# Homework 2

**Task 1 If there are 100 lines in the grating, what is the smallest detectable change in**

**motor-shaft angle?**

360/100=3.6

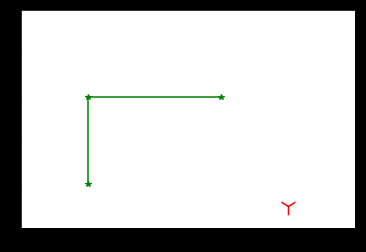
3.6/4=0.9

**Task 2 Explain how to determine the rotation directions if the following encoders are used. List two concerns while choosing an encoder.**

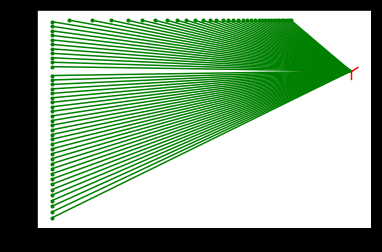
The ordering of which square wave produces a rising edge first identifies the direction of rotation. The accuracy of optical encoders and one or two。

**Task 3 Simulate the process of** **mapping of a room by using a** **moving range sensor which knows its location accurately (****randomly walking, or moving along a circle)**

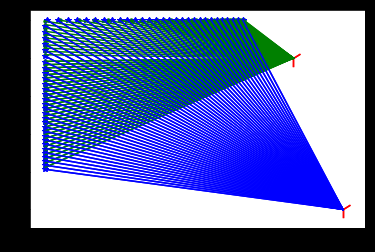
The environment picture is: (the green line is wall, and the red point is robot )



After using a moving range sensor:（the green point are sensed by robot）



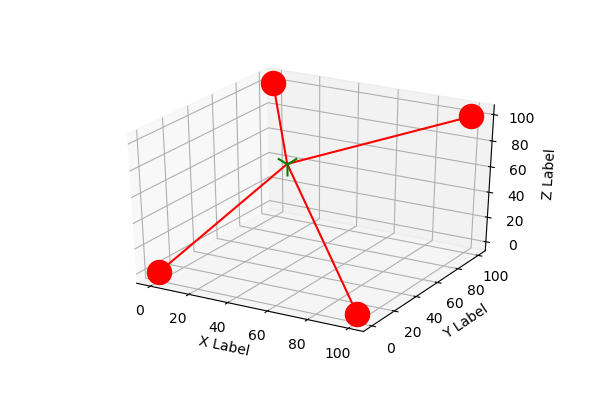
And then randomly walking:



**Task 4 Simulate the process of localization with GPS signals. When sender-receiver clocks are either synchronized or** **not synchronized, how many satellites are needed to achieve 3D accurate positions, respectively?**

synchronized :3 not synchronized :4

The four red points are our GPS, and the red lines are signals



**My code: (python3, jupter)**

**Task 3**

import matplotlib.pyplot as plt

import numpy as np

import math

def has\_point(k,b):#最简单的实现返回那个点的坐标

if k==0:

return -1

x2=(3-b)/k

y1=k\*1+b

if x2<=3 and x2>=1:

return x2,3.0

elif y1<=3 and y1>=1:

return 1.0,y1

else:

return -1

x=[1,1,3]

y=[1,3,3]#墙

plt.plot(x, y, 'g\*')

plt.plot(x[:2], y[:2],'g')

plt.plot(x[1:], y[1:],'g')

thex=[]

they=[]

robot=[4, 0.5]#robot当前位置

plt.scatter(robot[0],robot[1],color='r',marker='1',s=300)

for i in range(0,180):

deltat = i\*math.pi/180#每次多转1度

x1 = robot[0] + 10\*math.cos(deltat)

y1 = robot[1] + 10\*math.sin(deltat)

k=math.tan(deltat)

b=y1-x1\*k#计算当前发射信号的角度的函数

if has\_point(k,b)!=-1:

[x2,y2]=has\_point(k,b)

thex.append(x2)

they.append(y2)

plt.xlim(0,5)

plt.ylim(0,5)

plt.show()

plt.plot(thex,they, 'y\*')

plt.scatter(robot[0],robot[1],color='r',marker='1',s=300)

for i in range(len(thex)):

plt.plot([thex[i],robot[0]],[they[i],robot[1]],'b')

plt.show()

**Task 4**

import numpy as np

import matplotlib.pyplot as plt

from sympy import \*

#四个gps的点

G1=[100,100,100,0.03]

G2=[0,0,0,0.06]

G3=[0,100,100,0.055]

G4=[100,0,0,0.06]

d=[109,13,14,9]

#新建的变量

x=symbols('x')

y=symbols('y')

z=symbols('z')

v=symbols('v')

c=3\*(10\*\*3)

#解方程

solution=solve([

((G1[0]-x)\*\*2+(G1[1]-y)\*\*2+(G1[2]-z)\*\*2)-(c\*(G1[3]-v)-d[0])\*\*2,

((G2[0]-x)\*\*2+(G2[1]-y)\*\*2+(G2[2]-z)\*\*2)-(c\*(G2[3]-v)-d[1])\*\*2,

((G3[0]-x)\*\*2+(G3[1]-y)\*\*2+(G3[2]-z)\*\*2)-(c\*(G3[3]-v)-d[2])\*\*2,

((G4[0]-x)\*\*2+(G4[1]-y)\*\*2+(G4[2]-z)\*\*2)-(c\*(G4[3]-v)-d[3])\*\*2],

[x,y,z,v]

)

target=list(solution[0])

#画图

import mpl\_toolkits.mplot3d

ax=plt.subplot(111,projection='3d')

for e in range(0,len(target)):

target[e]=float(target[e])

x,y,z=np.linspace(target[0],G1[0],10),np.linspace(target[1],G1[1],10),np.linspace(target[2],G1[2],10)

ax.plot(x,y,z,c='r')

x,y,z=np.linspace(target[0],G2[0],10),np.linspace(target[1],G2[1],10),np.linspace(target[2],G2[2],10)

ax.plot(x,y,z,c='r')

x,y,z=np.linspace(target[0],G3[0],10),np.linspace(target[1],G3[1],10),np.linspace(target[2],G3[2],10)

ax.plot(x,y,z,c='r')

x,y,z=np.linspace(target[0],G4[0],10),np.linspace(target[1],G4[1],10),np.linspace(target[2],G4[2],10)

ax.plot(x,y,z,c='r')

ax.scatter(target[0],target[1],target[2], c='g', marker='1',s=300)

ax.scatter(G1[0],G1[1],G1[2], c='r', marker='o',s=300)

ax.scatter(G2[0],G2[1],G2[2], c='r', marker='o',s=300)

ax.scatter(G3[0],G3[1],G3[2], c='r', marker='o',s=300)

ax.scatter(G4[0],G4[1],G4[2], c='r', marker='o',s=300)

ax.set\_xlabel('X Label')

ax.set\_ylabel('Y Label')

ax.set\_zlabel('Z Label')

plt.savefig('test2png.png', dpi=100)

plt.show()