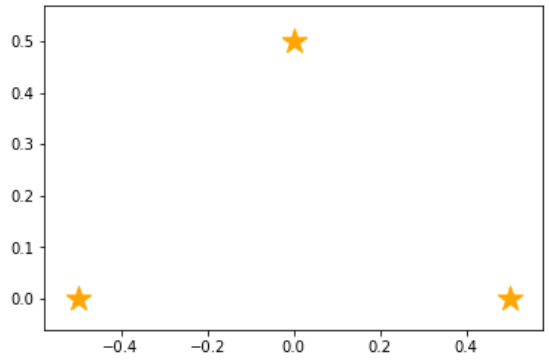
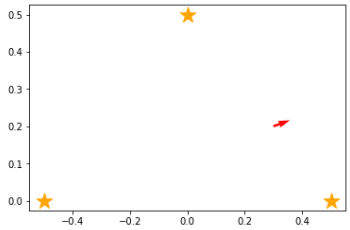
```
Python 3.6.1 |Anaconda 4.4.0 (64-bit)| (default, May 11 2017, 13:25:24) [MSC v.1900 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 5.3.0 -- An enhanced Interactive Python.
?         -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help      -> Python's own help system.
object?   -> Details about 'object', use 'object??' for extra details.

In [1]:

In [1]: %matplotlib inline
   ...: import numpy as np
   ...: from copy import copy
   ...: import math, random
   ...: import matplotlib.pyplot as plt                  #   for plotting data
   ...: from matplotlib.patches import Ellipse      #  for drawing

In [2]: actual_landmarks = [np.array([-0.5,0.0]),np.array([0.5,0.0]),np.array([0.0,0.5])]
   ...:
   ...: def draw_landmarks(landmarks):
   ...:     xs = [ e[0] for e in landmarks]
   ...:     ys = [ e[1] for e in landmarks]
   ...:     plt.scatter(xs,ys,s=300,marker="*",label="landmarks",color="orange")
   ...:
   ...: draw_landmarks(actual_landmarks)
```

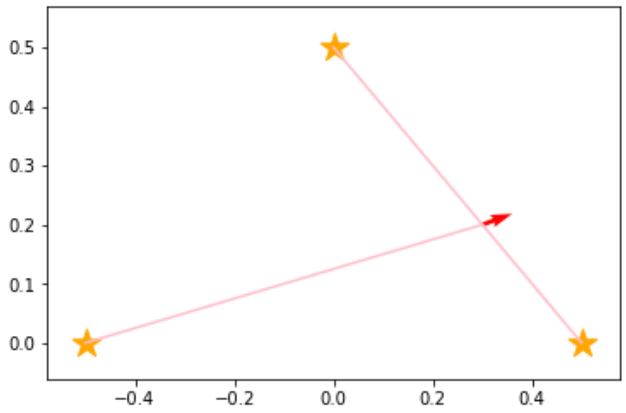


```
In [3]: actual_x = np.array([0.3,0.2,math.pi*20.0/180])   #ロボットの実際の姿勢
   ...:
   ...: def draw_robot(pose):
   ...:     plt.quiver([pose[0]],[pose[1]],[math.cos(pose[2])],[math.sin(pose[2])],color="red",label="actual robot motion")
   ...:
   ...: draw_robot(actual_x)
   ...: draw_landmarks(actual_landmarks)
```



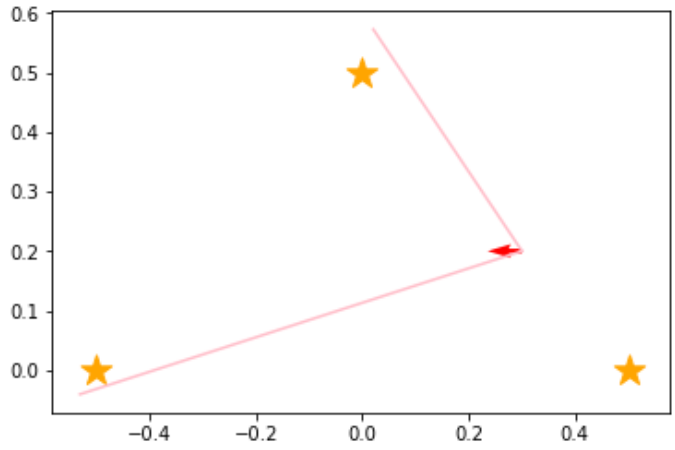

```
In [4]: def relative_landmark_pos(pose,landmark):
   ...:     x,y,theta = pose
   ...:     lx,ly = landmark
   ...:     distance = math.sqrt((x -lx)**2 + (y-ly)**2)
   ...:     direction = math.atan2(ly-y, lx-x) - theta
   ...:
   ...:     return (distance, direction,lx,ly) # 実際の位置も一緒に返す

In [5]: measurements = [ relative_landmark_pos(actual_x,e) for e in actual_landmarks]
   ...: print(measurements)
[(0.8246211251235323, -3.2456798408617948, -0.5, 0.0), (0.28284271247461906, -1.1344640137963142, 0.5, 0.0), (0.4242640687119285, 2.0071286397934789, 0.0, 0.5)]
```

```
In [6]: def draw_observation(pose,  measurement):
   ...:     x,y,theta = pose
   ...:     distance, direction,lx,ly = measurement
   ...:     lx = distance*math.cos(theta + direction) + x
   ...:     ly = distance*math.sin(theta + direction) + y
   ...:     plt.plot([pose[0],  lx],[pose[1],  ly],color="pink")
   ...:
   ...: def draw_observations(pose,  measurements):
   ...:     for m in measurements:
   ...:         draw_observation(pose, m)

In [7]: draw_landmarks(actual_landmarks)
   ...: draw_robot(actual_x)
   ...: draw_observations(actual_x,measurements)
```




```
In [8]: def observation(pose, landmark):
   ...:     actual_distance,actual_direction,lx,ly = relative_landmark_pos(pose,landmark)
   ...:     # 方向の制限(cosの値が正)
   ...:     if (math.cos(actual_direction) < 0.0):
   ...:         return None
   ...:
   ...:     measured_distance = random.gauss(actual_distance,actual_distance*0.1)
   ...:     measured_direction = random.gauss(actual_direction,5.0/180.0*math.pi)
   ...:
   ...:     return (measured_distance, measured_direction,lx,ly)
   ...:
   ...: def observations(pose,landmarks):
   ...:     return filter(lambda x: x != None, [ observation(pose,e) for e in landmarks])
```

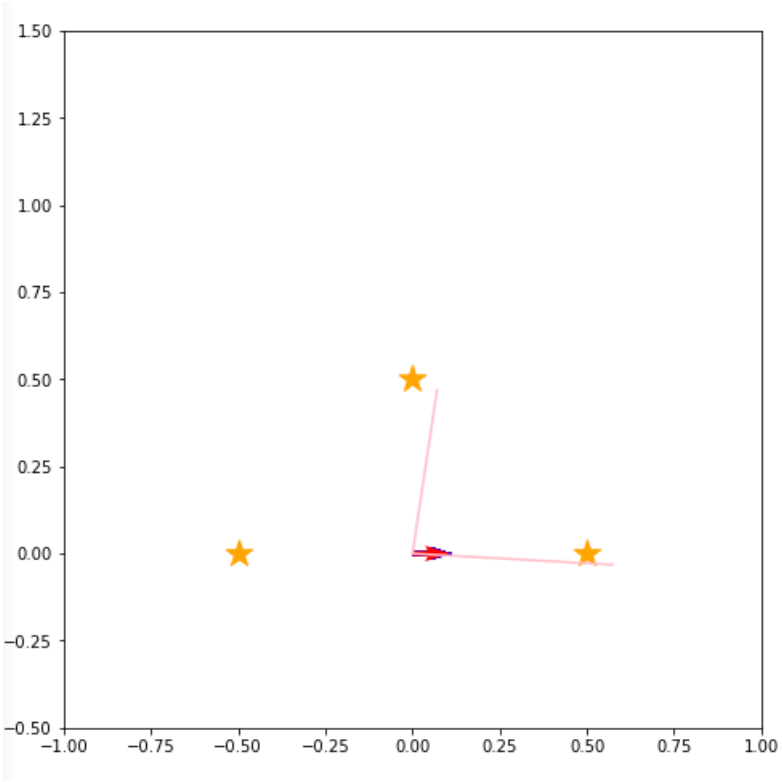
```
In [9]: actual_x = np.array([0.3,0.2,math.pi*180.0/180])   #姿勢は変えること
   ...: measurements = observations(actual_x, actual_landmarks)
   ...:
   ...: draw_landmarks(actual_landmarks)
   ...: draw_robot(actual_x)
   ...: draw_observations(actual_x, measurements)
```

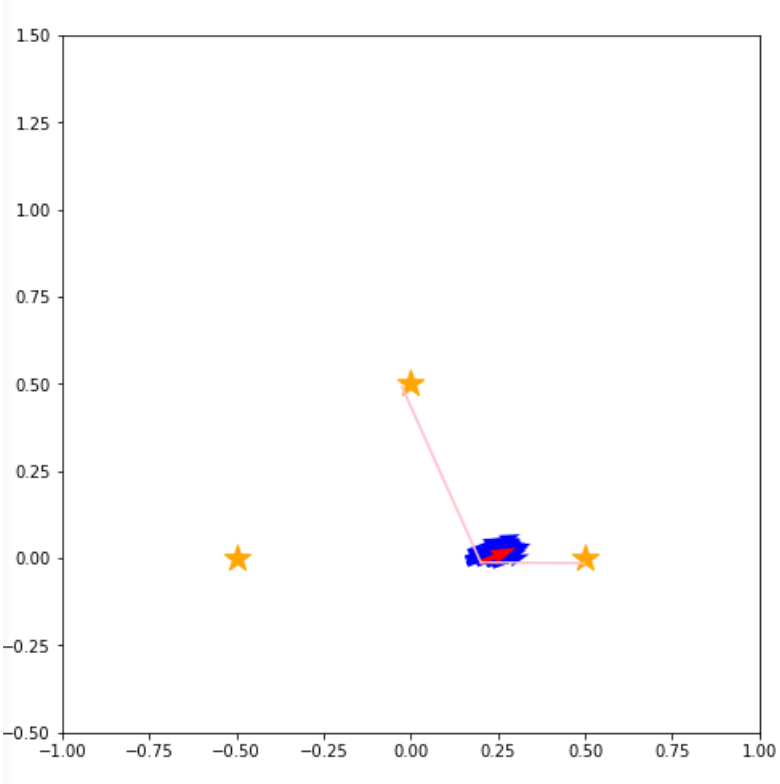


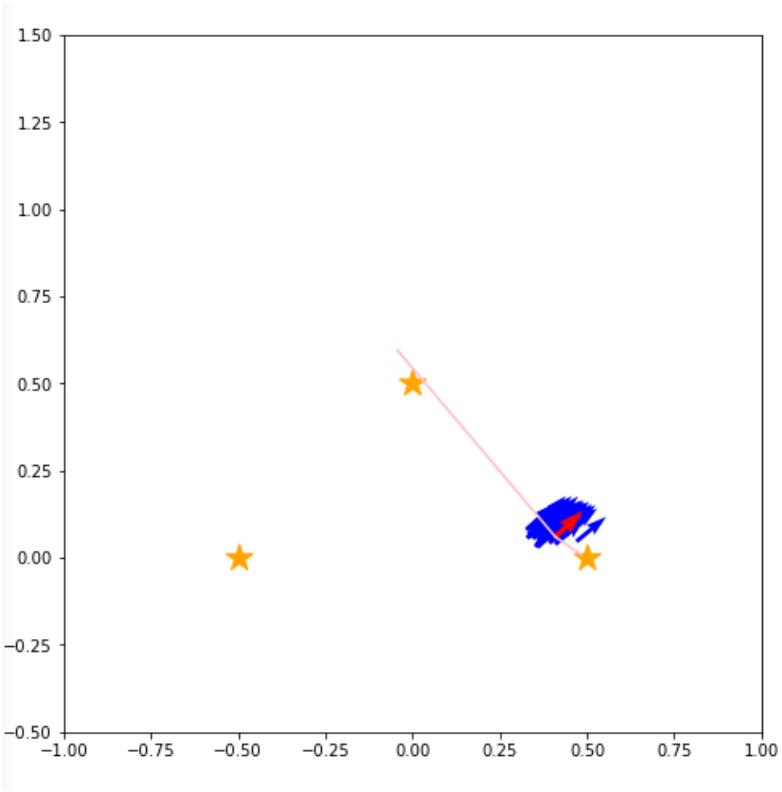

```
In [10]: class Particle:
    ...:     def __init__(self,w):
    ...:         self.pose = np.array([0.0,0.0,0.0])
    ...:         self.weight = w
    ...:
    ...:     def __repr__(self):
    ...:         return "pose: " + str(self.pose) + " weight: " + str(self.weight)
    ...:
    ...:
    ...: def f(x_old,u):
    ...:     pos_x, pos_y, pos_theta = x_old
    ...:     act_fw, act_rot = u
    ...:
    ...:     act_fw = random.gauss(act_fw,act_fw/10)
    ...:     dir_error = random.gauss(0.0, math.pi / 180.0 * 3.0)
    ...:     act_rot = random.gauss(act_rot,act_rot/10)
    ...:
    ...:     pos_x += act_fw * math.cos(pos_theta + dir_error)
    ...:     pos_y += act_fw * math.sin(pos_theta + dir_error)
    ...:     pos_theta += act_rot
    ...:
    ...:     return np.array([pos_x,pos_y,pos_theta])
    ...:
```

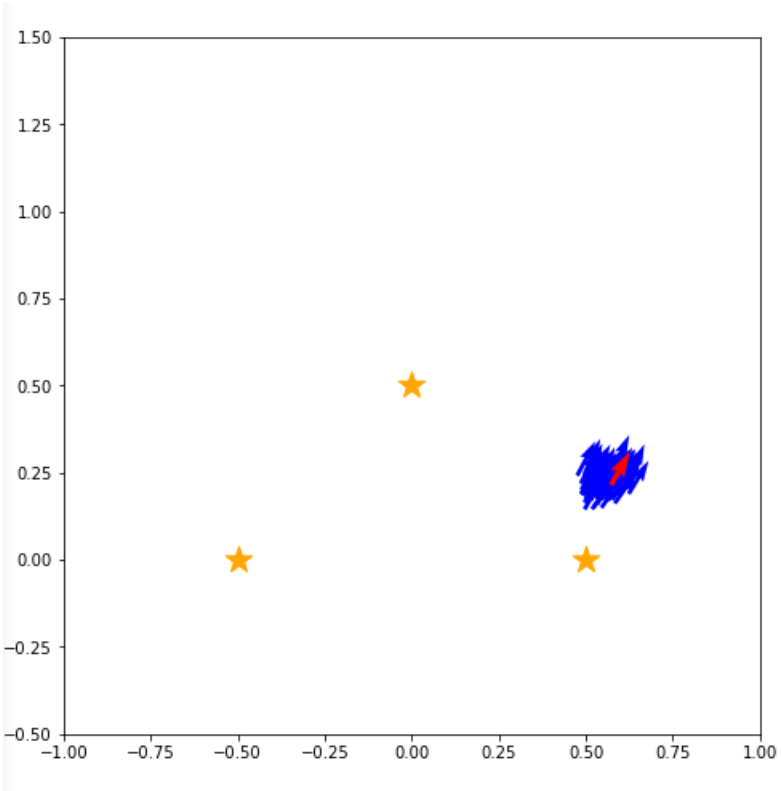
```
    ...:
    ...: def f(x_old,u):
    ...:     pos_x, pos_y, pos_theta = x_old
    ...:     act_fw, act_rot = u
    ...:
    ...:     act_fw = random.gauss(act_fw,act_fw/10)
    ...:     dir_error = random.gauss(0.0, math.pi / 180.0 * 3.0)
    ...:     act_rot = random.gauss(act_rot,act_rot/10)
    ...:
    ...:     pos_x += act_fw * math.cos(pos_theta + dir_error)
    ...:     pos_y += act_fw * math.sin(pos_theta + dir_error)
    ...:     pos_theta += act_rot
    ...:
    ...:     return np.array([pos_x,pos_y,pos_theta])
    ...:
    ...: ### 描画関数は少し変更を ###
    ...: def draw(pose,particles):
    ...:     fig = plt.figure(i,figsize=(8, 8))
    ...:     sp = fig.add_subplot(111, aspect='equal')
    ...:     sp.set_xlim(-1.0,1.0)
    ...:     sp.set_ylim(-0.5,1.5)
    ...:
    ...:     xs = [e.pose[0] for e in particles]
    ...:     ys = [e.pose[1] for e in particles]
    ...:     vxs = [math.cos(e.pose[2])*e.weight for e in particles]  #重みで長さを変えるようにしましょう
    ...:     vys = [math.sin(e.pose[2])*e.weight for e in particles]  #重みで長さを変えるようにしましょう
    ...:     plt.quiver(xs,ys,vxs,vys,color="blue",label="particles")
    ...:
    ...:     plt.quiver([pose[0]],[pose[1]],[math.cos(pose[2])],[math.sin(pose[2])],color="red",label="actual robot motion")

In [11]: actual_x = np.array([0.0,0.0,0.0])   #ロボットの実際の姿勢
    ...: particles = [Particle(1.0/100) for i in range(100)]
    ...: u = np.array([0.2,math.pi / 180.0 * 20]) #ロボットの移動
    ...:
    ...: import copy
    ...:
    ...: path = [actual_x]
    ...: particle_path = [copy.deepcopy(particles)]
    ...: measurementss = [observations(actual_x, actual_landmarks)]
    ...: for i in range(10):
    ...:     actual_x = f(actual_x,u)
    ...:     path.append(actual_x)
    ...:     measurementss.append(observations(actual_x,actual_landmarks))
    ...:
    ...:     for p in particles:
```

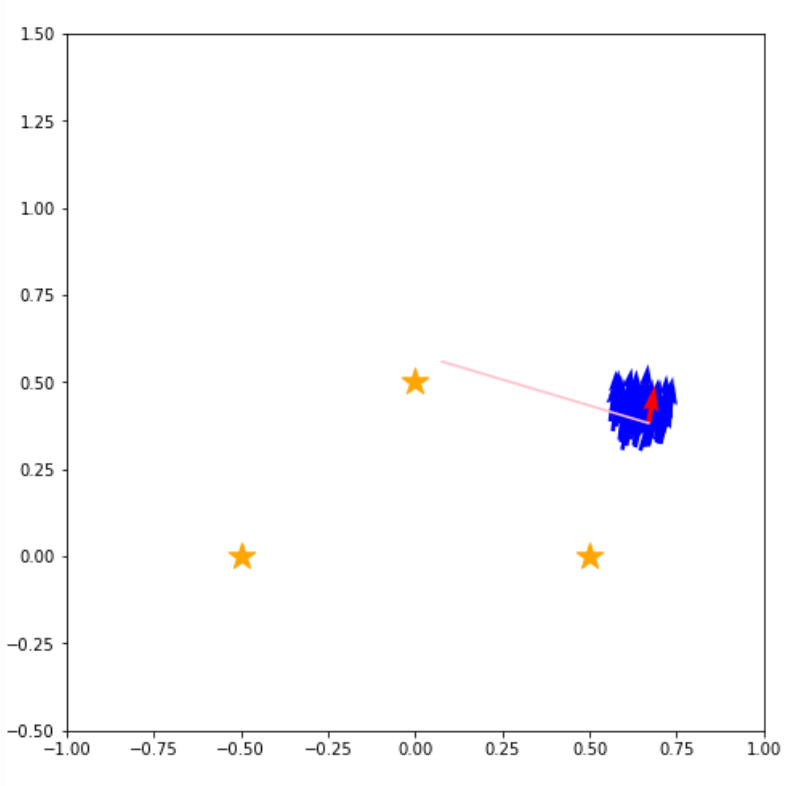
```
    ...:
    ...: import copy
    ...:
    ...: path = [actual_x]
    ...: particle_path = [copy.deepcopy(particles)]
    ...: measurementss = [observations(actual_x, actual_landmarks)]
    ...: for i in range(10):
    ...:     actual_x = f(actual_x,u)
    ...:     path.append(actual_x)
    ...:     measurementss.append(observations(actual_x,actual_landmarks))
    ...:
    ...:     for p in particles:
    ...:         p.pose = f(p.pose,u)
    ...:     particle_path.append(copy.deepcopy(particles))
    ...:
    ...: for i,p in enumerate(path):
    ...:     draw(path[i],particle_path[i])
    ...:     draw_landmarks(actual_landmarks)
    ...:     draw_observations(path[i],measurementss[i])
```

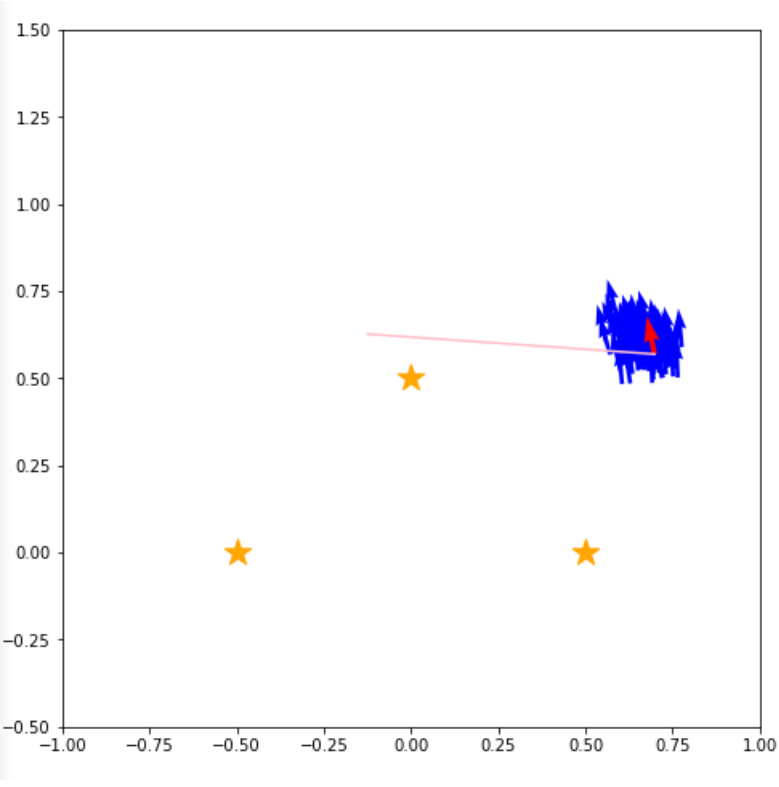

1.50
1.25
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00

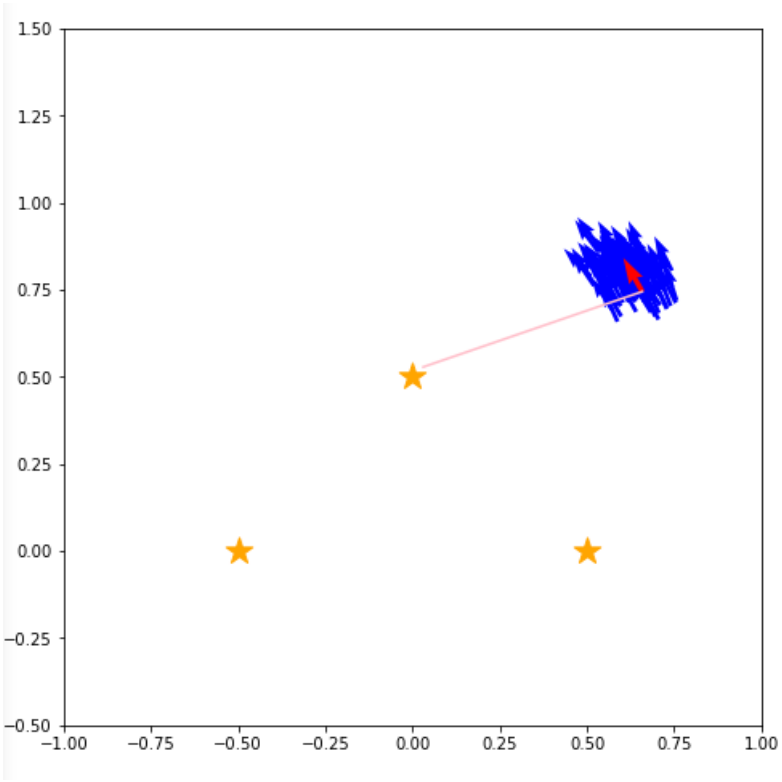

1.50
1.25
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00

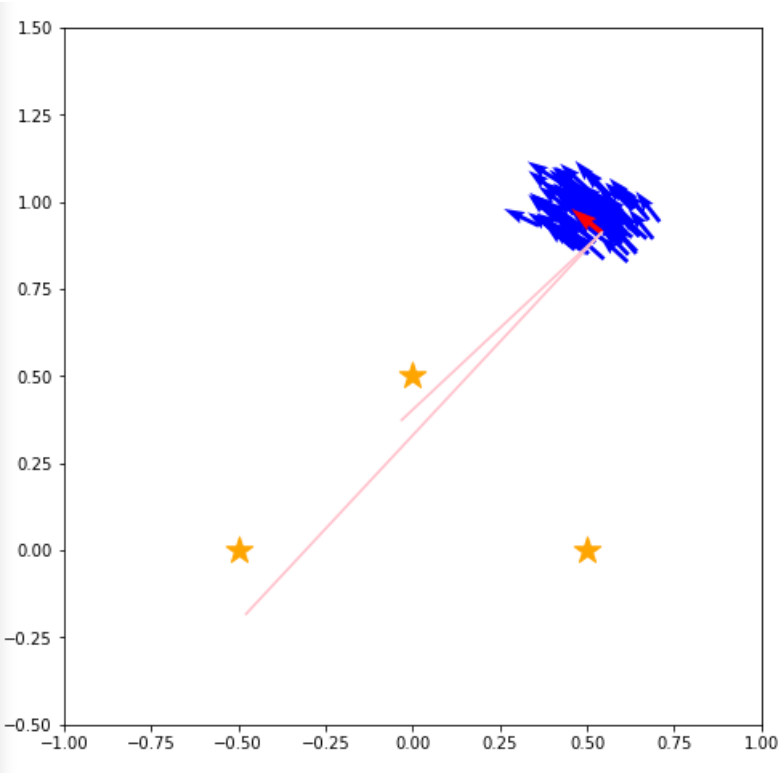

1.50
1.25
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00

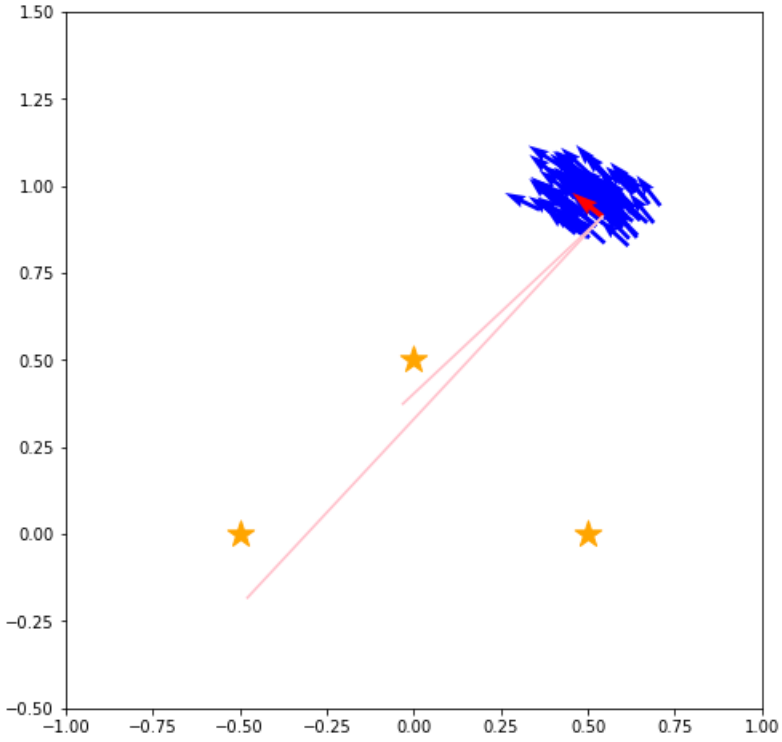

1.50
1.25
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00

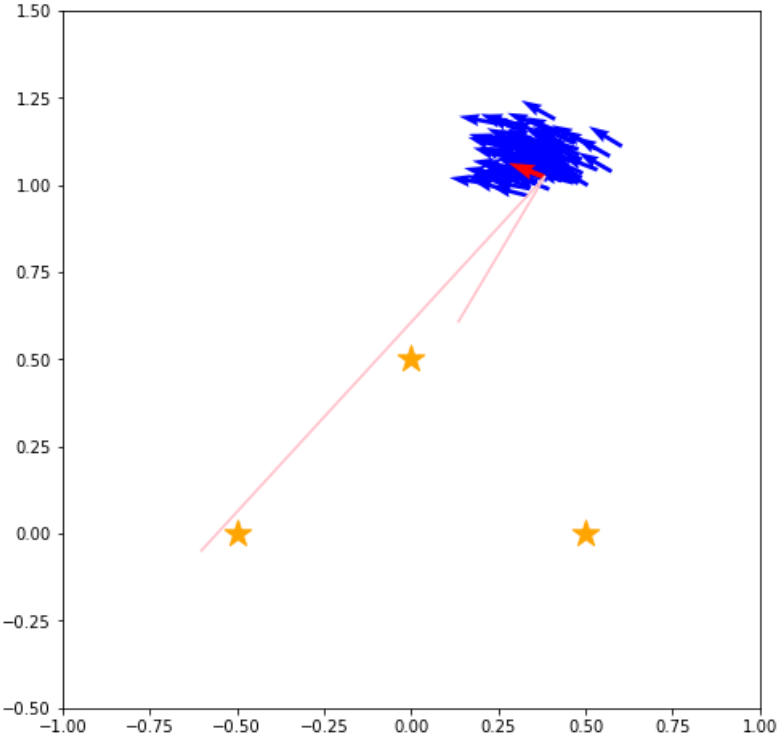

1.50
1.25
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00

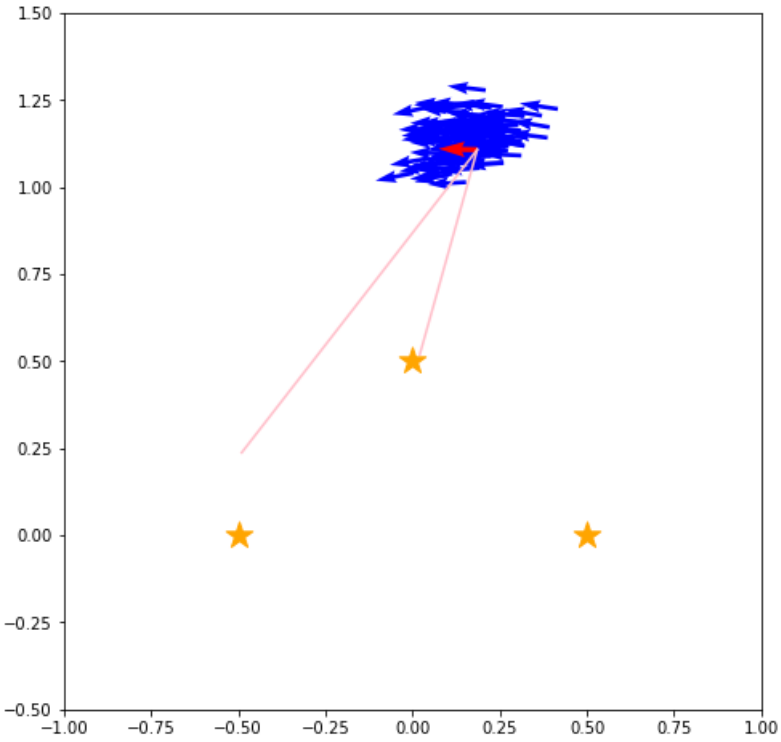

1.50
1.25
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00

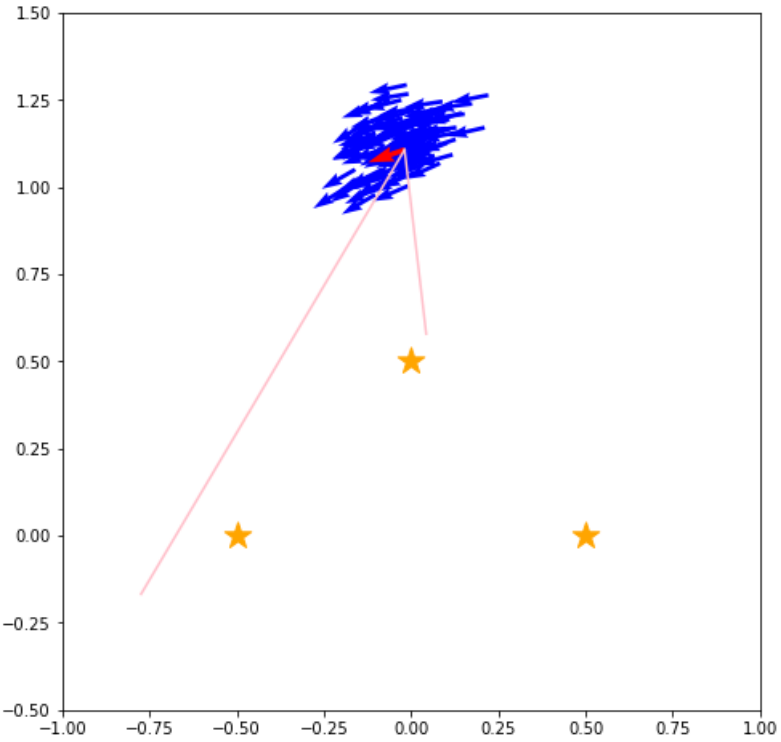

1.50
1.25
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00


1.50
1.25
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00


1.50
1.25
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00

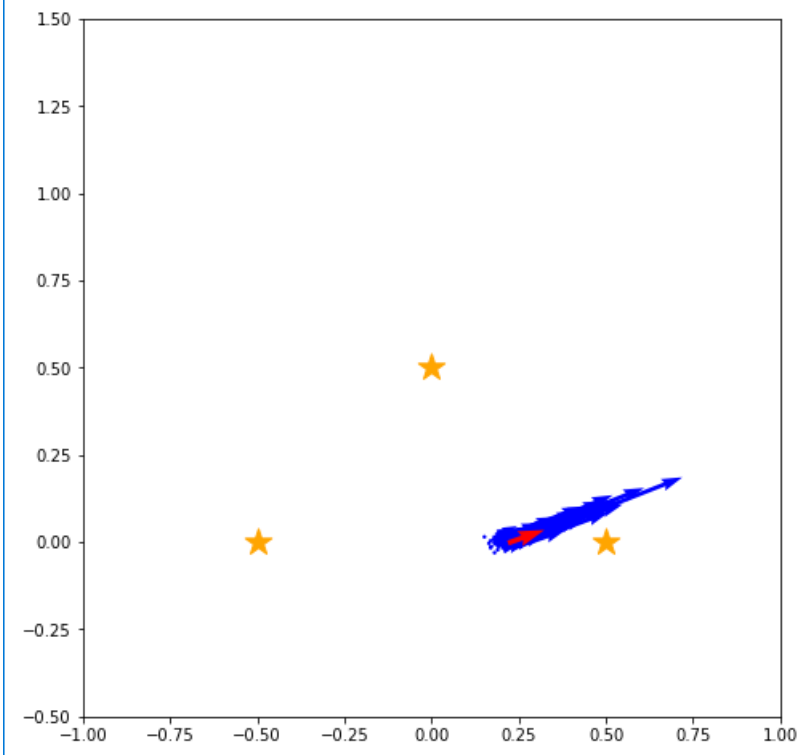

1.50
1.25
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00

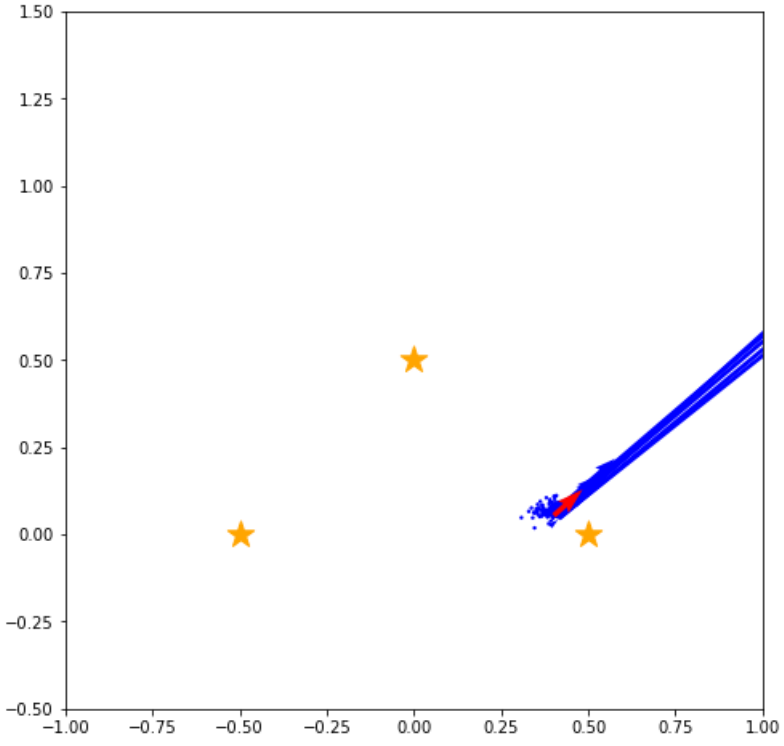

1.50
1.25
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00

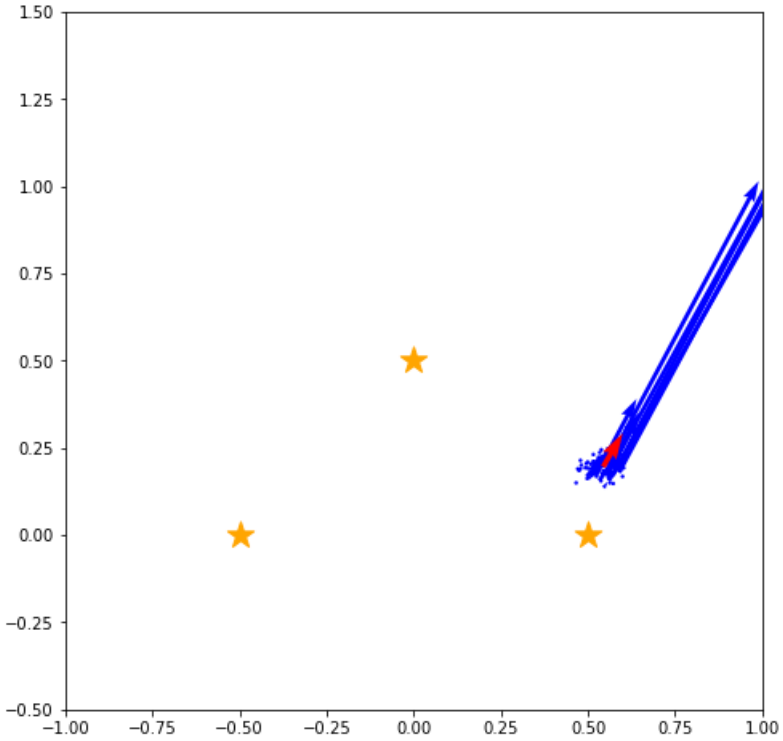

1.50
1.25
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00

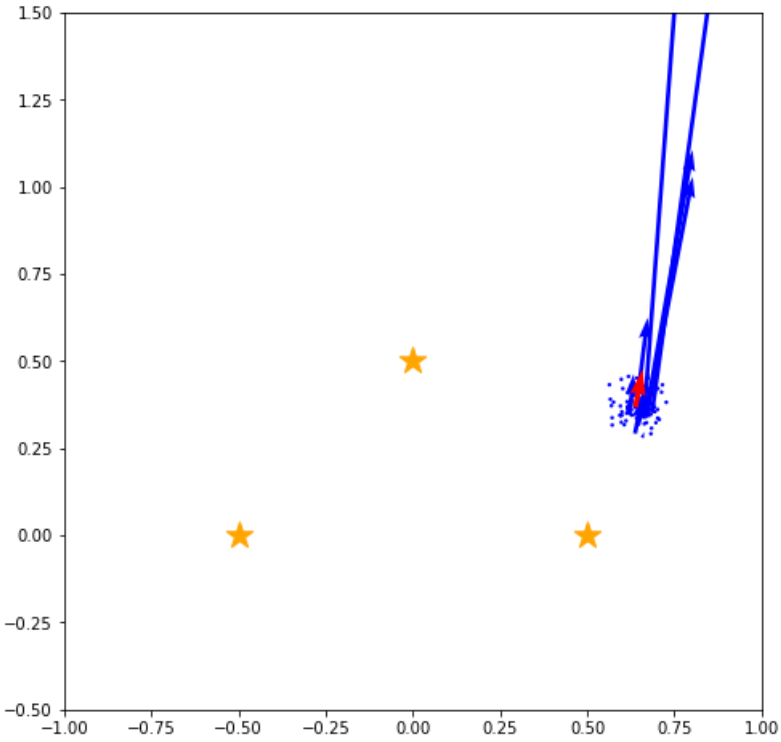
```
In [12]: from scipy.stats import norm  # ガウス分布（正規分布）のオブジェクトをインポート
    ...: def likelihood(pose, measurement):
    ...:     x,y,theta = pose
    ...:     distance, direction,lx,ly = measurement
    ...:
    ...:     # パーティクルの姿勢から観測されるはずのランドマークの距離と向き
    ...:     rel_distance, rel_direction, tmp_x,tmp_y = relative_landmark_pos(pose,(lx,ly))
    ...:
    ...:     # 誤差をガウスで評価
    ...:     return norm.pdf(x = distance - rel_distance, loc = 0.0, scale = rel_distance / 10.0)  \
    ...:             * norm.pdf(x = direction - rel_direction, loc = 0.0, scale = 5.0/180.0 * math.pi)
    ...:
    ...: ### パーティクル群の重みを変更する関数 ###
    ...: def change_weights(particles, measurement):
    ...:     for p in particles:
    ...:         p.weight *= likelihood(p.pose, measurement)
    ...:
    ...:     # 重みの合計を1に保つ
    ...:     ws = [ p.weight for p in particles ]
    ...:     s = sum(ws)
    ...:     for p in particles: p.weight = p.weight / s

In [13]: actual_x = np.array([0.0,0.0,0.0])   #ロボットの実際の姿勢
    ...: particles = [Particle(1.0/100) for i in range(100)]
    ...: u = np.array([0.2,math.pi / 180.0 * 20]) #ロボットの移動
    ...:
    ...: path = [actual_x]
    ...: particle_path = [copy.deepcopy(particles)]
    ...: measurementss = [observations(actual_x, actual_landmarks)]
    ...: for i in range(10):
    ...:     actual_x = f(actual_x,u)
    ...:     path.append(actual_x)
    ...:     ms = observations(actual_x,actual_landmarks)
    ...:     measurementss.append(ms)
    ...:
    ...:     for p in particles:
    ...:         p.pose = f(p.pose,u)
    ...:
    ...:     for m in ms:
    ...:         change_weights(particles, m)
    ...:     particle_path.append(copy.deepcopy(particles))
    ...:
    ...: for i,p in enumerate(path):
```

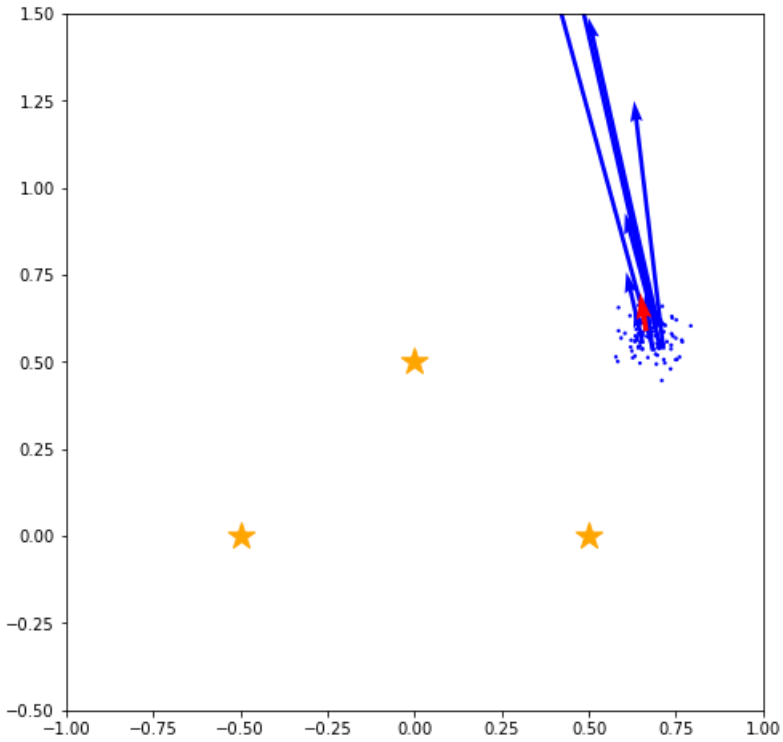
```
    ...:     draw(path[i],particle_path[i])
    ...:     draw_landmarks(actual_landmarks)
    ...:     draw_observations(path[i],measurementss[i])
```

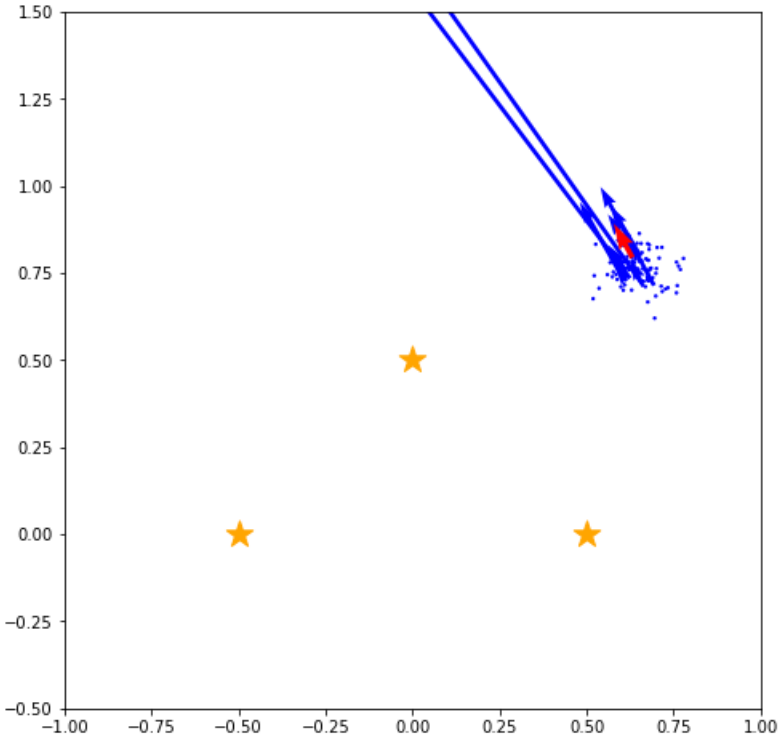

1.50
1.25
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00

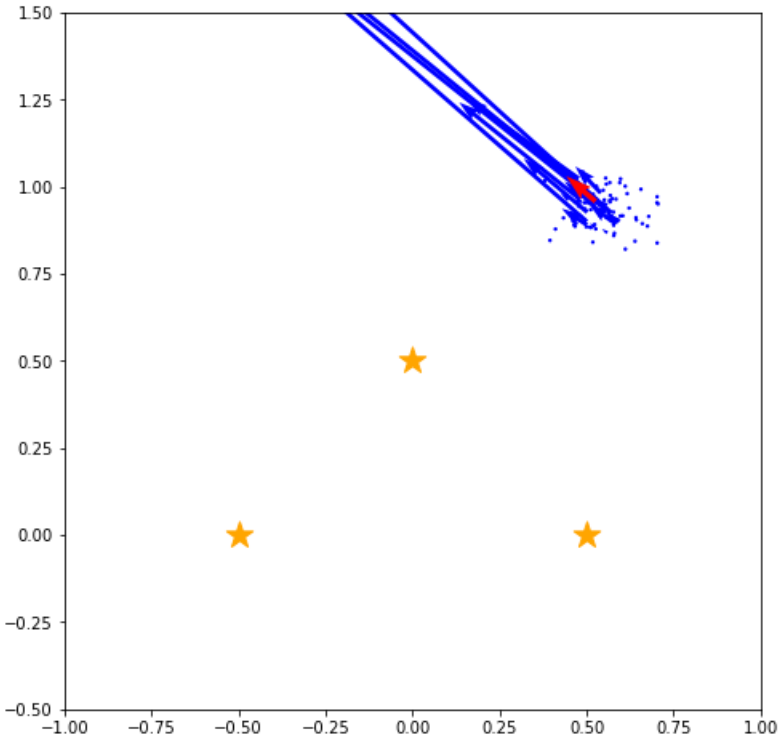

1.50
1.25
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00

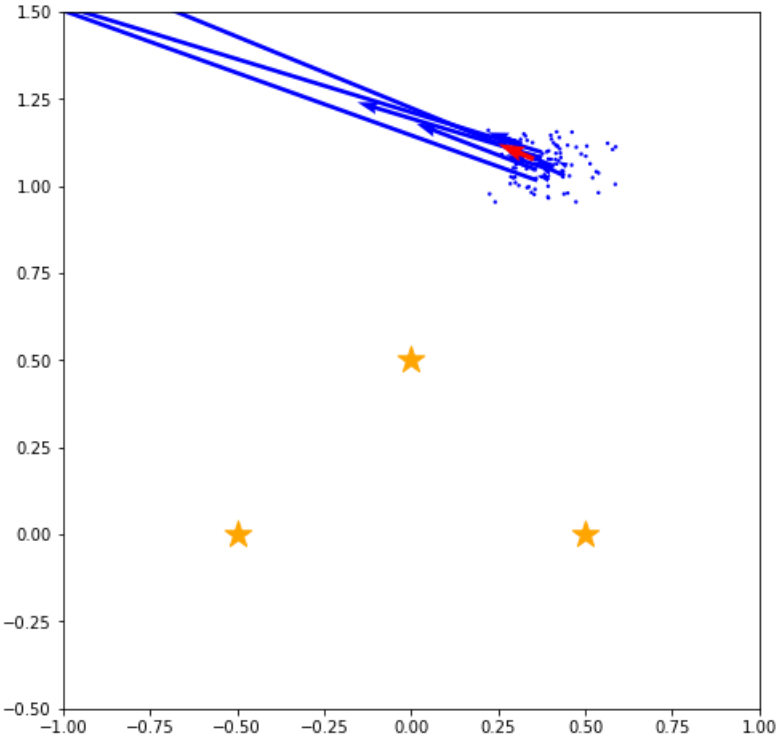

1.50
1.25
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00

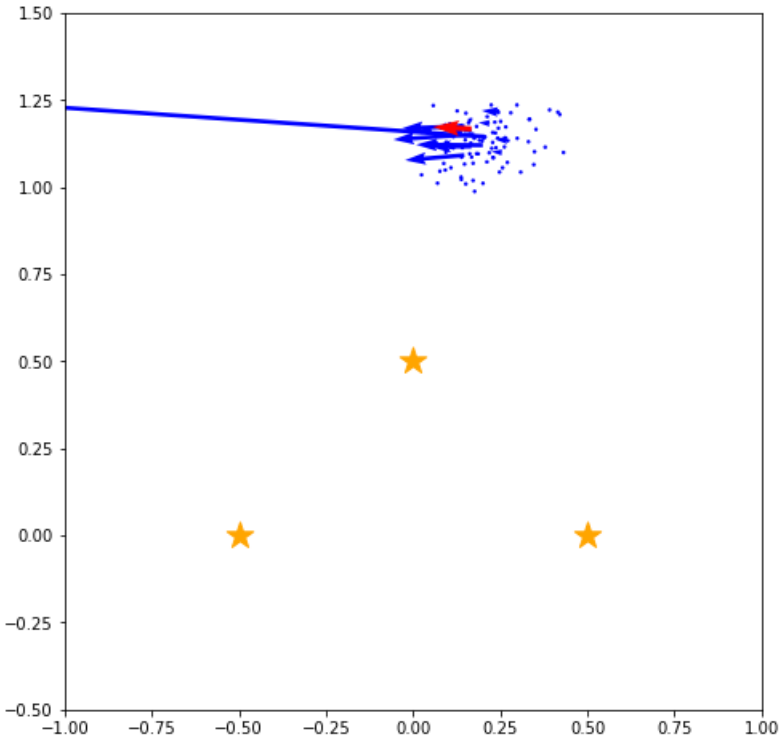

1.50
1.25
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00

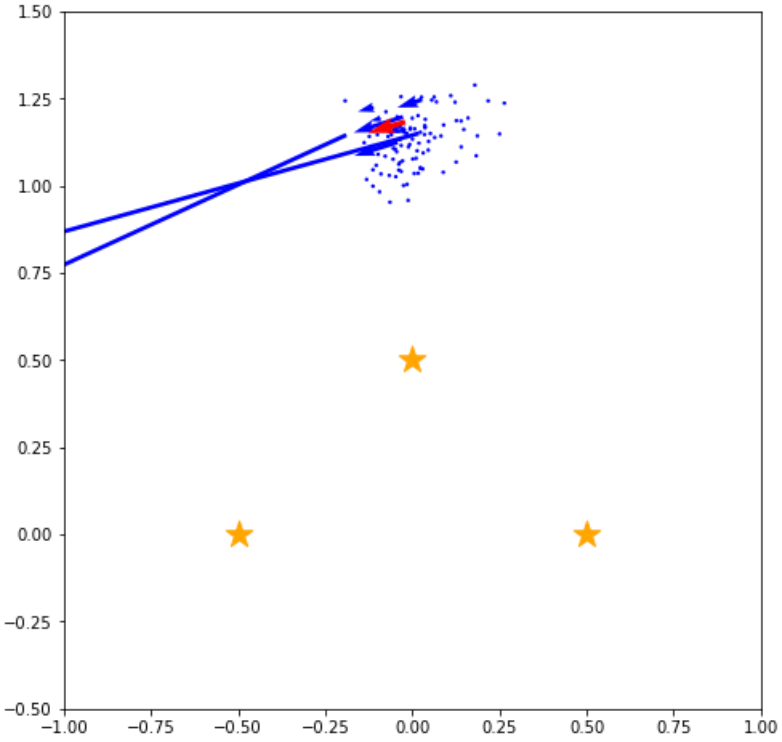

1.50
1.25
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00


1.50
1.25
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00


1.50
1.25
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00


1.50
1.25
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00


1.50
1.25
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00


1.50
1.25
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00

```
In [14]: class Particle:
    ...:     def __init__(self,p,w ):  #姿勢も指定できるようにしましょう
    ...:         self.pose = np.array(p)
    ...:         self.weight = w
    ...:
    ...:     def __repr__(self):
    ...:         return "pose: " + str(self.pose) + " weight: " + str(self.weight)

In [14]:

In [15]: for p in particles:
    ...:     print(p)
pose: [-0.05476359  1.10692165  3.59601041] weight: 5.7411538688e-13
pose: [ 0.0244286   1.05041836  3.58368082] weight: 1.9536943494e-19
pose: [-0.00583324  1.16574625  3.5532049 ] weight: 3.36134389678e-11
pose: [-0.04525744  1.0975496   3.5078176 ] weight: 9.05914031268e-13
pose: [ 0.0894455   1.17408742  3.38515905] weight: 0.000656630327731
pose: [ 0.12246137  1.23951214  3.27765244] weight: 4.48610415837e-06
pose: [-0.12160113  1.1023389   3.52093928] weight: 1.61958963704e-05
pose: [-0.06441491  0.95264128  3.66505825] weight: 8.69478148864e-31
pose: [ 0.04647053  1.12695917  3.39226882] weight: 0.000381333159858
pose: [ 0.02601559  1.25392287  3.32604974] weight: 2.7200701964e-06
pose: [-0.03501192  1.16887249  3.44214941] weight: 0.000239550287583
pose: [-0.04235128  1.12450017  3.44467012] weight: 0.0117466062461
pose: [-0.11318924  0.99864277  3.69736359] weight: 1.59669974247e-21
pose: [-0.09010489  1.03370588  3.58899269] weight: 6.17160623093e-15
pose: [ 0.11019313  1.25819101  3.3590285 ] weight: 1.29547017133e-07
pose: [-0.13151291  1.01807074  3.5860631 ] weight: 3.20698206407e-16
pose: [-0.18965072  1.14555783  3.57208257] weight: 0.107389522912
pose: [-0.19308273  1.24341458  3.55439971] weight: 4.63636341056e-09
pose: [-0.011317    1.11125954  3.50703921] weight: 0.000160444524461
pose: [ 0.02275531  1.15328588  3.47145222] weight: 1.94295087209e-05
pose: [-0.03972917  1.16345137  3.54194951] weight: 0.00328651184503
pose: [-0.10385744  1.0584909   3.46773514] weight: 3.63500940783e-07
pose: [ 0.13971217  1.18565971  3.28401844] weight: 2.14277418724e-05
pose: [-0.09069571  1.19316502  3.38680951] weight: 0.00287160432166
pose: [ 0.05558823  1.24555087  3.44389793] weight: 1.48563870886e-10
pose: [-0.09438581  0.98234153  3.71324632] weight: 9.1484223841e-29
pose: [ 0.0135653   1.10015307  3.62418761] weight: 6.65140881681e-17
pose: [-0.10050643  1.14646985  3.55812258] weight: 2.25482679324e-07
pose: [ 0.06242378  1.25427536  3.3085003 ] weight: 1.62304404764e-08
pose: [ 0.02212361  1.05093234  3.57390812] weight: 4.17191950096e-18
pose: [-0.02829461  1.07116397  3.58916071] weight: 1.10458337705e-07
```

```
pose: [ 0.02212361  1.05093234  3.57390812] weight: 4.17191950096e-18
pose: [-0.02829461  1.07116397  3.58916071] weight: 1.10458337705e-07
pose: [-0.01851169  1.16223934  3.46462509] weight: 0.00321039839794
pose: [ 0.1792381   1.28838462  3.23335609] weight: 8.06757611045e-15
pose: [ 0.03585434  1.16737274  3.60988504] weight: 1.18741573058e-15
pose: [-0.01625782  1.15034256  3.49839186] weight: 3.74497805861e-09
pose: [ 0.03704013  1.19021487  3.48395021] weight: 7.35019488271e-11
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.843573945771
pose: [ 0.00654218  1.16305665  3.51715836] weight: 0.000685009348479
pose: [-0.03556662  1.13223949  3.49705141] weight: 6.11603508059e-11
pose: [ 0.18323734  1.08588144  3.33553383] weight: 0.0
pose: [ 0.12541677  1.06768024  3.32076861] weight: 0.0
pose: [-0.0759425   1.21200222  3.49578956] weight: 4.5331449049e-07
pose: [ 0.03467637  1.14684904  3.35137167] weight: 6.10647057054e-06
pose: [-0.03279119  1.25528641  3.41899996] weight: 1.36175026151e-06
pose: [ 0.03602108  1.16712915  3.3578147 ] weight: 6.4531485181e-07
pose: [-0.0062987   1.16172919  3.47789989] weight: 3.47563617388e-05
pose: [-0.03921016  1.04473148  3.62714614] weight: 7.01379165545e-27
pose: [-0.05339222  1.16141276  3.58439781] weight: 0.000258961131177
pose: [-0.04778464  1.02589959  3.67597885] weight: 3.74554421015e-18
pose: [ 0.04543728  1.10208722  3.39462876] weight: 1.43308737845e-05
pose: [-0.06029394  1.14593815  3.58751641] weight: 4.69329824032e-06
pose: [-0.06627931  1.02928949  3.80821701] weight: 6.51002398072e-27
pose: [ 0.18633815  1.14292641  3.34021269] weight: 2.10993422218e-09
pose: [-0.07013011  1.14108031  3.59660534] weight: 6.34644722374e-07
pose: [ 0.01059382  1.07865574  3.64126949] weight: 2.55322404497e-19
pose: [-0.03217552  1.04116327  3.53455471] weight: 8.65428985764e-12
pose: [ 0.14761103  1.16120377  3.30276946] weight: 0.0
pose: [ 0.01873718  1.12193583  3.45720437] weight: 5.51910746507e-08
pose: [ 0.14007379  1.11114376  3.48221069] weight: 1.68433369505e-20
pose: [-0.02559934  1.00070484  3.56345009] weight: 7.80265521226e-19
pose: [ 0.01620536  1.20114874  3.48436991] weight: 0.00101749420162
pose: [-0.03914007  1.06140729  3.59536889] weight: 6.23826725085e-18
pose: [-0.09832014  1.08992294  3.6901751 ] weight: 3.57828144943e-11
pose: [ 0.15962009  1.19407811  3.46312683] weight: 6.48135270196e-17
pose: [ -4.34901322e-04   1.19364614e+00   3.49032952e+00] weight: 7.93878354336e-08
pose: [-0.01228211  0.95806009  3.61514737] weight: 3.15829250596e-31
pose: [-0.0781477   1.08456893  3.59600196] weight: 1.91294141989e-13
pose: [-0.03224292  1.11537039  3.50693704] weight: 1.76085734157e-05
pose: [-0.03600038  1.06610911  3.63278444] weight: 7.92638268071e-16
pose: [-0.04925884  1.10502067  3.58693994] weight: 2.28637917364e-10
pose: [ -2.06130597e-05   1.13378190e+00   3.47555935e+00] weight: 1.83497429449e-07
pose: [-0.12377558  1.14133286  3.65526754] weight: 6.24365285492e-16
pose: [ 0.08852443  1.03846592  3.59092531] weight: 5.24897183533e-29
```

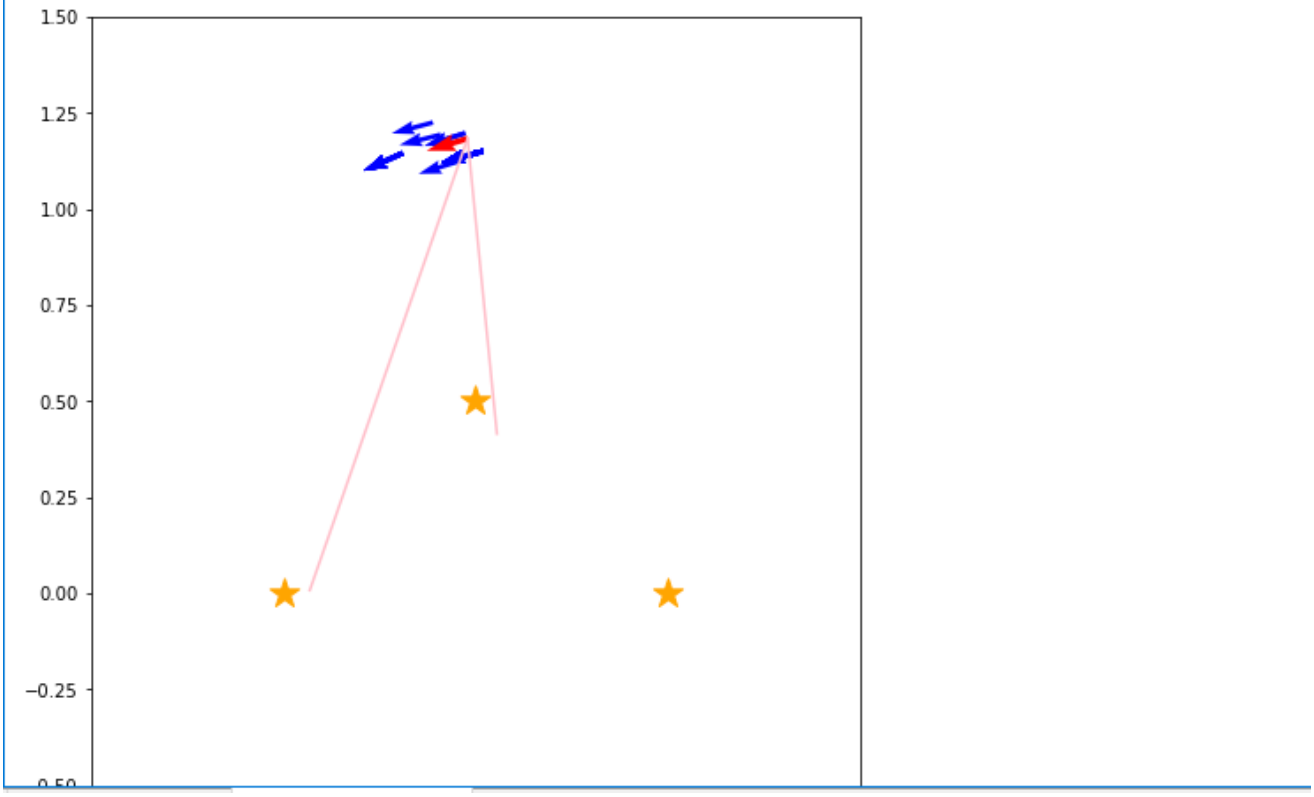
```
pose: [ 0.21730132  1.24319773  3.2758035 ] weight: 2.67802614116e-15
pose: [-0.13855433  1.12344791  3.54751022] weight: 5.72953019042e-07
pose: [-0.04258569  1.17047142  3.5797889 ] weight: 7.64746530977e-13
pose: [-0.00793619  1.11822811  3.52342329] weight: 5.56954014279e-19
pose: [-0.13451364  1.09478601  3.6082131 ] weight: 0.0
pose: [  1.02365489e-03   1.07659396e+00   3.49894359e+00] weight: 8.38287006982e-09
pose: [ 0.01676785  1.03578058  3.52669746] weight: 1.12369237407e-19
pose: [ 0.08190962  1.1426435   3.47123142] weight: 1.24052110141e-15
pose: [-0.04381077  1.1975565   3.46374502] weight: 4.91781377377e-11
pose: [-0.02708127  1.19732543  3.44872685] weight: 0.0133591759613
pose: [ 0.00520981  1.03501973  3.51967541] weight: 3.44581106834e-20
pose: [ 0.26373837  1.23684383  3.22954542] weight: 1.52941060429e-17
pose: [ 0.0122797   1.24088285  3.47623743] weight: 1.5186739192e-16
pose: [-0.11326216  1.04563504  3.5670833 ] weight: 1.67994979404e-09
pose: [-0.01524923  1.00401923  3.59616316] weight: 6.40172341762e-24
pose: [-0.05546758  1.07572031  3.40285694] weight: 6.28466038424e-06
pose: [ 0.06900334  1.24033925  3.44956075] weight: 7.10039344556e-16
pose: [ 0.03091876  1.09380195  3.54674156] weight: 4.83798748775e-15
pose: [ 0.06403839  1.13818626  3.51388499] weight: 1.34179135982e-10
pose: [ 0.02962597  1.04799515  3.48308777] weight: 0.0
pose: [ 0.03537532  1.17220432  3.50719304] weight: 1.73677558083e-13
pose: [ 0.24998154  1.14836172  3.22775234] weight: 1.28258997097e-10
pose: [-0.0229741   1.15252859  3.4477315 ] weight: 2.30615188402e-05

In [16]: accum = []
    ...: sm = 0.0
    ...: for p in particles:
    ...:     accum.append(p.weight + sm)
    ...:     sm += p.weight
    ...:
    ...: print(accum)
[5.7411538687960873e-13, 5.741155822490437e-13, 3.4187554550053313e-11, 3.5093468581321374e-11, 0.00065663036282408887, 0.00066111646698245596, 0.00067731236335281117, 0.00067731236335281117,
0.0010586455232112209, 0.0010613655934076254, 0.0013009158809905511, 0.013047522127057514, 0.013047522127057514, 0.013047522127063687, 0.013047651674080819, 0.013047651674080819, 0.12043717458600946,
0.12043717922237286, 0.12059762374683364, 0.12061705325555452, 0.12390356510058095, 0.12390392860152173, 0.12392535634339411, 0.12679696066505217, 0.12679696081361605, 0.12679696081361605, 0.12679696081361611,
0.12679718629629544, 0.1267972025267359, 0.1267972025267359, 0.1267973129850736, 0.13000771138301687, 0.13000771138302494, 0.13000771138302614, 0.13000771512800419, 0.13000771520150614, 0.97358166097241383,
0.9742666703208932, 0.9742666703820535, 0.9742666703820535, 0.9742666703820535, 0.974267123696544, 0.97427323016711453, 0.97427459191737609, 0.97427523723222786, 0.97430999359396675, 0.97430999359396675,
0.97456895472514393, 0.97456895472514393, 0.97458328559892848, 0.97458797889716886, 0.97458797889716886, 0.97458798100710309, 0.97458861565182542, 0.97458861565182542, 0.97458861566047972, 0.97458861566047972,
0.97458867085155432, 0.97458867085155432, 0.97458867085155432, 0.97560616505317677, 0.97560616505317677, 0.97560616508895959, 0.9756061650889597, 0.97560624447679511, 0.97560624447679511, 0.97560624447698641,
0.97562385305040211, 0.97562385305040289, 0.97562385327904078, 0.97562403677647025, 0.97562403677647092, 0.97562403677647092, 0.9756240381521929, 0.97987921751859264, 0.97987923511115205, 0.98661089453105411,
0.98661089453105677, 0.9866114674840758, 0.98661146748484052, 0.98661146748484052, 0.98661146748484052, 0.98661147586771059, 0.98661147586771059, 0.98661147586771181, 0.98661147591688991, 0.99997065187820833,
0.99997065187820833, 0.99997065187820833, 0.99997065187820844, 0.99997065355815828, 0.99997065355815828, 0.9999769382185425, 0.99997693821854317, 0.99997693821854805, 0.99997693835272716, 0.99997693835272716,
0.9999769383529008, 0.99997693848115976, 0.99999999999999989]
```

```
In [17]: pointer = random.uniform(0.0,1.0/len(particles))
    ...: print(pointer)
0.006788874556447383

In [18]: new_particles = []
    ...: particles_num = len(particles)
    ...:
    ...: while pointer < 1.0:
    ...:     if accum[0] >= pointer:
    ...:         new_particles.append(
    ...:             Particle(copy.deepcopy(particles[0].pose),1.0/particles_num)
    ...:         )
    ...:         pointer += 1.0/particles_num
    ...:     else:
    ...:         accum.pop(0)
    ...:         particles.pop(0)

In [19]: particles = new_particles
    ...:
    ...: for p in particles:
    ...:     print(p)
pose: [-0.04235128  1.12450017  3.44467012] weight: 0.01
pose: [-0.18965072  1.14555783  3.57208257] weight: 0.01
pose: [-0.18965072  1.14555783  3.57208257] weight: 0.01
pose: [-0.18965072  1.14555783  3.57208257] weight: 0.01
pose: [-0.18965072  1.14555783  3.57208257] weight: 0.01
pose: [-0.18965072  1.14555783  3.57208257] weight: 0.01
pose: [-0.18965072  1.14555783  3.57208257] weight: 0.01
pose: [-0.18965072  1.14555783  3.57208257] weight: 0.01
pose: [-0.18965072  1.14555783  3.57208257] weight: 0.01
pose: [-0.18965072  1.14555783  3.57208257] weight: 0.01
pose: [-0.18965072  1.14555783  3.57208257] weight: 0.01
pose: [-0.18965072  1.14555783  3.57208257] weight: 0.01
pose: [-0.09069571  1.19316502  3.38680951] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
```

```
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
```

```
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [ 0.01833412  1.15011915  3.41117134] weight: 0.01
pose: [-0.11105312  1.22492206  3.40199646] weight: 0.01
pose: [-0.02708127  1.19732543  3.44872685] weight: 0.01
pose: [-0.02708127  1.19732543  3.44872685] weight: 0.01
```

```
In [20]: path = [actual_x]
    ...: particle_path = [copy.deepcopy(particles)]
    ...: measurementss = [observations(actual_x, actual_landmarks)]
    ...: for i in range(10):
    ...:     actual_x = f(actual_x,u)
    ...:     path.append(actual_x)
    ...:     ms = observations(actual_x,actual_landmarks)
    ...:     measurementss.append(ms)
    ...:
    ...:     for p in particles:
    ...:         p.pose = f(p.pose,u)
    ...:
    ...:     for m in ms:
    ...:         change_weights(particles, m)
    ...:     particle_path.append(copy.deepcopy(particles))
    ...:
    ...: for i,p in enumerate(path):
    ...:     draw(path[i],particle_path[i])
    ...:     draw_landmarks(actual_landmarks)
    ...:     draw_observations(path[i],measurementss[i])
```



（↑1 回分を表示しています）