Warmup questions

1. For what kinds of maps is it especially helpful to have higher MAX_RANGE?

2. If your robot might move to a random location sometimes, how can we model that with our particle filter and recover?

3. Why might you want to add particles in random locations even if you know your robot will never be kidnapped?

How to run the particle filter:

1. Download sork4.stanford.edu/pf.zip
Put it in My Documents\Matlab
Right-cick "pf" and choose "Extract All..."
Keep pressing "next"

2. Open MatLab

3. In the command window, type:
cd pf/pf

Parameters are defined at the top of the file "pf.m"
To run the particle filter, just type "pf" in the command window
If you want to cancel it while it's running, type ctrl-c in the command window
Let's use MAP_NUMBER = 1 for the first experiments

Lab questions

Easy:

1. Try experimenting with N (number of particles). What happens when you don't have enough particles?

Ans:
For N = 100 → Average LMS error = 28.2365
For N = 300 → Average LMS error = 22.8844
For N = 500 → Average LMS error = 10.7256
Increasing the number of particles results in a low LMS error, while decreasing it to 100 particles leads to a high LMS error.

2. Try experimenting with different MAX_RANGE values. See how short values reduce the ability of the particle filter to distinguish between different locations.

For N = 20 → Average LMS error = 24.3536

For N = 50 → Average LMS error = 19.8304

For N = 10 → Average LMS error = 38.2824

If we increasing the max_range = 50 error is low. We can see from the middle figure the particles are very far to point .

3. Try experimenting with different MEASUREMENT_NOISE values. (I suggest at first you always set MEASUREMENT_NOISE2 to be twice your chosen value of MEASUREMENT_NOISE.) Notice how much worse your tracking gets when the noise increases.

For Measurement_noise = 4,
Measurement_noise = 8 → Average LMS error = 25.5944

For Measurement_noise = 4,
Measurement_noise = 32 → Average LMS error = 43.4056

Higher sensor noise may lead to a broader distribution of particles, making it more challenging for the filter to converge to the correct state.

More interesting:

4. Do you think you can compensate for higher measurement noise with a sensor that can see farther? Try it!

For Max range = 60, Measurement_noise = 4,
Measurement_noise = 32 → Average LMS error = 21.551
Increasing the **MAX_RANGE** while keeping the noise levels constant results in a lower LMS value, and the particles exhibit a higher degree of precision as they become more attracted to the robot. This leads to more accurate tracking when doubling the **MAX_RANGE** to 60 compared to when it is set to 30.

5. What happens when your assumed motion noise is much less than the actual motion noise? Try setting MOTION_NOISE = 1 and MOTION_NOISE2 = .3
Do you see what's wrong with the particle filter now when your robot's speed changes quickly?

`Average lms error: 7.7925`
Failure to capture rapid changes, lack of particle diversity and difficulty in recovering errors

6. Now try the opposite, where MOTION_NOISE = .3 and MOTION_NOISE2 = 2
With the assumed motion noise is much more than the actual motion noise, how does the particle filter behave differently from before?
Incresed predictive uncertainity, greater particle diversity, improved handfling of rapid changes, robustness to modelling errors
`lms error: 13.784`

7. What do you think is worse: underestimating the motion noise or overestimating it?
Robot accuracy decreases when its mobility exceeds motion noise2, as the robot is unable to adjust to its surroundings.

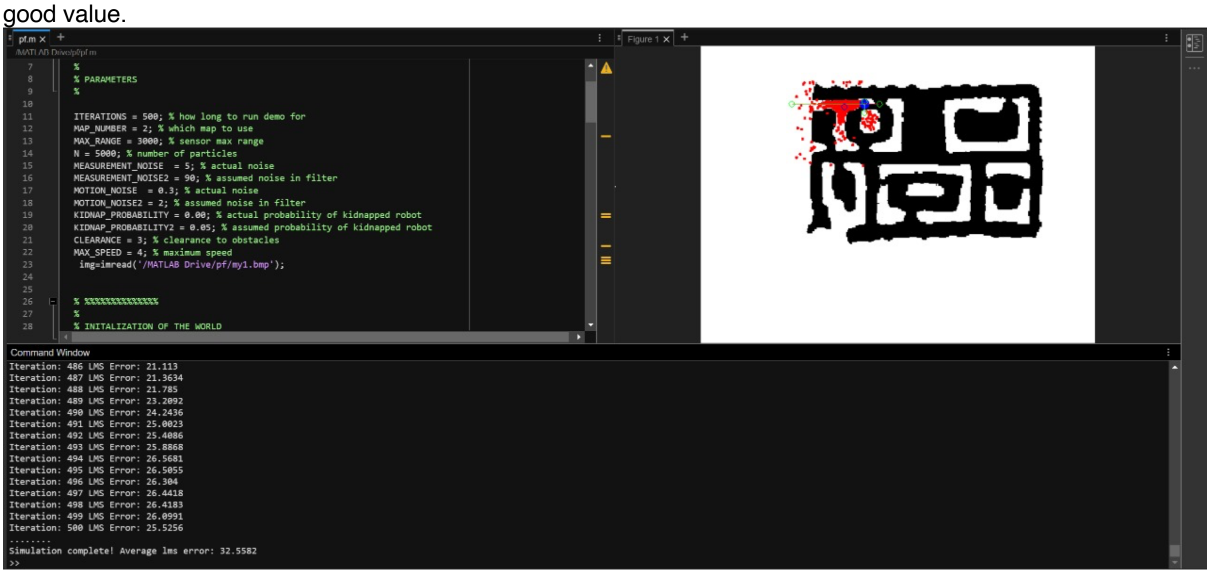
Excessive estimation

Robotic motion is less than motion noise2, which causes it to become extremely cautious and unable to determine its true state.

The worst, in my opinion, is underestimating the motion noise.

Since the robot doesn't care as much about the surroundings while it is moving faster—that is, if it overestimates—it may avoid a trap rather than accelerating with insufficient environmental information when there is a hole or trap in front of it.

8. Set MEASUREMENT_NOISE = 5 and try different values of MEASUREMENT_NOISE2. What happens when the assume measurement noise is too small? What happens when it's much larger than reality? Explain why using too low a value is bad, explain why using too large a value is bad, and make an opinion about how to pick a good value.



The key is to strike a balance by setting `MEASUREMENT_NOISE2` to a value that accurately represents the uncertainty in sensor measurements. This balance ensures that the particle filter is both responsive to observations and robust in the face of noise.