Particle Filter (just intuitions)

1. Model the location of the object with a particle at that location.

Since we know the object's location probabilistically, let's scatter $N$ particles at most likely initial loc. and then compute a weight for each particle based on its agreement with motion models & measurement.

2. At any time, the particles capture my belief about the object locations are distributed over space.

3. How

4. Move $N$ particles based on velocity dist. $P(v)$

5. Obtain sensor measurement

6. Update weight based on measurement error distribution $P(m|s)$

7. Redistribute low weight particles to the locations of existing particles in proportion to probabilities.

8. Lose velocity distribution $P(v)$

9. Further...
Sing: mines

But more particles where belief is greater.

Apply motion model $p(v)$ on each particle and repeat

Observe:

Always $N$ particles in the system. Scattering of $N$ particles captures my current belief. Particles are like magnifying lens (or clones) → you allocate more particles to places/trajectories that you want to search with high resolution.

Observe:

① P.F. operating on continuous space.
② Any kind of motion model can be incorporated.
③ Noise can be modeled.
④ $S_k = f(S_{k-1}) + \mathcal{N}$ any $f$ and $g$ possible.
⑤ CPU load is function of $N$, so you can choose $N$ based on system resources.

Kalman Filter: When $f$ and $g$ are linear AND noise is gaussian → simple algorithm in that case.
Main goal:
Can I compute both the sensor location and the visible objects' locations?

\[
\langle L_1, L_2, L_3, L_4 \rangle \\
\langle L_{10}, L_{11} \rangle
\]

Continuos pictures of objects
Motion model of sensor

Basically, I want:

\[
P(S_{1:m}^{(c)} | \overline{S}_{1:n}^{(o)} \mid w_{1:n})
\]