ISTANBUL TECHNICAL UNIVERSITY

COMPUTER ENGINEERING

# Assignment 2

*Artificial Intelligence (BLG 521E)*

*Dec 19th, 2021*

Ahmet Kemal Yetkin

504201506

# Problem 1 - Localization with Particle Filter

1. **Describe what each particle represents (sometimes also called a hypothesis).**

   Particles represent the likely places that this robot can be in. The initial robot location is unknown. Therefore, all locations of the environment are equally likely to contain the robot. Initially all particles are distributed. Then, whenever robot moves in the environment, these likely locations will be represented by different particle sets.

2. **Describe how the quality of a particle is measured (probability of it being true).**

   There are two phases for measuring the particles quality.
   - Dynamics update (also called as motion update)
   - Observation update

   Dynamics update and observation update are continually applied for tracking an unobservable variable state in the environment. Initially we have many likely places in the environment. Whenever agent gets new observation continually, these particles which are inconsistent will disappear for the robot and finally the robot will be able to detect its own position. Since those particles will be disappeared, the remain particles will be the measure of quality.

3. **Briefly explain your usage of particle filter to solve this problem, give an example scenario of what would happen from start (unknown initial position of robot) to end (the robot position is determined, only a single particle left).**

   Every particle in impact provides a guess as to the location of the robot in the particle filter. The initial robot location is unknown. Therefore, all locations of the environment are equally likely to contain the robot. Initially all particles are distributed. Then, whenever robot moves in the environment, these likely locations will be represented by different particle sets. When the agent gets observations in the environment, the particles likelihood changes. Agent will be more certain about its current position and the particles will take place in that particular location.

   Let’s examine the solution on an example scenario:

   If the entire map was as shown in the figure, the laser would have scanned the entire map and therefore the robot would be able to determine its direct position with the laser data without starting its movement. However, since the given figure is not representing the all map, we can assume that there can be a similar room on the map. (Figure 1)

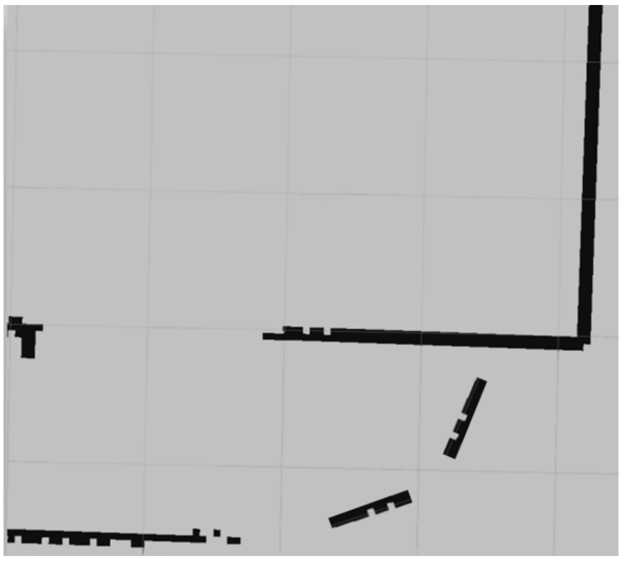


*Figure 1. Similar Room*

Since the location of the robot is unknown at first, it can be in any particle on the map. The robot starts its movement by receiving the laser data. With the laser data, the number of likely locations can be on is reduced. (Figure 2)

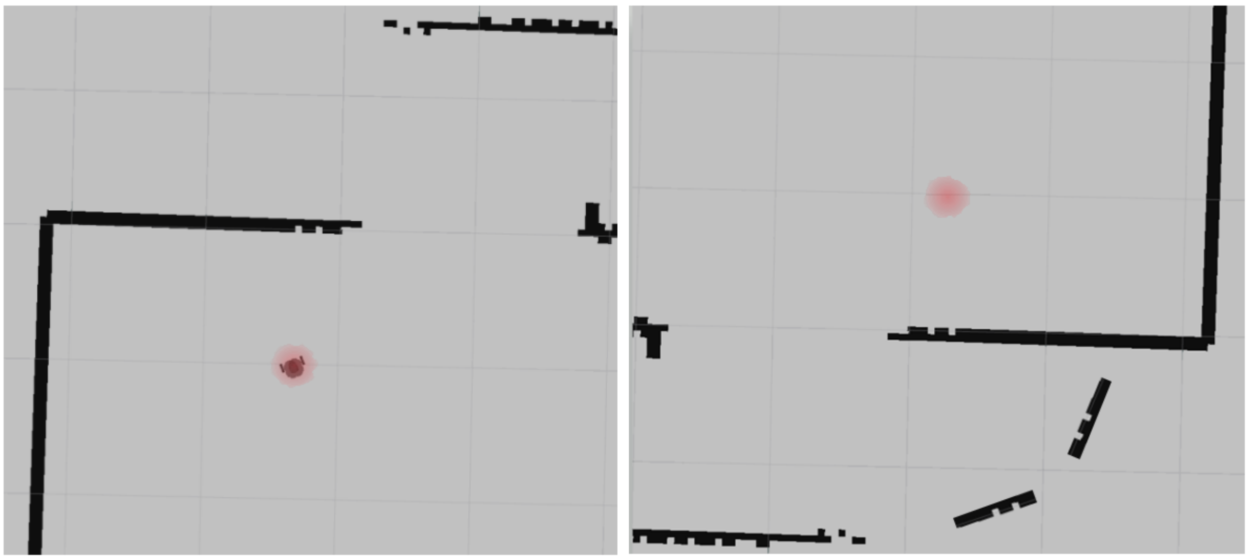


*Figure 2. Likely locations*

Then, the robot will continue its movement (dynamics update) and whenever it moves on the environment, it will gather the data by observation update. Let’s assume that the robot moves to the left. The likely location will be still the same since there is still no difference for particle sets. (Figure 3)

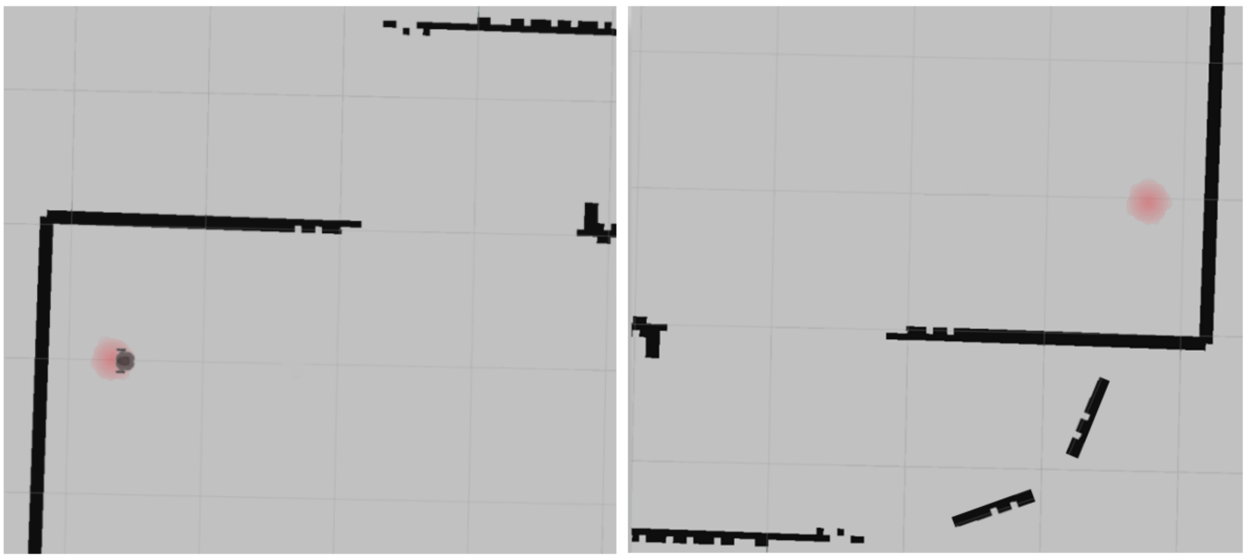


*Figure 3. Likely locations after movement*

As long as the robot dynamics continues to update, it will continue to receive observations. However, since there are some obstacles separating the two areas, the robot will be able to locate itself. (Figure 4)

It has been shown that the laser beams continue without hitting any obstacle.

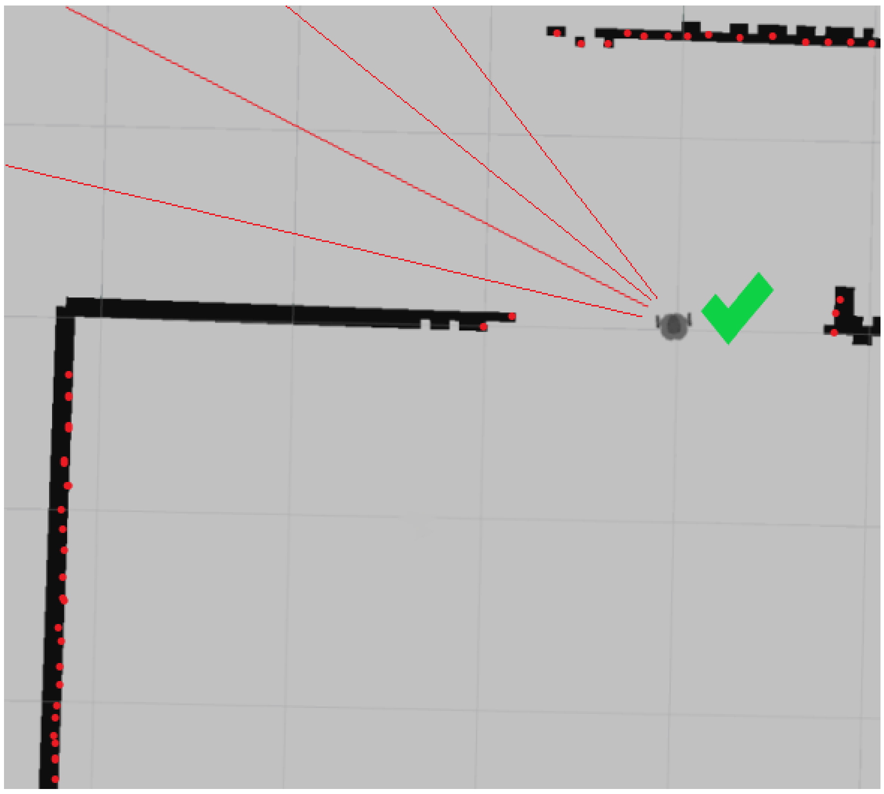


*Figure 4. Only a single particle left*

4. **(BONUS) Give pseudocode of your particle evaluation function. Hint: You need to make use of the world map, the laser scanner data of the current time and the possible location of the robot.**

```
observationUpdate():
    list[] ← compareWeights(Weigths,PreviousWeights)
    locations.append(list)
    return locations // tuple list of likely locations


ParticleFilterAlgorithm():

    while (isLocated ≠ True):
        generateDistanceByLaserScanner() // generate particle set according
                                            to obstacle and angle of laser beam
        observation ← observationUpdate()
        resample(Weigths,AllParticles)

        if length(observationUpdate()) == 1:
           isLocated ← True

        else if length(observationUpdate()) > 1:
           dynamicsUpdate()//Path selection and movement according to obstacle
                             expansion

        return observation
```

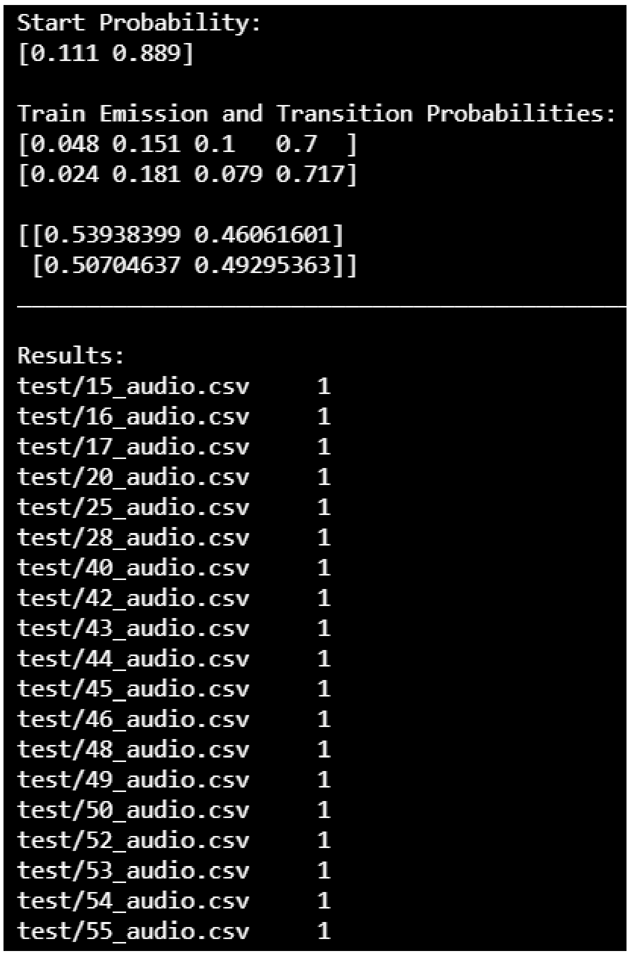
## Problem 2 - Application of Methods in Problem Domains

*Please download libraries before executing the file. If you have already those libraries you can ignore this and continue for analysis. Libraries: Pomegranate, Pandas, and NumPy.*

First of all, we collect all the data to be trained in a single array. However, we keep the failed and not failed files from the label files in separate arrays for the emission probability display. We use the total number of failed and not failed files in the file to be trained to calculate the start probability.

Since emission probability could not be drawn from the algorithm, it was calculated manually. While making this calculation, the ratio of the anomaly amounts of the class types to the total amount is taken as a basis. For the algorithm used, the total length of the data must be equal. Therefore, this condition is met by adding "None" values. (None has been added to have a different value from the 1,2,3, and 4 type classes in order not to disturb Bayes.)

These values were then used to train the model. After the model was trained, test data were drawn and predictions were made on these data.



```
Start Probability:
[0.111 0.889]

Train Emission and Transition Probabilities:
[0.048 0.151 0.1   0.7  ]
[0.024 0.181 0.079 0.717]

[[0.53938399 0.46061601]
 [0.50704637 0.49295363]]
____________________________________________

Results:
test/15_audio.csv     1
test/16_audio.csv     1
test/17_audio.csv     1
test/20_audio.csv     1
test/25_audio.csv     1
test/28_audio.csv     1
test/40_audio.csv     1
test/42_audio.csv     1
test/43_audio.csv     1
test/44_audio.csv     1
test/45_audio.csv     1
test/46_audio.csv     1
test/48_audio.csv     1
test/49_audio.csv     1
test/50_audio.csv     1
test/52_audio.csv     1
test/53_audio.csv     1
test/54_audio.csv     1
test/55_audio.csv     1
```

*Figure 5. Output of Algorithm*

The biggest reason for the high number of failed files as a result of the estimation is that most of the data we use while training our model consists of failed data.

Running command:

```
python BLG521E_HW_2_504201506.py
```