MATLAB



Hector SLAM

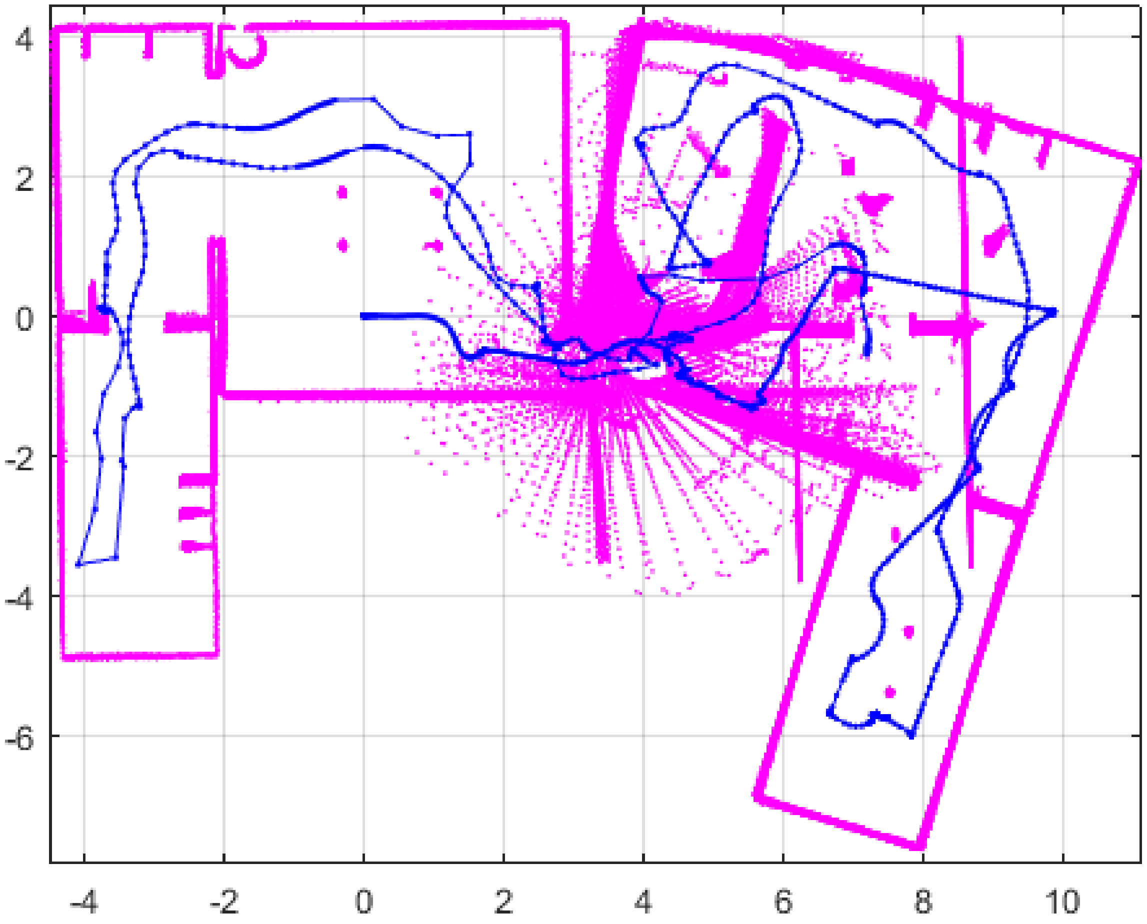


Gmapping

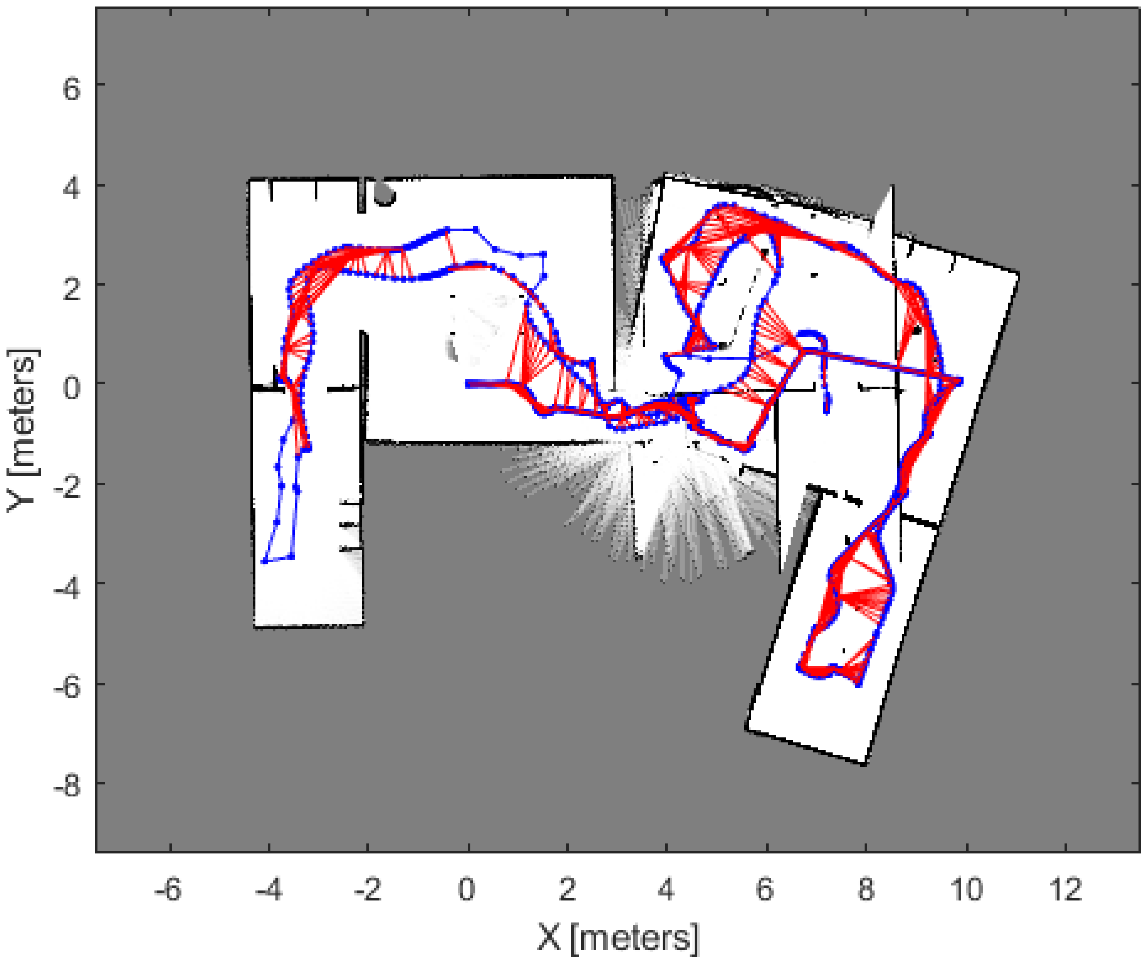


Gazebo house

Pose Graph



MATLAB



Gmapping

Hector SLAM

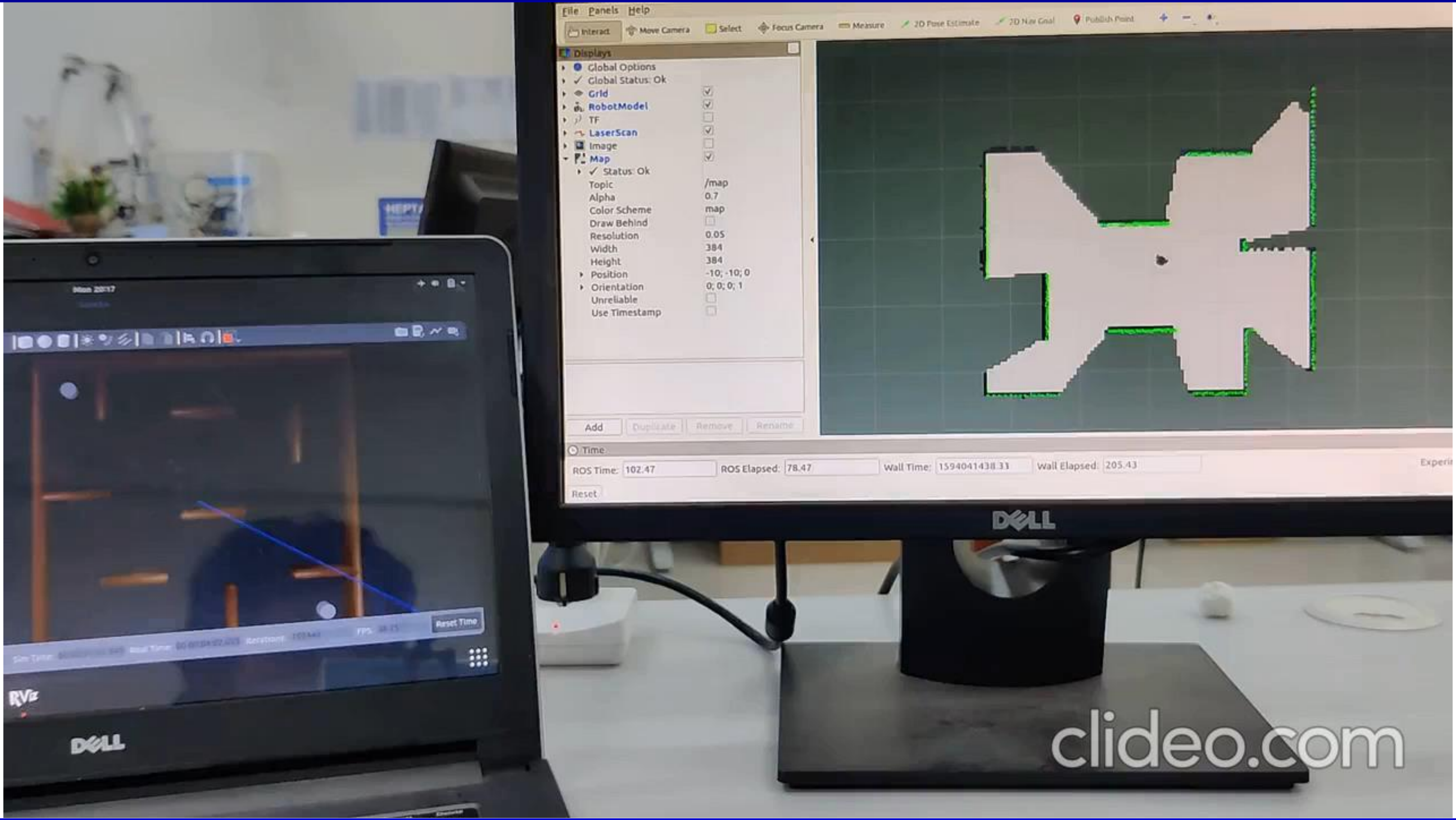
Conclusion and Recommendation

Conclusion

- SLAM algorithm is presented
- ROS, Hector SLAM, Gmapping SLAM, MATLAB SLAM are presented
- WMR, Sensor, and Environment are simulated using Gazebo simulation
- Occupancy grid map is created via the incoming data from simulation

Recommendation Future work

- Implement in real world condition with the real data from the WMR and sensor.
- Using sensor data filter for corrupted sensor data to correctly create an acceptable map
- Implement for real purpose using, such as path planning, autonomous driving and dynamic environment navigation
- Sensor fusion


File Panels Help
Interact
Move Camera
Select
Focus Camera
Measure
2D Pose Estimate
2D Nav Goal
Publish Point
Displays
Global Options
Global Status: Ok
Grid
RobotModel
TF
LaserScan
Image
Map
Status: Ok
Topic /map
Alpha 0.7
Color Scheme map
Draw Behind
Resolution 0.05
Width 384
Height 384
Position -10; -10; 0
Orientation 0; 0; 0; 1
Unreliable
Use Timestamp
Add
Duplicate
Remove
Rename
Time
ROS Time: 102.47
ROS Elapsed: 78.47
Wall Time: 1594041438.33
Wall Elapsed: 205.43
Reset
DELL
Reset Time
RViz
DELL
clideo.com

THANK YOU