Format of Data File

- N: the number of robot frames
  60
- X_i, Y_i, Q_i: X-position, Y-position, Angle Q of robot at i-th robot frame
  600 1800 0
- M_i: the number of measurement at i-th frame
  36
- S_Q[]: the angle of each measurement [degree]
  0 10 20 30 40 50 60 70 80 90 100 110 120 130 140 150 160 170 180 190
  200 210 220 230 240 250 260 270 280 290 300 310 320 330 340 350
- S_D[]: the distance of each measurement [mm], -1 if no measurement
  -1 -1 -1 -1 934 784 693 639 610 600 610 639 693 784 -1 -1 -1 -1 -1 -1 -1 -1
  -1 782 692 638 609 600 609 638 692 782 932 -1 -1 -1
- i = 1, … N

read_data.py (1)

```
# Read N, the number of cycles
N = int( input() )

# Initialize X, Y, Q, M by the size of N
X = [ 0 ] * N
Y = [ 0 ] * N
Q = [ 0 ] * N
M = [ 0 ] * N
S_Q = [ [] ] * N
S_D = [ [] ] * N
```

read_data.py (2)

```
# Read every frame
for i in range( N ):
    X[ i ], Y[ i ], Q[ i ] = map( float, input().split() )
    M[ i ] = int( input() )
    S_Q[ i ] = [ 0 ] * M[ i ]
    S_D[ i ] = [ 0 ] * M[ i ]

    S_Q[ i ] = [ float( x ) for x in input().split() ]
    S_D[ i ] = [ float( x ) for x in input().split() ]
```

read_data.py (3)

```
# Output read data
print( N )
print( X )
print( Y )
print( Q )
print( M )
print( S_Q )
print( S_D )
```

read_data_disp.py

```
# Output read data
for i in range( N ):
    x = []
    y = []
    for j in range( M[ i ] ):
        if S_D[ i ][ j ] != -1:
            r = math.radians( S_Q[ i ][ j ] )
            x.append( math.cos( r ) * S_D[ i ][ j ] )
            y.append( math.sin( r ) * S_D[ i ][ j ] )
    plt.scatter( x, y )
    plt.xlim( -1000, 1000 )
    plt.ylim( -1000, 1000 )
    plt.title( i )
    plt.show( )
```

occupancy_grid_map_disp.py (1)

```
import math
import matplotlib.pyplot as plt
import numpy as np

def frame_trans( x_local, y_local, x, y, q ):
    r = math.radians( q )
    x_world = x_local * math.cos( r ) - math.sin( r ) * y_local + x
    y_world = x_local * math.sin( r ) + math.cos( r ) * y_local + y
    return x_world, y_world
```

occupancy_grid_map_disp.py (2)

```
# Map generation
x = []
y = []
for i in range( N ):
    for j in range( M[ i ] ):
        if S_D[ i ][ j ] != -1:
            r = math.radians( S_Q[ i ][ j ] )
            x_local = math.cos( r ) * S_D[ i ][ j ]
            y_local = math.sin( r ) * S_D[ i ][ j ]
            x_world, y_world = frame_trans( x_local, y_local, X[ i ], Y[ i ], Q[ i ] )
            x.append( x_world )
            y.append( y_world )
```

occupancy_grid_map_disp.py (3)

```
# Output read data
heatmap, xedges, yedges = np.histogram2d( x, y, bins = 100 )
extent = [xedges[0], xedges[-1], yedges[0], yedges[-1]]
plt.imshow( heatmap.T, extent = extent, origin = 'lower' )
plt.show()
```