A53238541 Chun Hu

1. Introduction

Simultaneous localization and mapping (SLAM) is an important technique using in robotics to localize robot's position in unknow environment and build a map of the environment at the same time. While doing SLAM, we need to use filter to keep track of the robot's position and orientation. In addition, we can use the filter to help us merge different sensor data and improve localization.

In this project, we use robot's odometry and lidar data to perform SLAM and use particle filter to keep track of the robot's position and orientation in the world.

2. Problem Formulation

Given a series of sensor observation from timestamp 1 to timestamp t $O_{1:t}$, we want to compute the robot's state x_t and the map m_t.

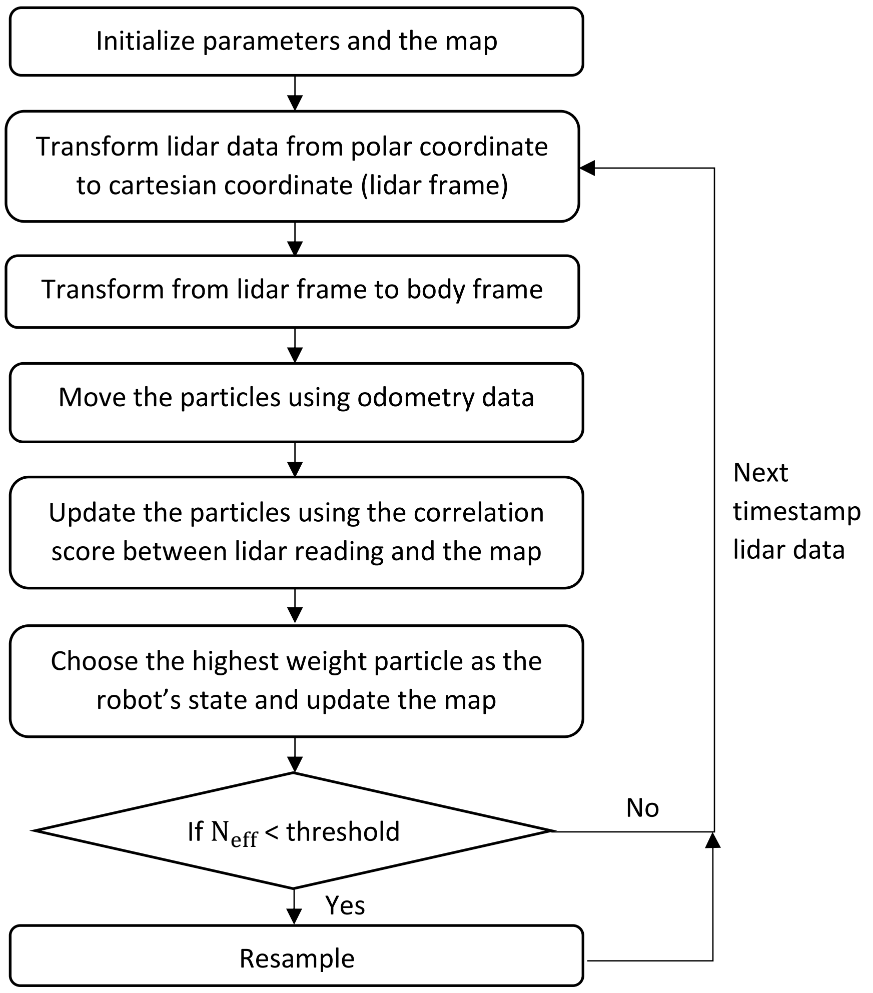
The probability we want to compute is: $P(x_t, m_t|O_{1:t})$

By applying Bayes' rule:

$$P(x_t|O_{1:t}, m_t) = \sum_{m_{t-1}} P(O_t|x_t, m_t) \sum_{x_{t-1}} P(x_t|x_{t-1}) P(x_{t-1}|m_t, O_{1:t-1})$$

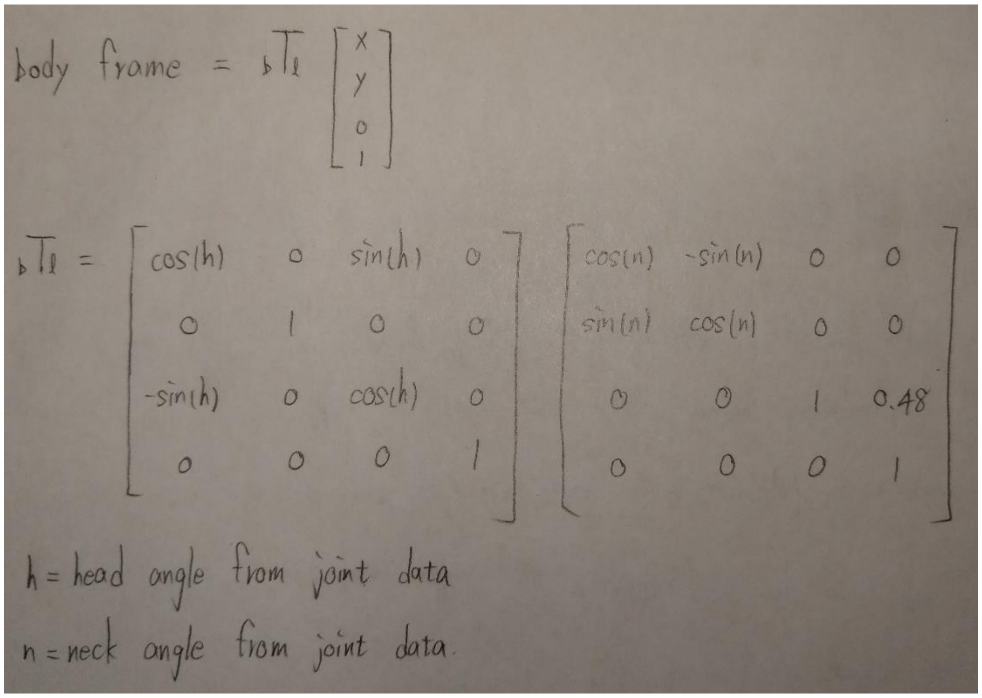
$$P(m_t|O_{1:t}, x_t) = \sum_{x_t} \sum_{m_t} P(m_t|x_t, m_{t-1}, O_t) P(m_{t-1}, x_t|m_{t-1}, O_{1:t-1})$$

3. Technical Approach


Initialize parameters and the map
Transform lidar data from polar coordinate to cartesian coordinate (lidar frame)
Transform from lidar frame to body frame
Move the particles using odometry data
Update the particles using the correlation score between lidar reading and the map
Choose the highest weight particle as the robot's state and update the map
If N_eff < threshold
No
Yes
Resample
Next timestamp lidar data

A53238541 Chun Hu

(1) Initialize parameters and the map
 Decide the map's size, resolution and initialize every grid to be zero
 Set particle number = 100
 For i = 1~100
  Set x^i = [0;0;0], $\alpha^i = \frac{1}{100}$, $\beta^i = \log(\alpha^i)$
(2) Polar coordinate to cartesian coordinate
 [x, y] = [r*cos(θ), r*sin(θ)]
 r = lidar data (Distance for each bin)
 θ = -135 ~135 (Information from the spec)
(3) Lidar frame to body frame
 a. Find the head angle that has closest timestamp to lidar data
  Use a for loop to find the minimum absolute timestamp difference between lidar data and joint data
 b. Use the head angle to transform lidar frame to body frame



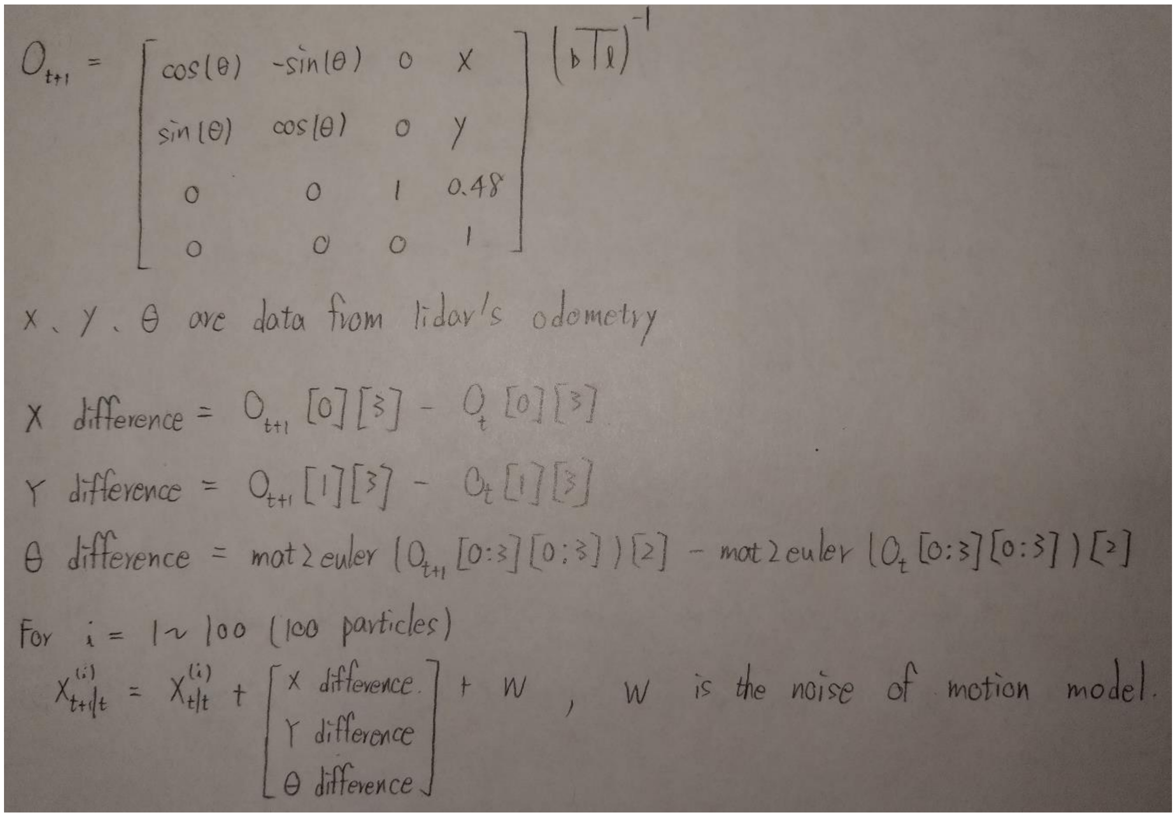
$$\text{body frame} = {}_bT_l \begin{bmatrix} x \\ y \\ 0 \\ 1 \end{bmatrix}$$

$${}_bT_l = \begin{bmatrix} \cos(h) & 0 & \sin(h) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(h) & 0 & \cos(h) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(n) & -\sin(n) & 0 & 0 \\ \sin(n) & \cos(n) & 0 & 0 \\ 0 & 0 & 1 & 0.48 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

h = head angle from joint data
n = neck angle from joint data.

A53238541   Chun Hu

(4) Move the particles using odometry data (Predict step for the particle filter)



$$O_{t+1} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 & x \\ \sin(\theta) & \cos(\theta) & 0 & y \\ 0 & 0 & 1 & 0.48 \\ 0 & 0 & 0 & 1 \end{bmatrix} ({}_{b}T_{l})^{-1}$$

x、y、θ are data from lidar's odometry

X difference = $O_{t+1}[0][3] - O_t[0][3]$

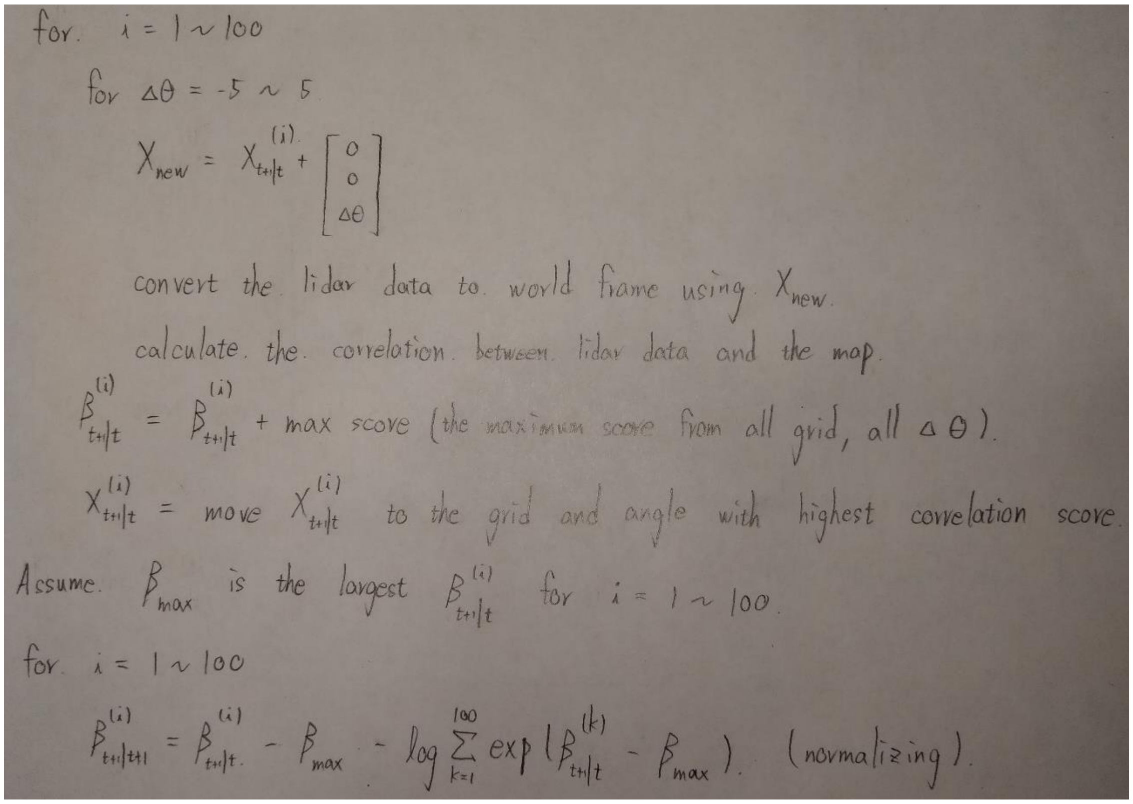
Y difference = $O_{t+1}[1][3] - O_t[1][3]$

θ difference = $\mathrm{mat2euler}(O_{t+1}[0:3][0:3])[2] - \mathrm{mat2euler}(O_t[0:3][0:3])[2]$

For $i = 1 \sim 100$ (100 particles)

$$X^{(i)}_{t+1|t} = X^{(i)}_{t|t} + \begin{bmatrix} X \text{ difference} \\ Y \text{ difference} \\ \theta \text{ difference} \end{bmatrix} + W \quad , \quad W \text{ is the noise of motion model.}$$

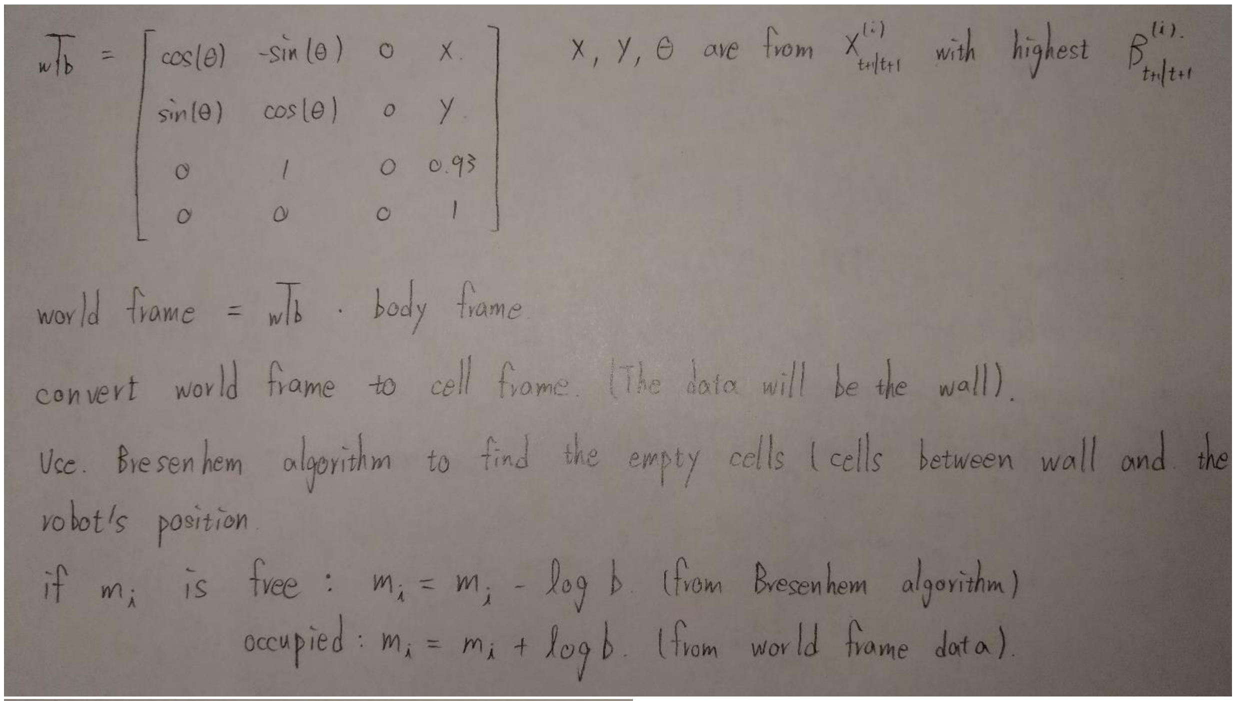
(5) Update the particles using the correlation score between lidar reading and the map (Update step for the particle filter)



```
for i = 1 ~ 100
    for Δθ = -5 ~ 5
        $X_{new} = X^{(i)}_{t+1|t} + \begin{bmatrix} 0 \\ 0 \\ \Delta\theta \end{bmatrix}$
        convert the lidar data to world frame using $X_{new}$.
        calculate the correlation between lidar data and the map.
    $\beta^{(i)}_{t+1|t} = \beta^{(i)}_{t+1|t}$ + max score (the maximum score from all grid, all $\Delta\theta$).
    $X^{(i)}_{t+1|t}$ = move $X^{(i)}_{t+1|t}$ to the grid and angle with highest correlation score.
Assume $\beta_{max}$ is the largest $\beta^{(i)}_{t+1|t}$ for $i = 1 \sim 100$.
for i = 1 ~ 100
    $\beta^{(i)}_{t+1|t+1} = \beta^{(i)}_{t+1|t} - \beta_{max} - \log \sum_{k=1}^{100} \exp(\beta^{(k)}_{t+1|t} - \beta_{max})$.  (normalizing).
```

A53238541 Chun Hu

(6) Choose the highest weight particle as the robot’s state and update the map



$$_{w}T_{b} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 & x \\ \sin(\theta) & \cos(\theta) & 0 & y \\ 0 & 1 & 0 & 0.93 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

x, y, θ are from $X^{(i)}_{t+1|t+1}$ with highest $\beta^{(i)}_{t+1|t+1}$.

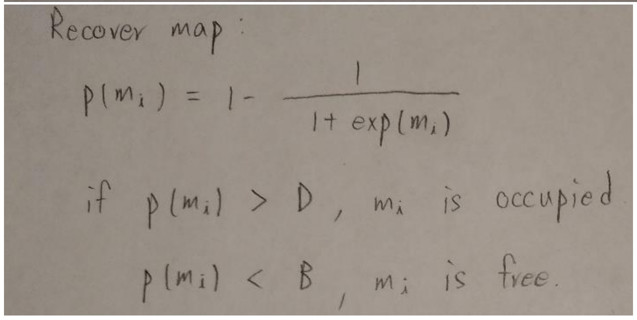
world frame = $_{w}T_{b}$ · body frame

convert world frame to cell frame. (The data will be the wall).

Uce. Bresenhem algorithm to find the empty cells (cells between wall and the robot's position.

if m_i is free : $m_i = m_i - \log b$ (from Bresenhem algorithm)

occupied : $m_i = m_i + \log b$. (from world frame data).



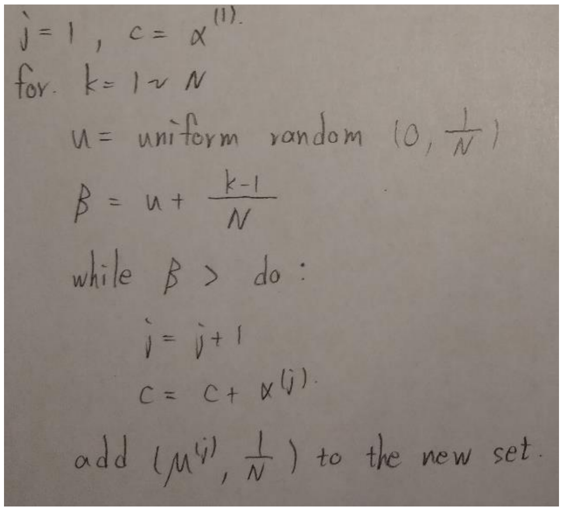
Recover map:

$$p(m_i) = 1 - \frac{1}{1+\exp(m_i)}$$

if $p(m_i) > D$, m_i is occupied.

$p(m_i) < B$, m_i is free.

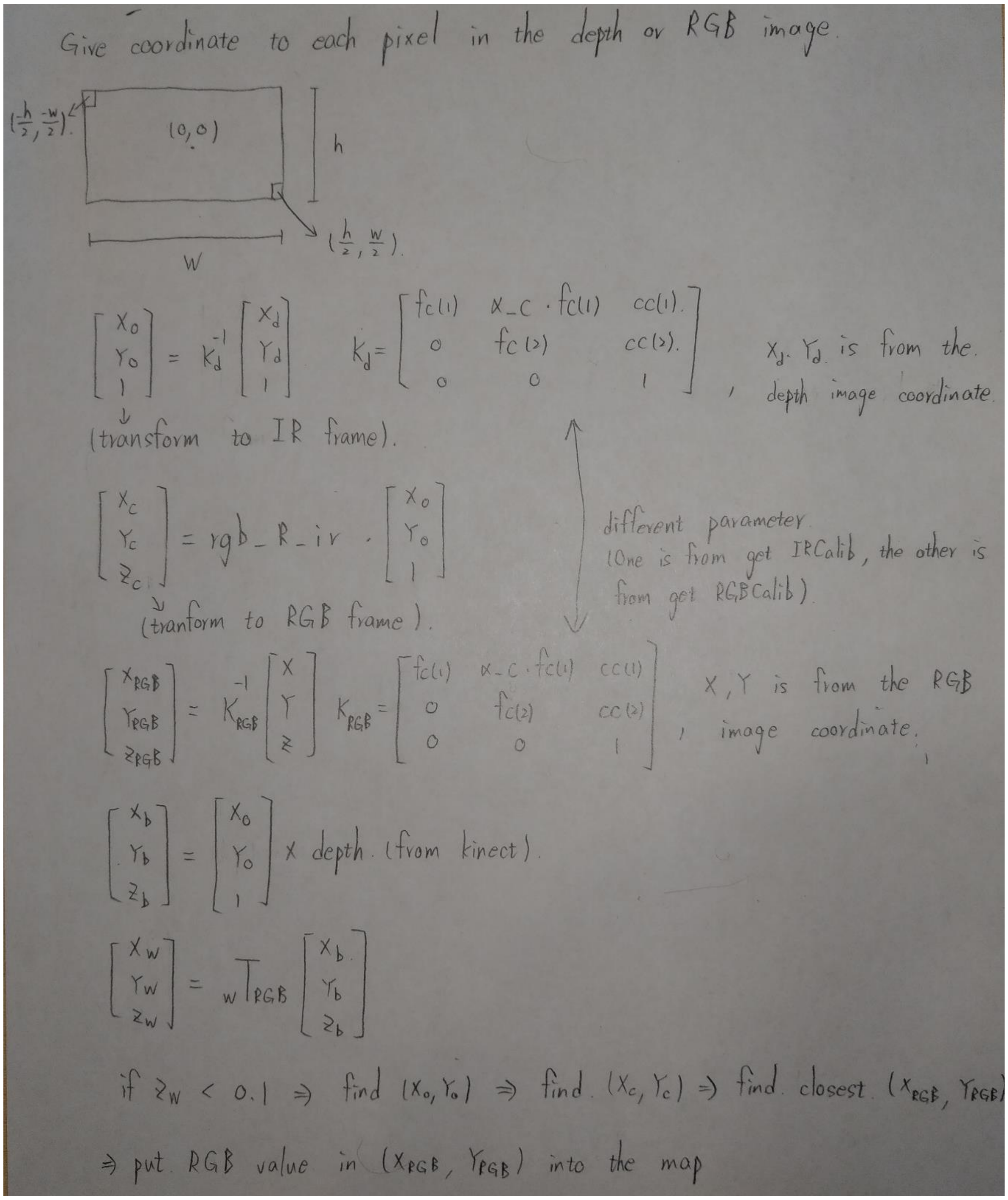
(7) Resample



```
$j = 1$, $c = \alpha^{(1)}$.
for. $k = 1 \sim N$
    $u =$ uniform random $(0, \frac{1}{N})$
    $\beta = u + \frac{k-1}{N}$
    while $\beta >$ do :
        $j = j + 1$
        $c = c + \alpha^{(j)}$.
    add $(\mu^{(j)}, \frac{1}{N})$ to the new set.
```

A53238541 Chun Hu

(8) Texture (Only implement coordinate for each pixel and time synchronize)



Give coordinate to each pixel in the depth or RGB image.

$(\frac{-h}{2}, \frac{-w}{2})$
$(0,0)$
h
W
$(\frac{h}{2}, \frac{w}{2})$

$$\begin{bmatrix} X_0 \\ Y_0 \\ 1 \end{bmatrix} = K_d^{-1} \begin{bmatrix} X_d \\ Y_d \\ 1 \end{bmatrix} \quad K_d = \begin{bmatrix} fc(1) & \alpha\_c \cdot fc(1) & cc(1) \\ 0 & fc(2) & cc(2) \\ 0 & 0 & 1 \end{bmatrix}$$

, X_d, Y_d is from the depth image coordinate.

(transform to IR frame).

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = rgb\_R\_ir \cdot \begin{bmatrix} X_0 \\ Y_0 \\ 1 \end{bmatrix}$$

(tranform to RGB frame).

different parameter. (One is from get IRCalib, the other is from get RGBCalib)

$$\begin{bmatrix} X_{RGB} \\ Y_{RGB} \\ Z_{RGB} \end{bmatrix} = K_{RGB}^{-1} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \quad K_{RGB} = \begin{bmatrix} fc(1) & \alpha\_c \cdot fc(1) & cc(1) \\ 0 & fc(2) & cc(2) \\ 0 & 0 & 1 \end{bmatrix}$$

, X, Y is from the RGB image coordinate.

$$\begin{bmatrix} X_b \\ Y_b \\ Z_b \end{bmatrix} = \begin{bmatrix} X_0 \\ Y_0 \\ 1 \end{bmatrix} \times \text{depth}$$ (from kinect).

$$\begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix} = {}_wT_{RGB} \begin{bmatrix} X_b \\ Y_b \\ Z_b \end{bmatrix}$$

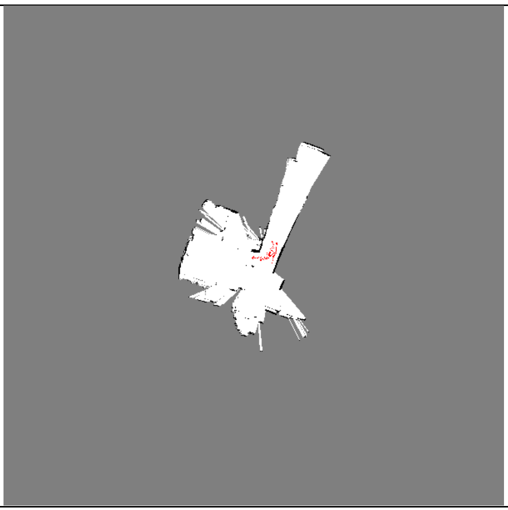
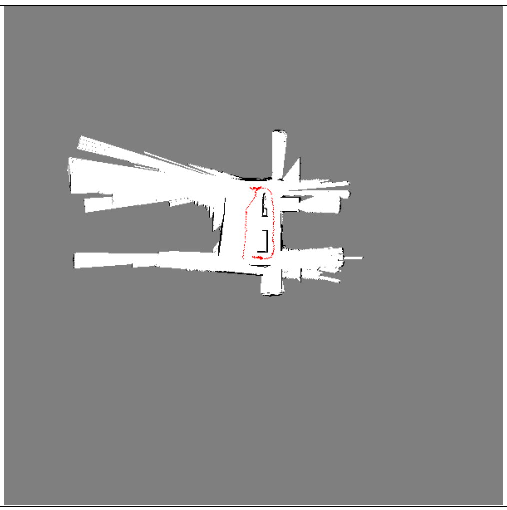
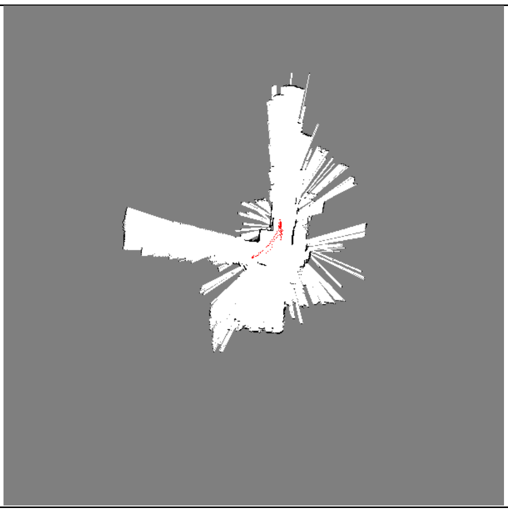
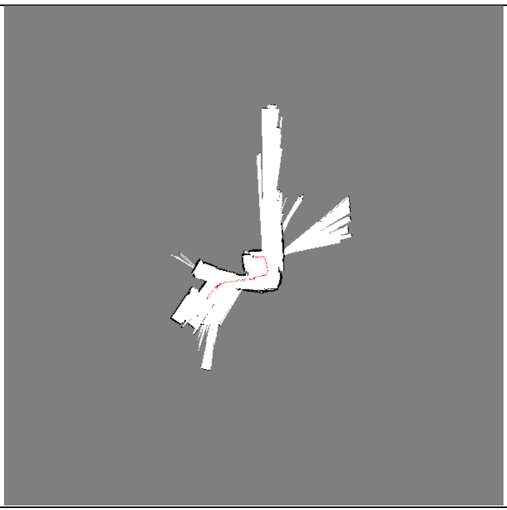
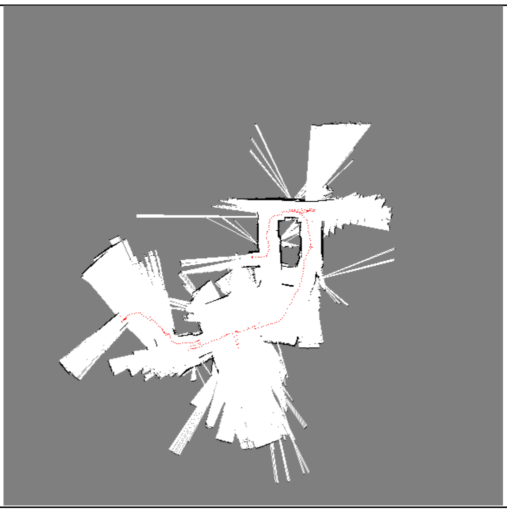
if $Z_w < 0.1$ $\Rightarrow$ find (X_0, Y_0) $\Rightarrow$ find (X_c, Y_c) $\Rightarrow$ find closest (X_{RGB}, Y_{RGB})

$\Rightarrow$ put RGB value in (X_{RGB}, Y_{RGB}) into the map

A53238541 Chun Hu

4. Result

(1) Final picture (Image over time is at the end of the file)

Train data 0	Train data 1
	
Train data 2	**Train data 3**
	
Test data	
	

A53238541 Chun Hu

(2) Discussion

a. Parameter

(a) Motion model noise

The noise is used to estimate the robot's odometry error in the particle filter predict step. When we are not confident in the odometry data, we need to set it in large number. When we are confident in the odometry, we need to set it in small value.
In this project, I have also tried to set the noise to be proportional to the odometry value. (Since many of the sensors have the error proportional to its value). However, it didn't work well in the result.

(b) The yaw angle adjustment during update step

To have a better result, we need to adjust the particle's orientation (yaw angle) in the particle filter update step. If we set the adjusting angle too large, one update step can destroy the map because lidar data will map to the wrong place. Nevertheless, if we set the angle too small, it won't be useful to do the adjustment.

(c) Predict and update time interval

Particle filter's predict and update step will take a lot of time to compute, so we can just do the predict and update step once in a few lidar timestamp. The interval we choose will significantly influence the result. If the interval is too large, there will not be enough map information to do the correlation. Thus, the lidar data might map to the wrong place and cause a bad result.

b. Result

(a) Data set 0

I think my program works well in data set 0, we can see there is only a little position and orientation error in the corridor on the top right of the image.

(b) Data set 1

The program works well before the robot goes to the upper part of the image. I have checked the odometry data, the change of the orientation is large when the robot is in the upper part of the image. Thus, I set larger motion model noise and hoped to solve the problem. However, when the noise is large, the robot will localize at the wrong place in the beginning. So I set the noise to be proportional to the odometry data. But after several tries, the method still not work as well as what parameter I used in the beginning. Therefore, I decided to set the parameter just like what I used in the beginning.

(c) Data set 2

I think the lidar bins in the right part of the image is caused by the robot's orientation error. They should be merge to the wall of the corridor.

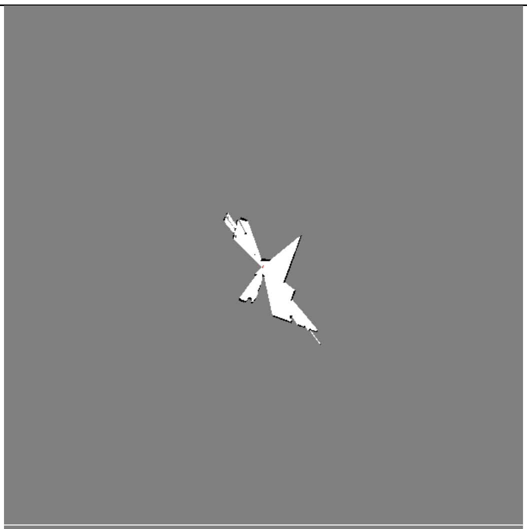
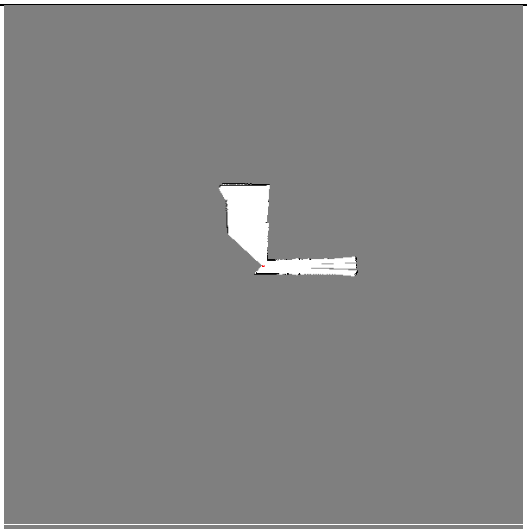
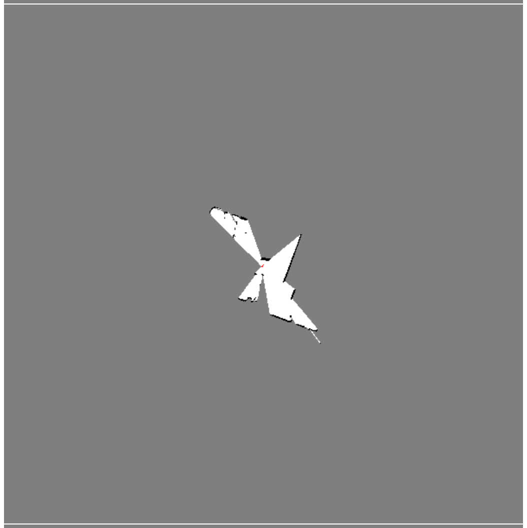
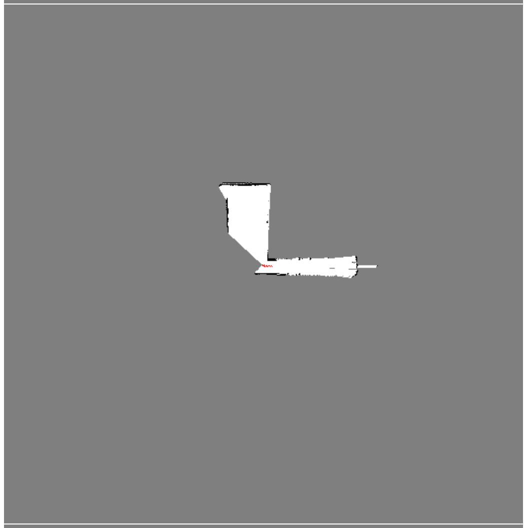
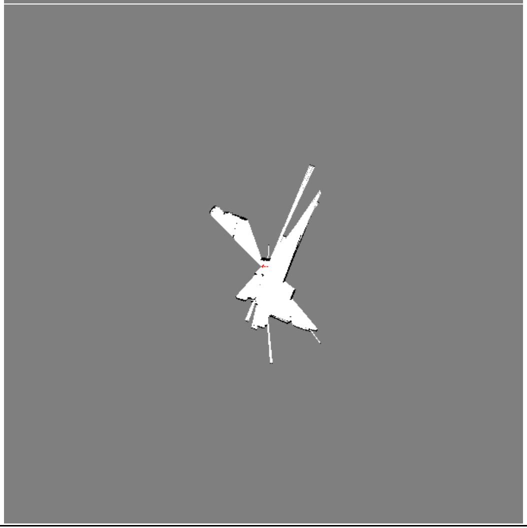
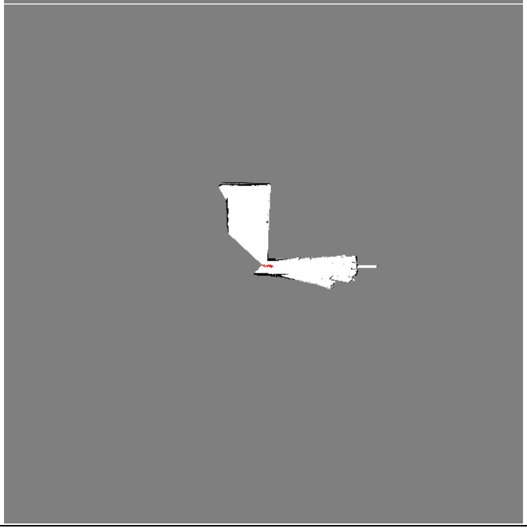
(d) Data set 3

The result is good in data set 3 except for the empty cells on the right of the corridor.

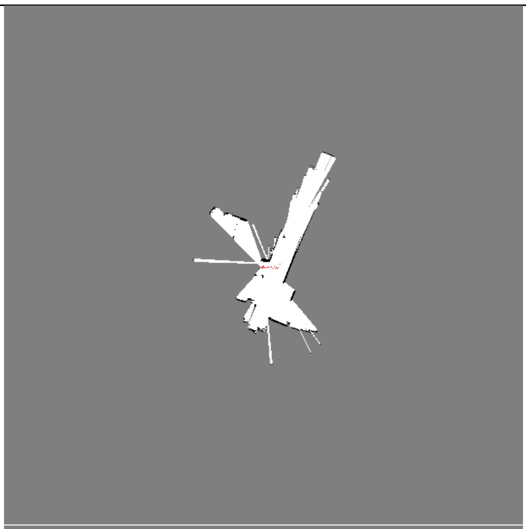
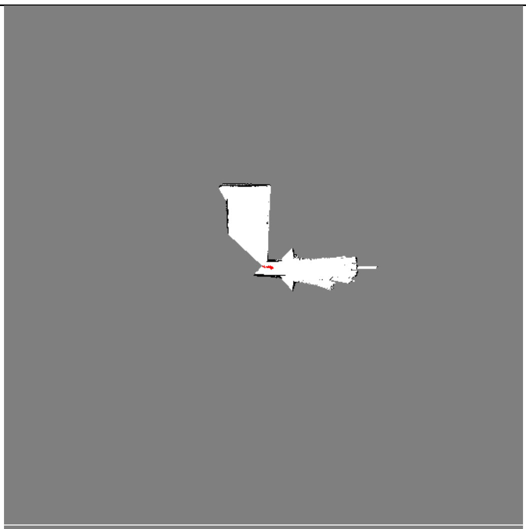
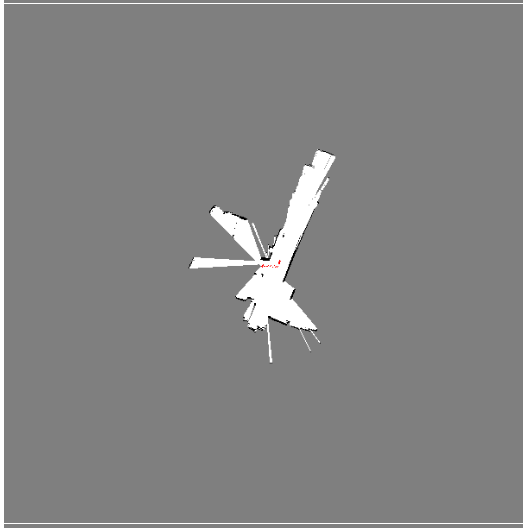
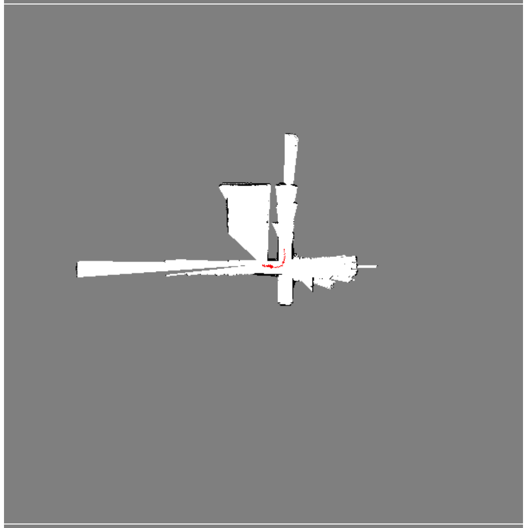
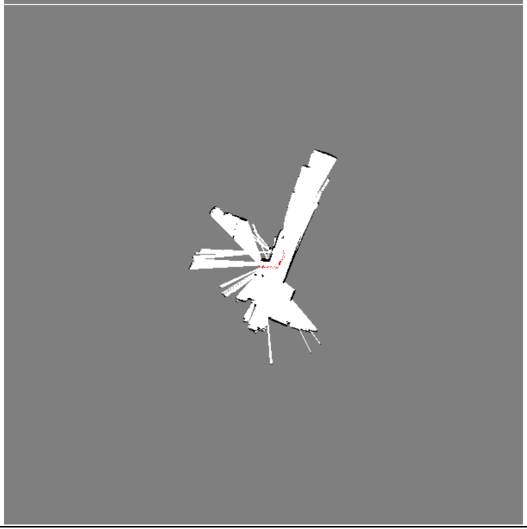
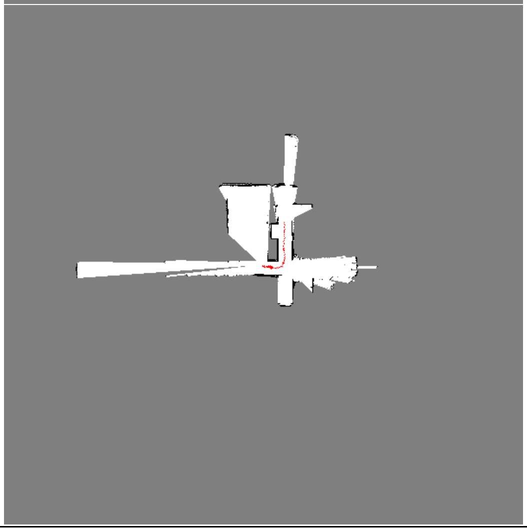
(e) Test set

I guess the corridor in the robot's starting point should connect to the corridor the robot seen at last. In my result, there is about 30-degree difference between the two corridors' orientation. This means that my SLAM program is not good enough for the robot to localize in unknow map with little overlap.

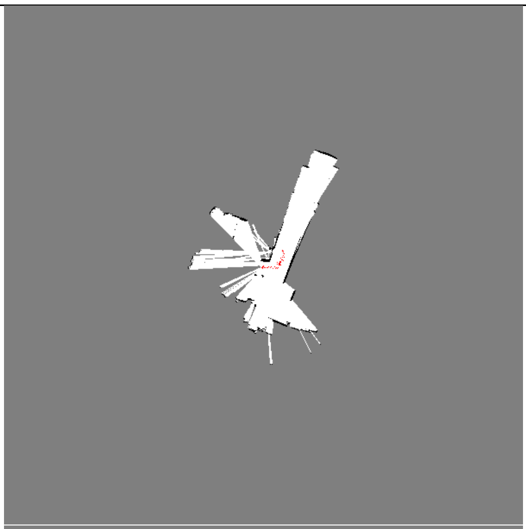
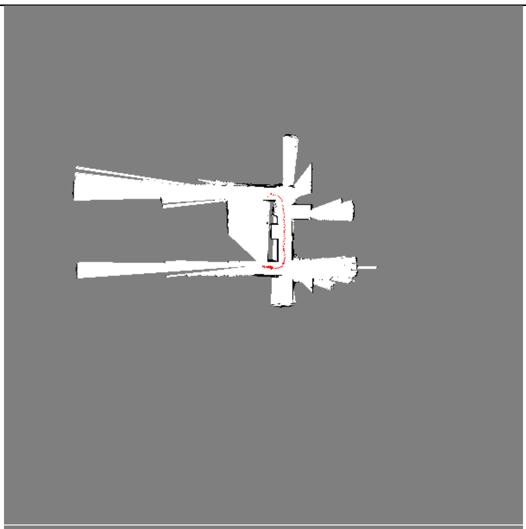
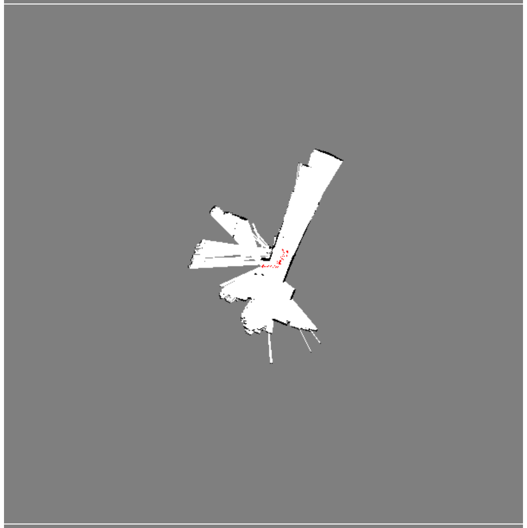
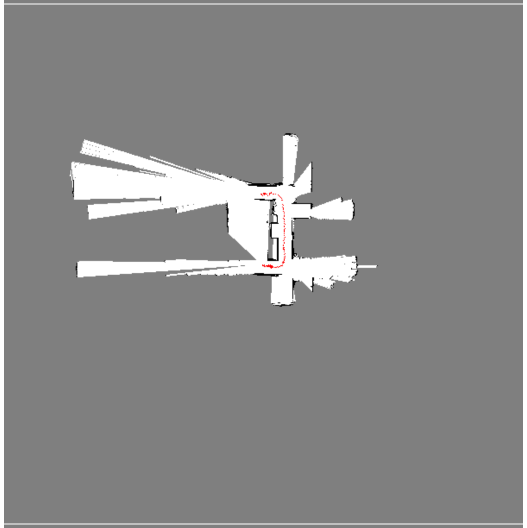
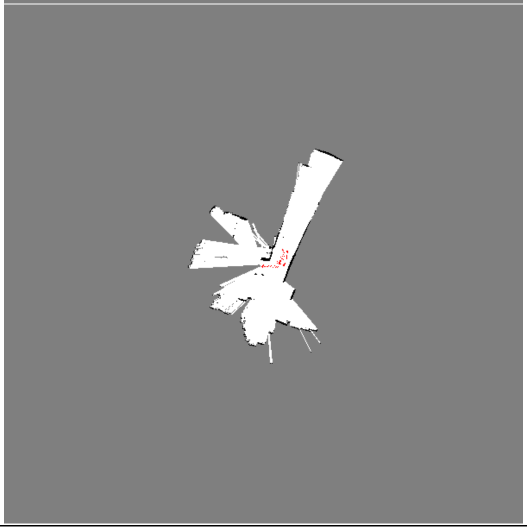
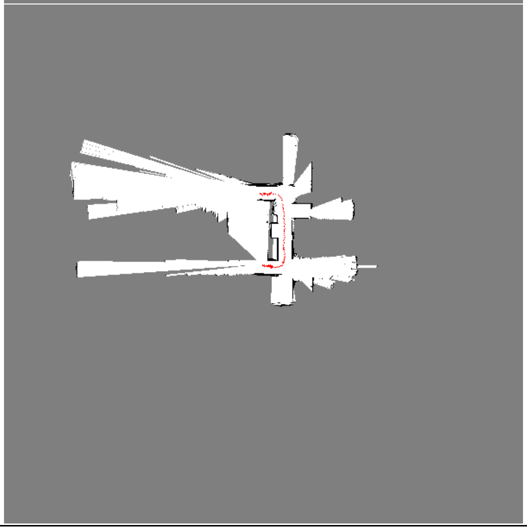
A53238541   Chun Hu

Data 0	Data 1
	
	
	

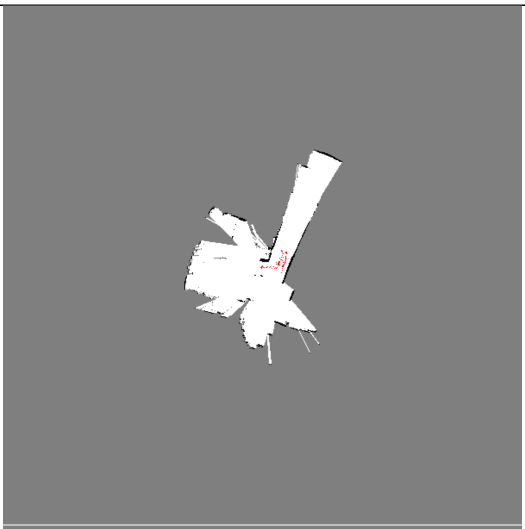
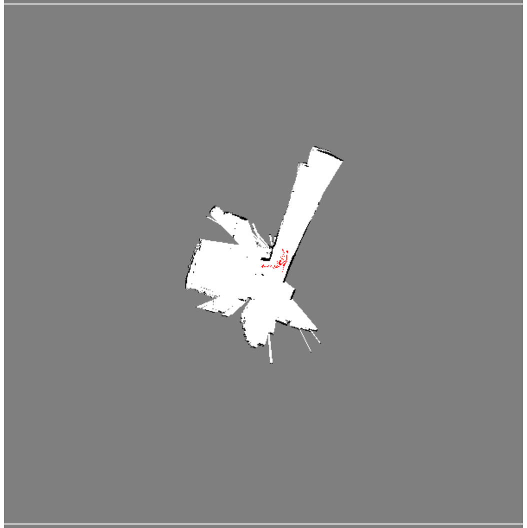
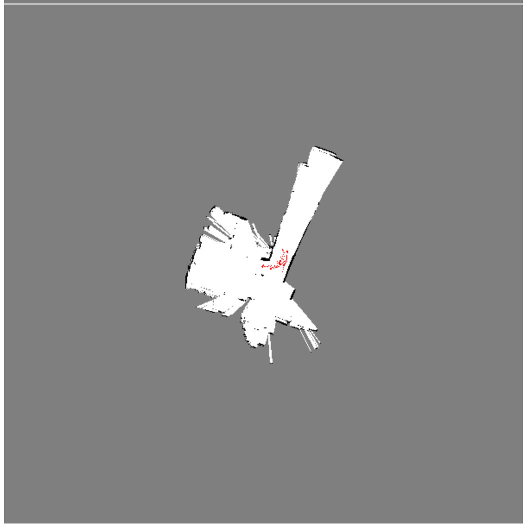
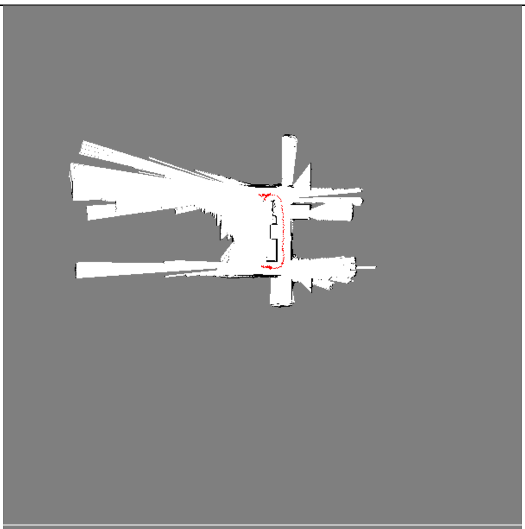
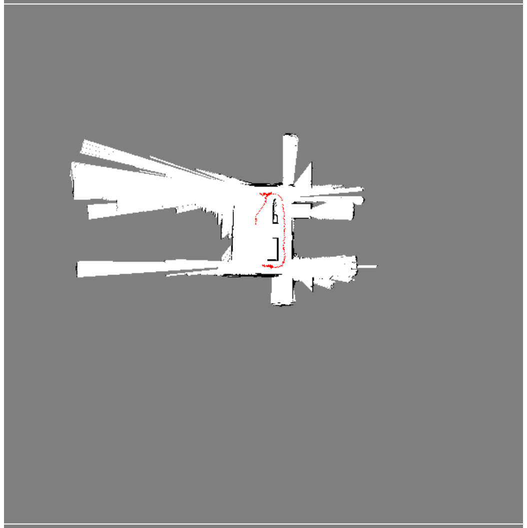
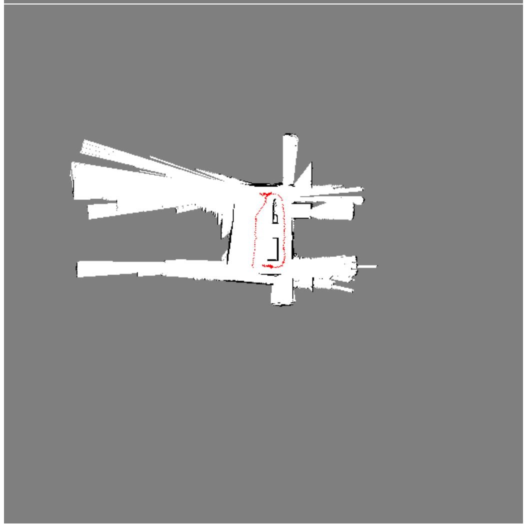
A53238541   Chun Hu

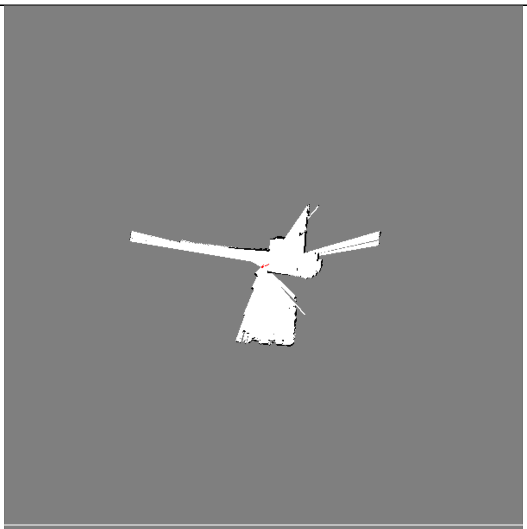
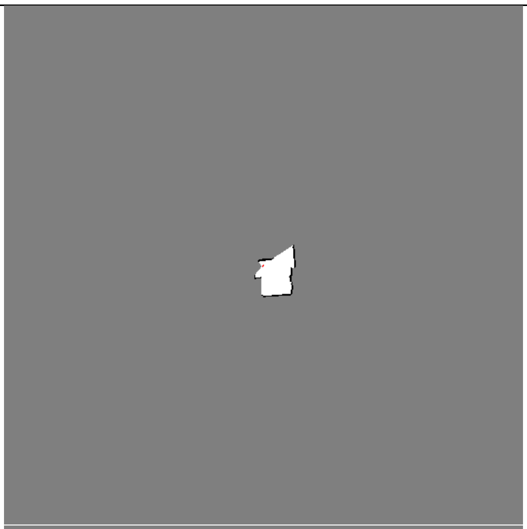
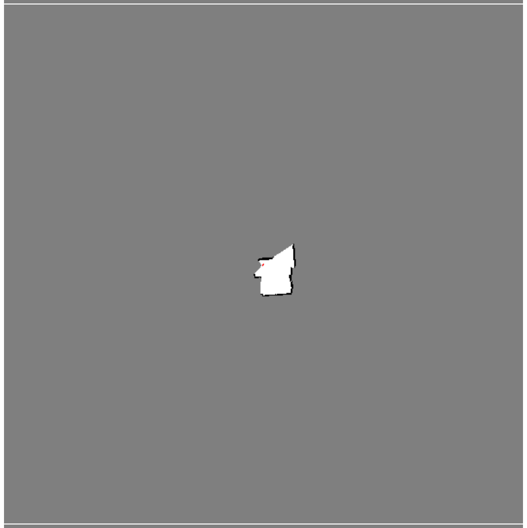
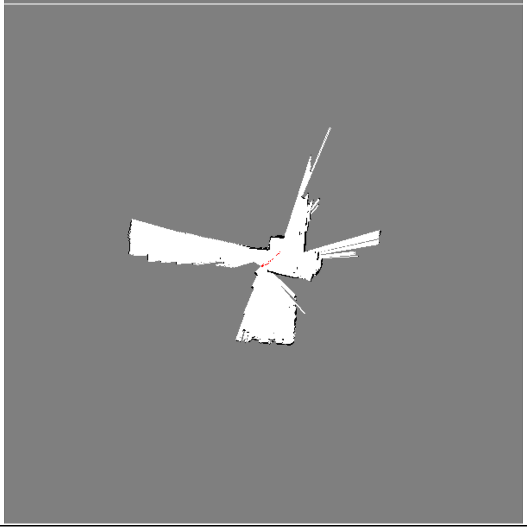
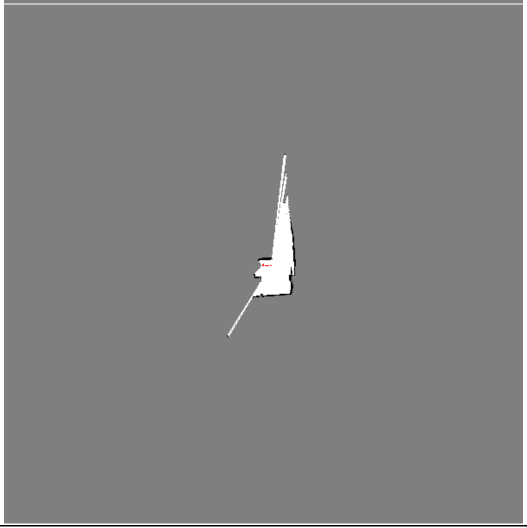
A53238541   Chun Hu

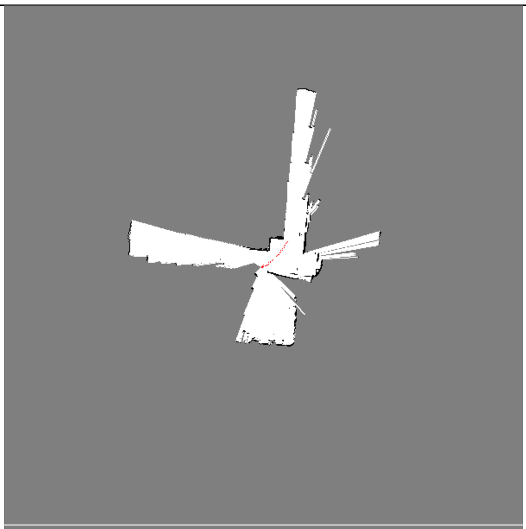
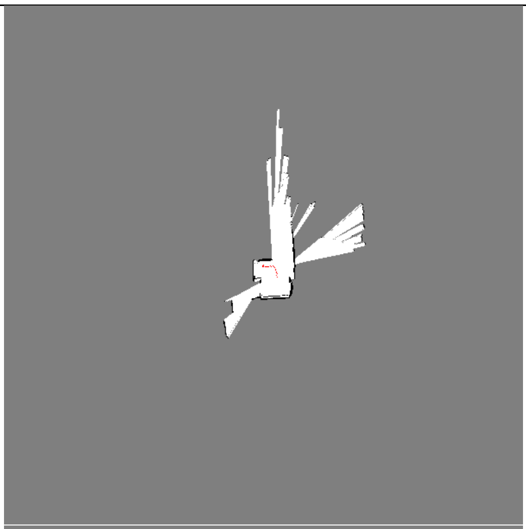
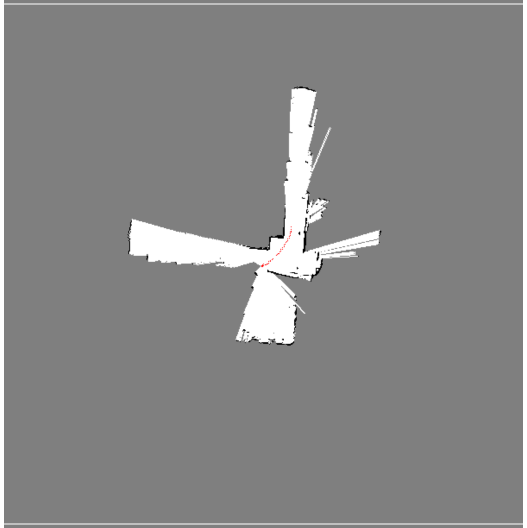
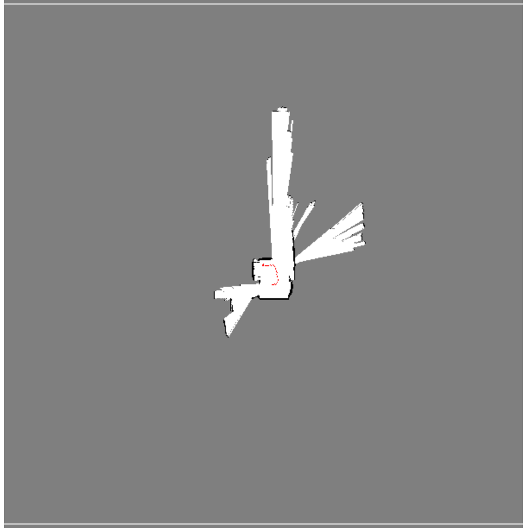
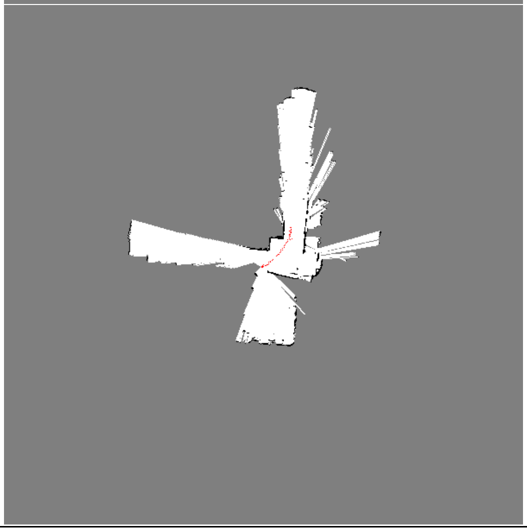
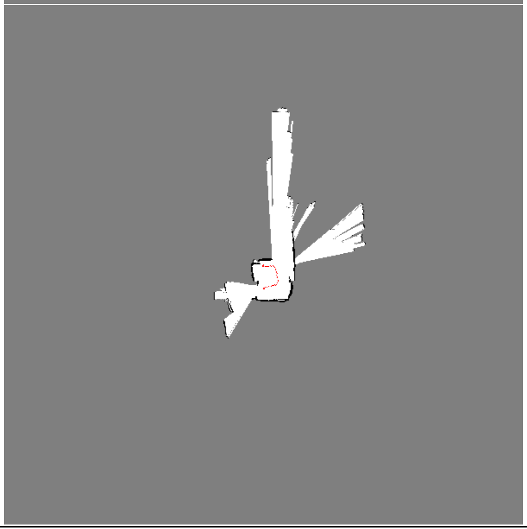
A53238541 Chun Hu

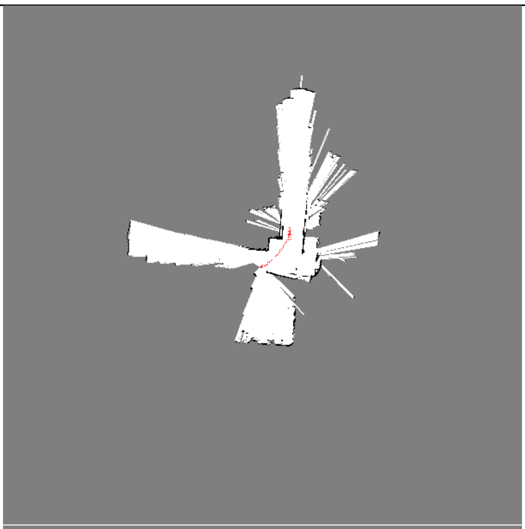
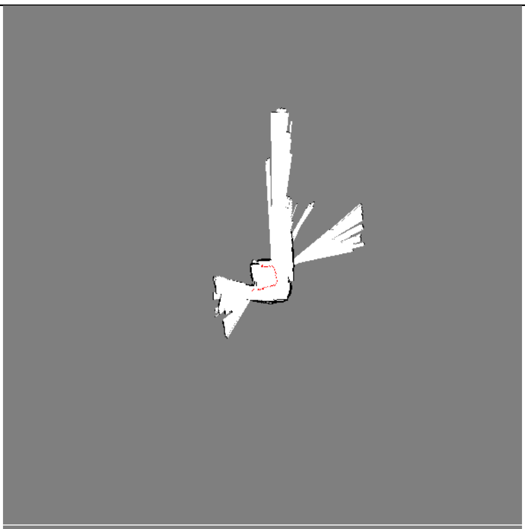
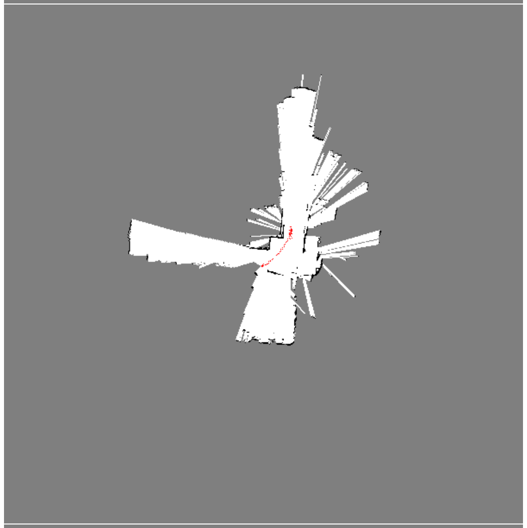
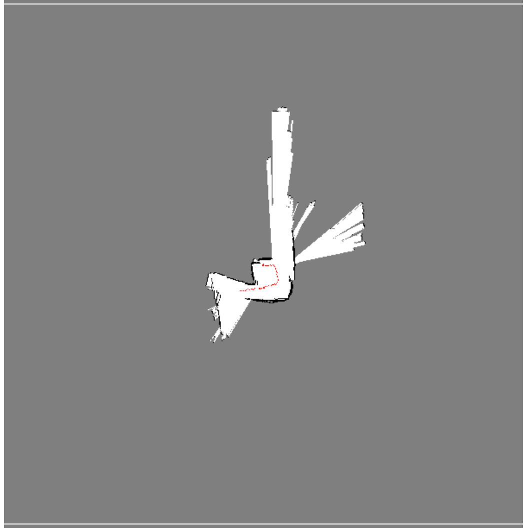
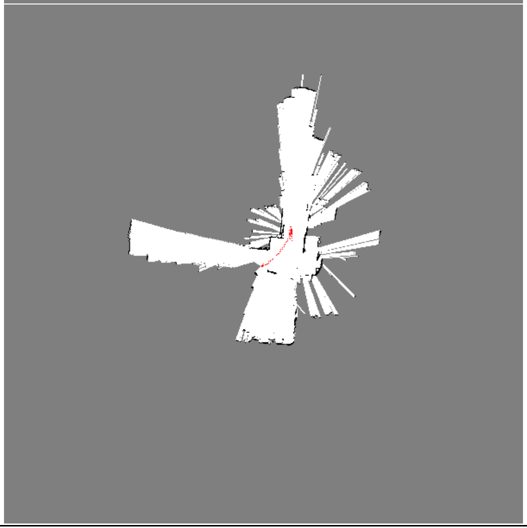
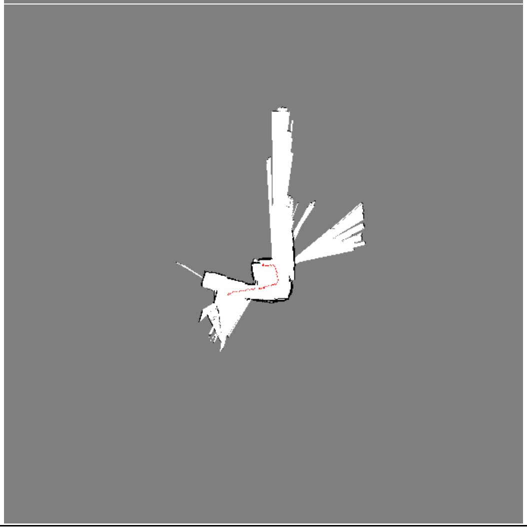
A53238541 Chun Hu

Data 2	Data 3
	
	
	

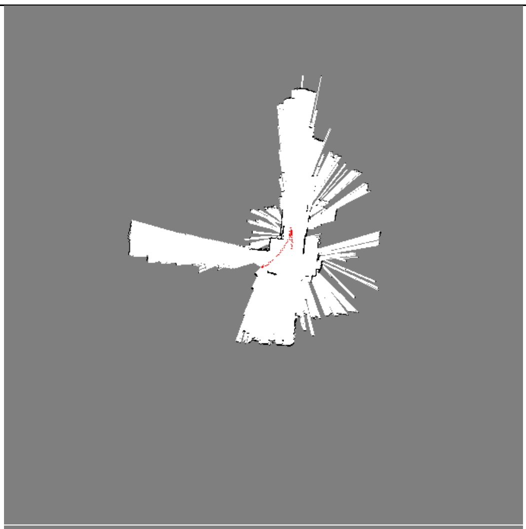
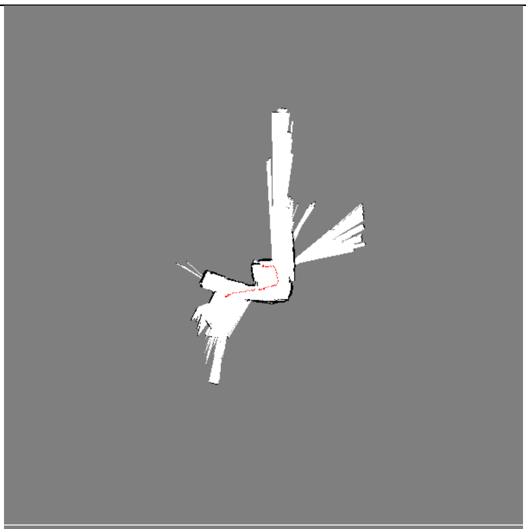
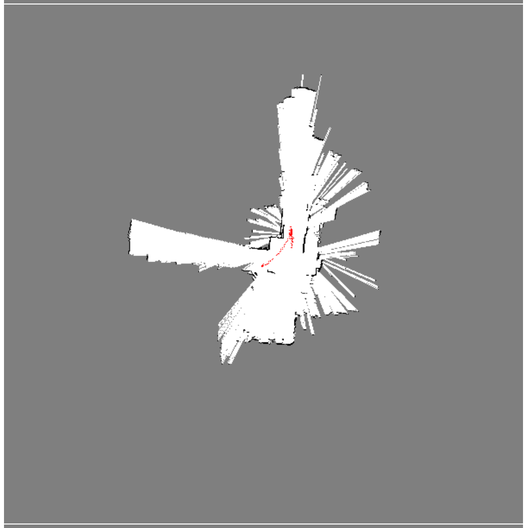
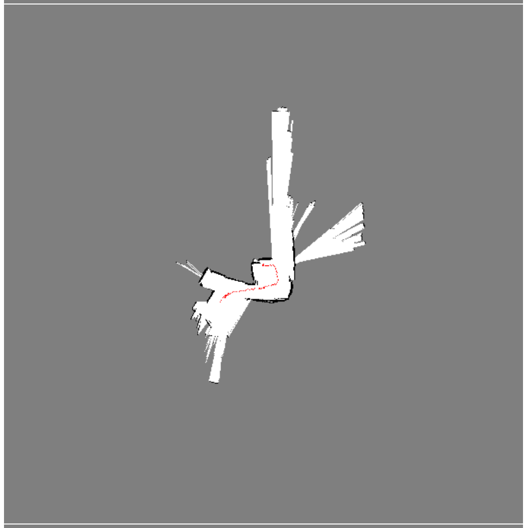
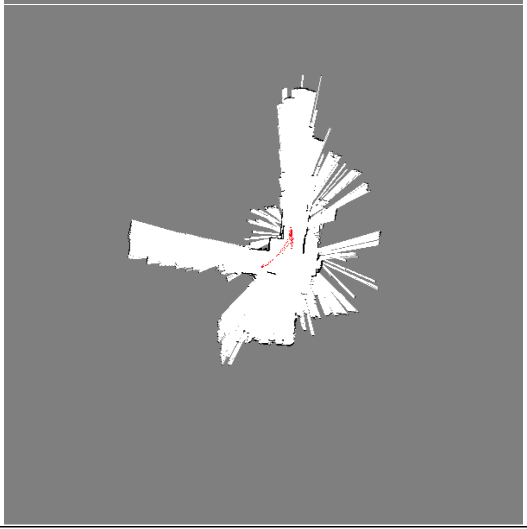
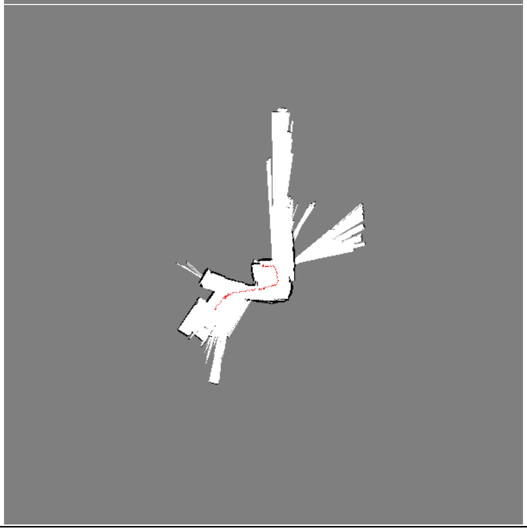
A53238541   Chun Hu

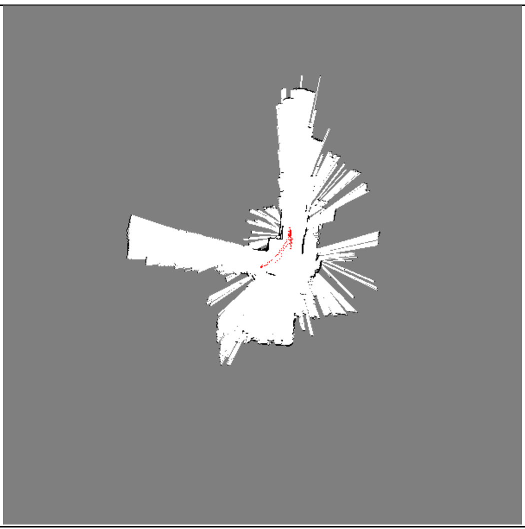
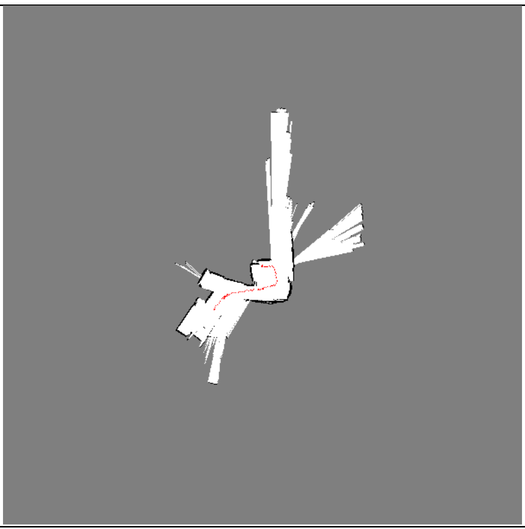
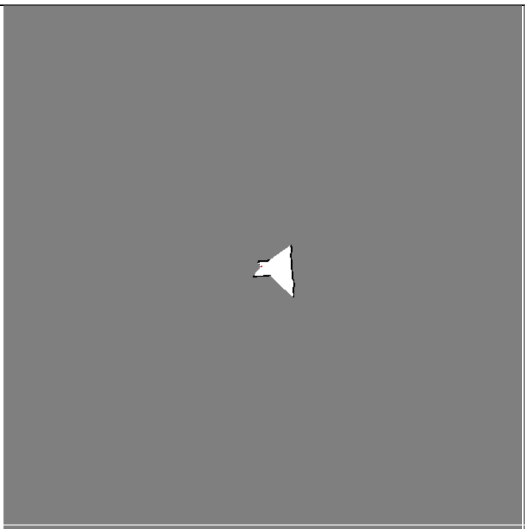
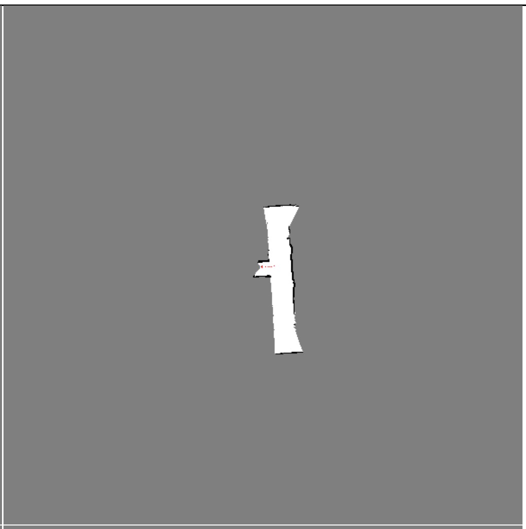
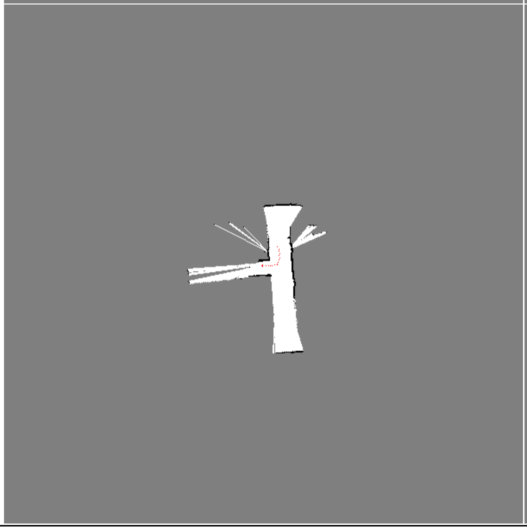
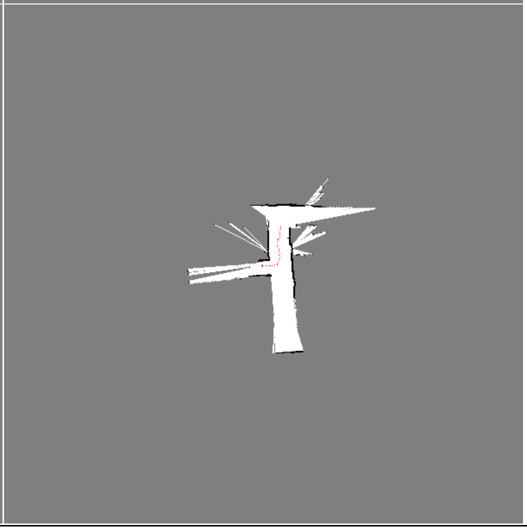
A53238541 Chun Hu

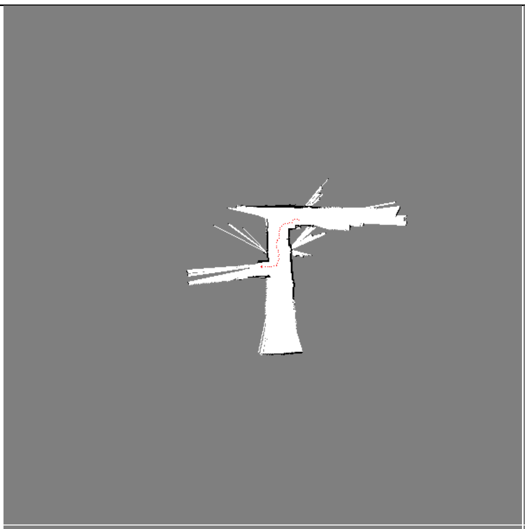
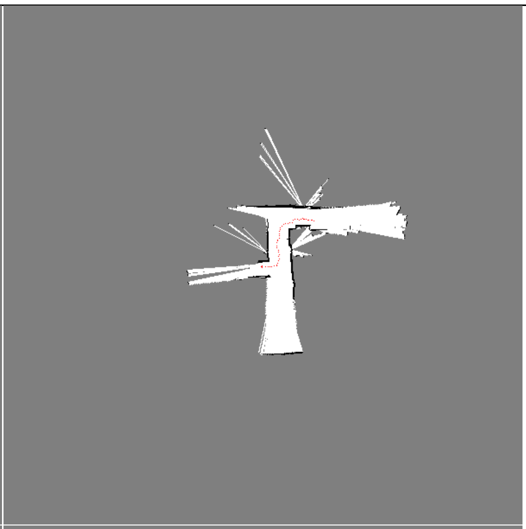
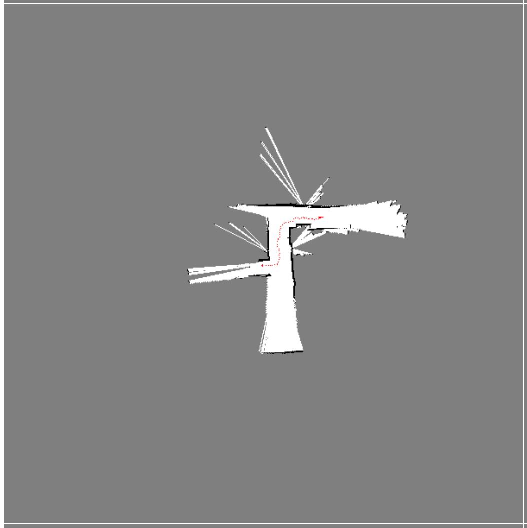
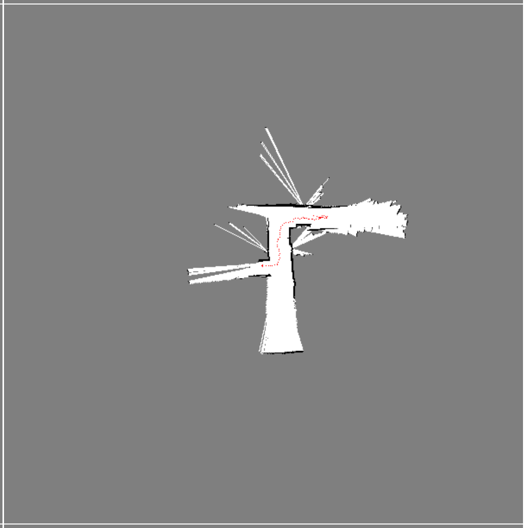
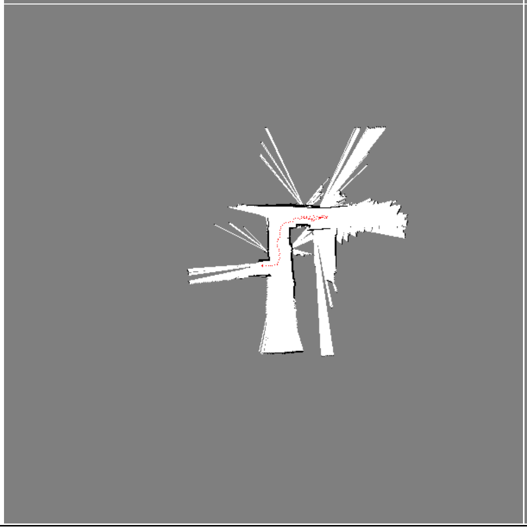
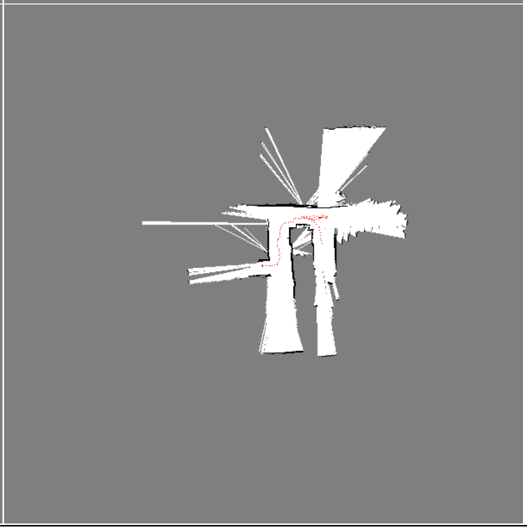
	
	
	

A53238541 Chun Hu

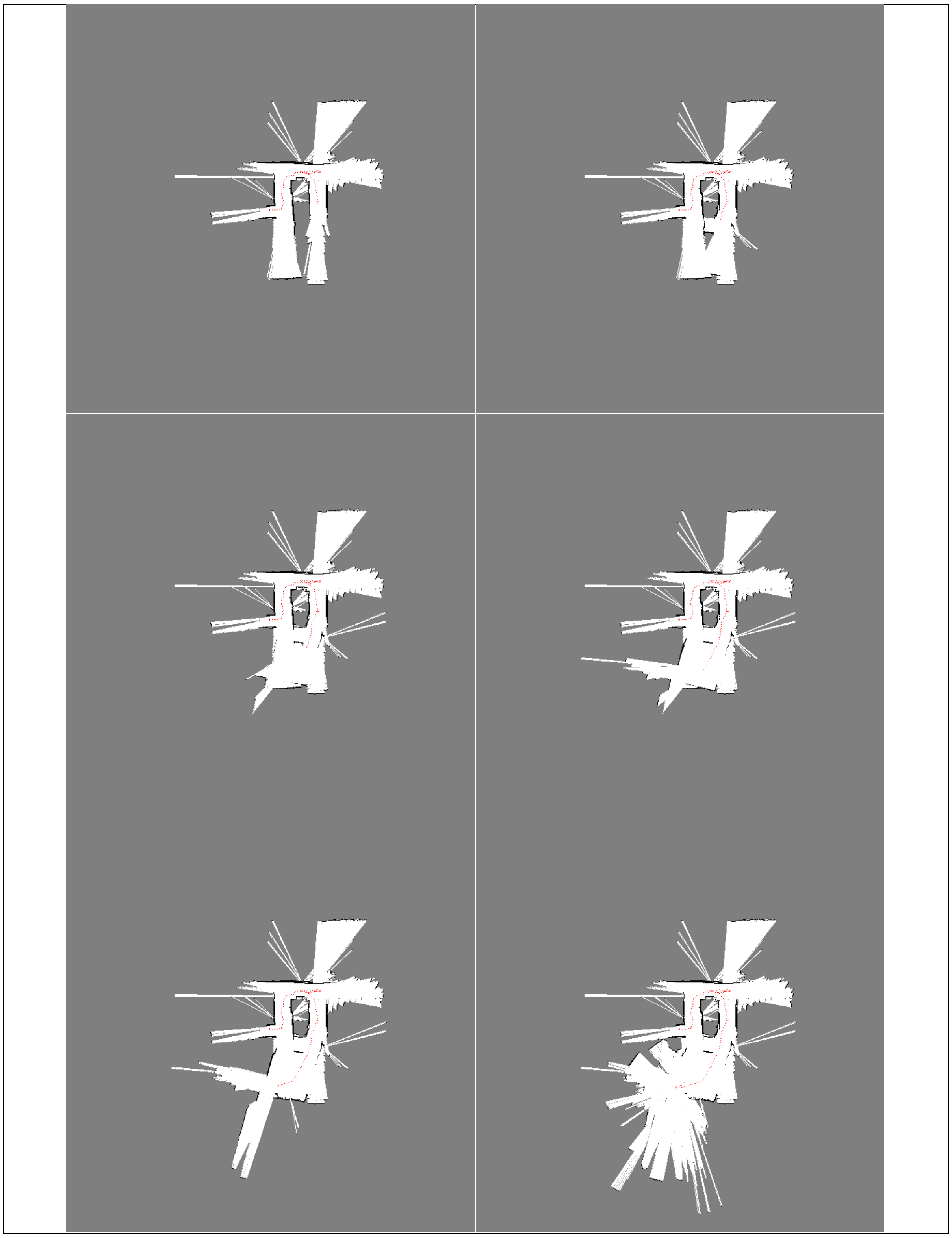
	
	
	

A53238541 Chun Hu

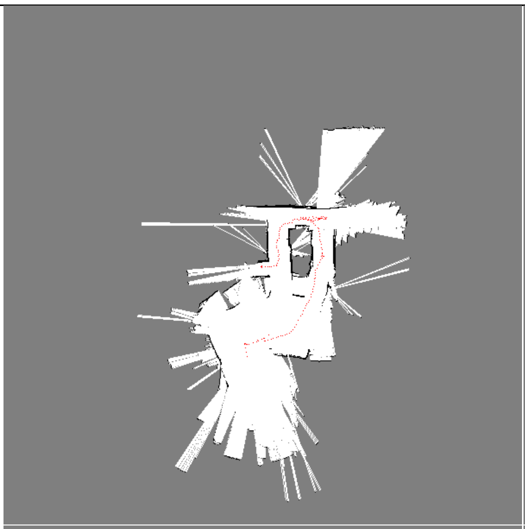
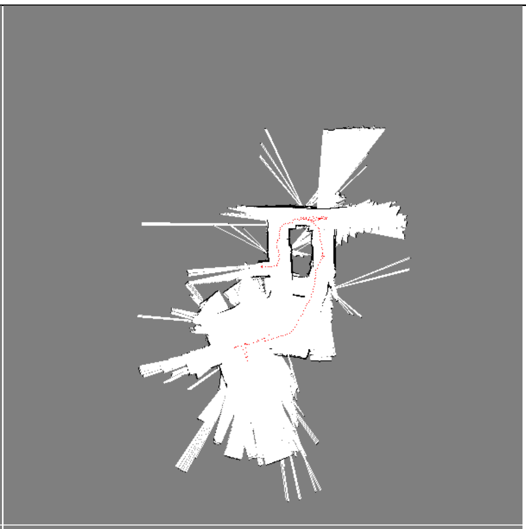
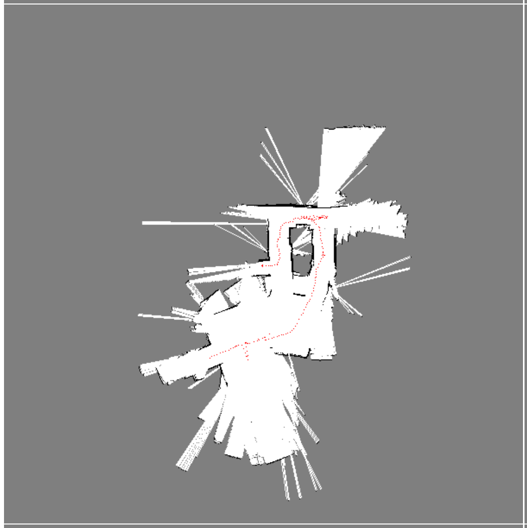
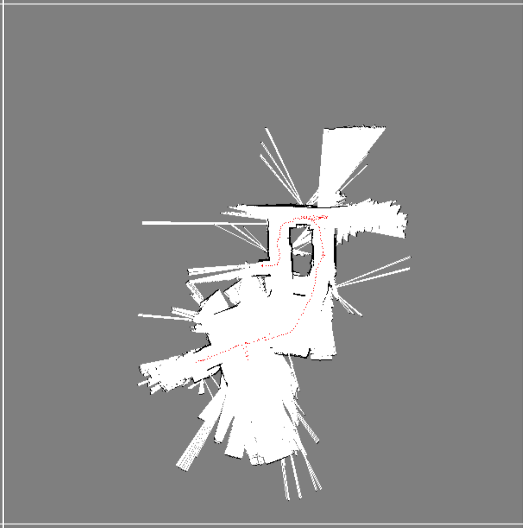
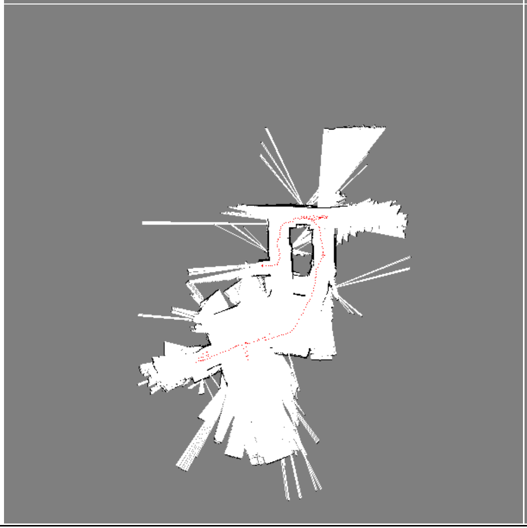
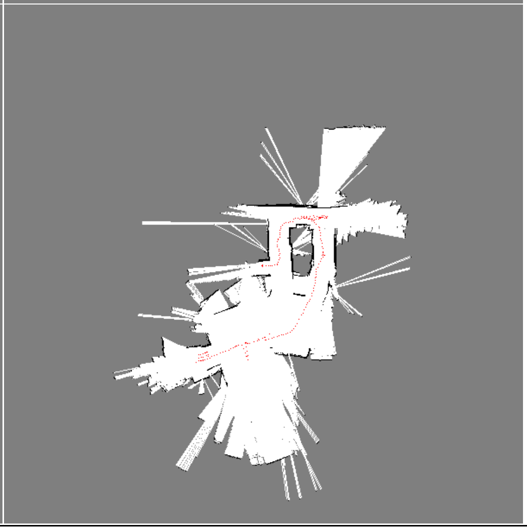
	
Test data	
	
	

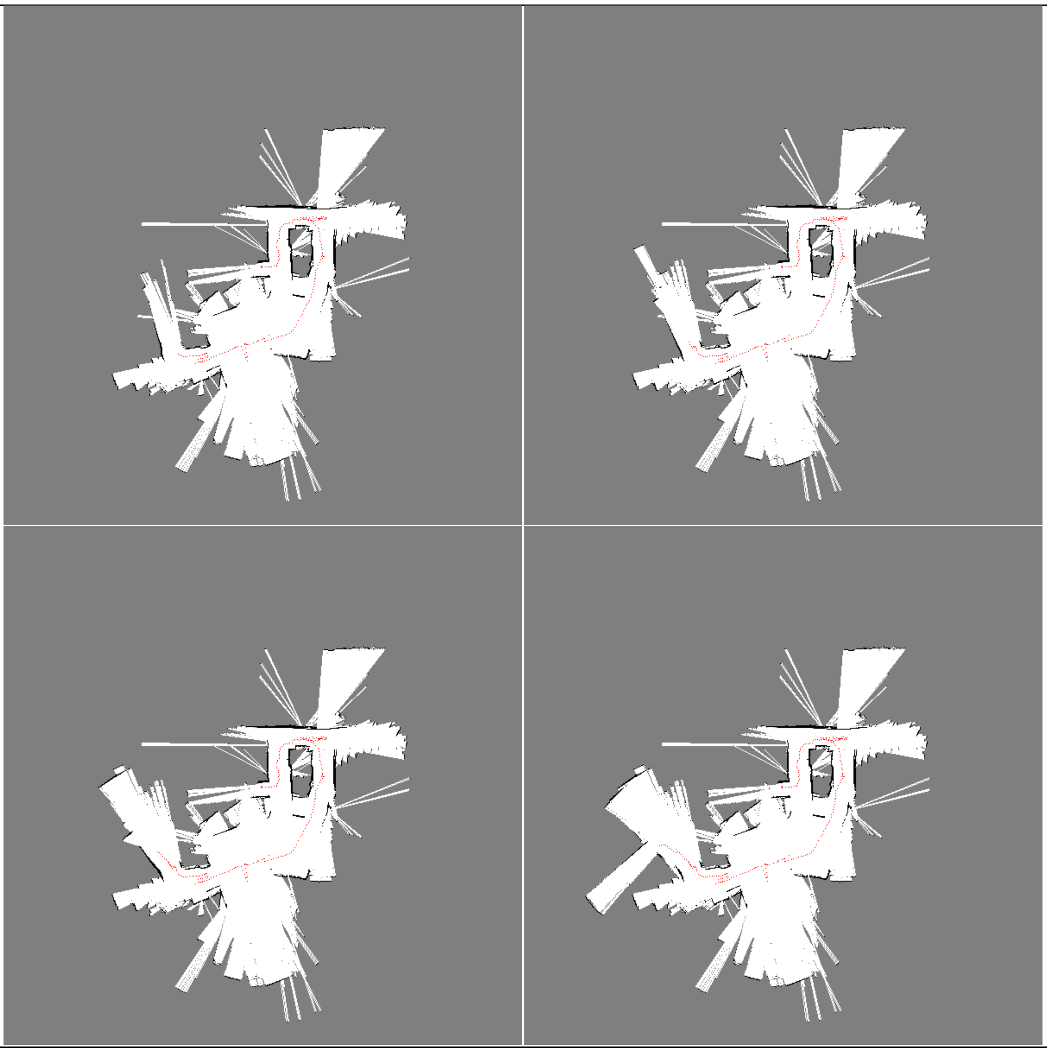
A53238541   Chun Hu



A53238541   Chun Hu

A53238541   Chun Hu



A53238541   Chun Hu