Particle Filter Walkthrough

July 29, 2020

1 What is a particle filter?

A particle filter tracks a target by maintaining a collection of particles, each representing one guess as to the target's true position.

On each measurement, the filter resamples the particles based on how closely each particle matches the measurement.

On each motion, the filter updates each particle according to the motion model.

2 A simple example

Let's use a particle filter to track a simple target.

Our target object is located in a one-dimensional space (in other words, it only has an x coordinate), starting at zero, and moves a fixed distance on every timestep.

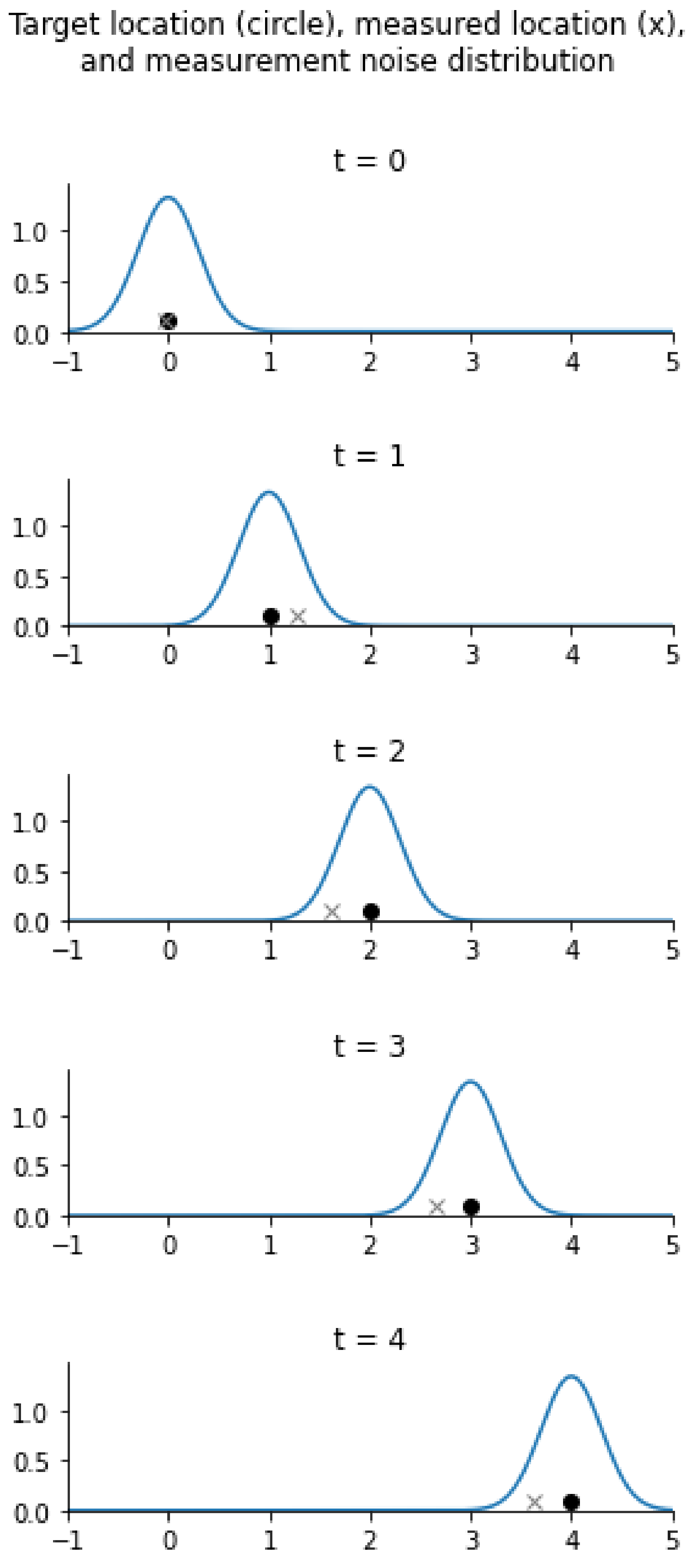
The target's *motion* is completely noise free, meaning that it moves exactly the same amount on every timestep without over- or under-shooting.

The targets *measurement* includes Guassian noise. This noise is parameterized by $\sigma_{measurement}$, with larger values resulting in noisier measurements.

In our example, we will use these values:

- $\sigma_{measurement}$: 0.3
- starting location: 0.0
- movement per timestep: 1.0
- # steps to plot: 5

This example is so simple that it does not really require a particle filter to track it, but it will allow us to step through the particle filter algorithm.

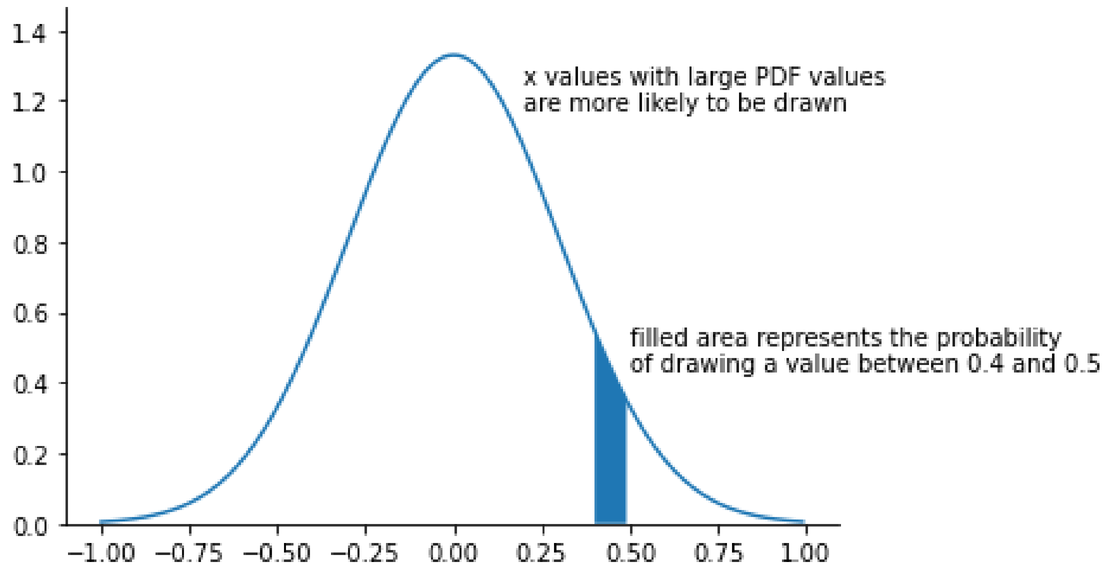

Target location (circle), measured location (x),
and measurement noise distribution
t = 0
t = 1
t = 2
t = 3
t = 4
1.0
0.5
0.0
−1
0
1
2
3
4
5

3 What is a probability density function?

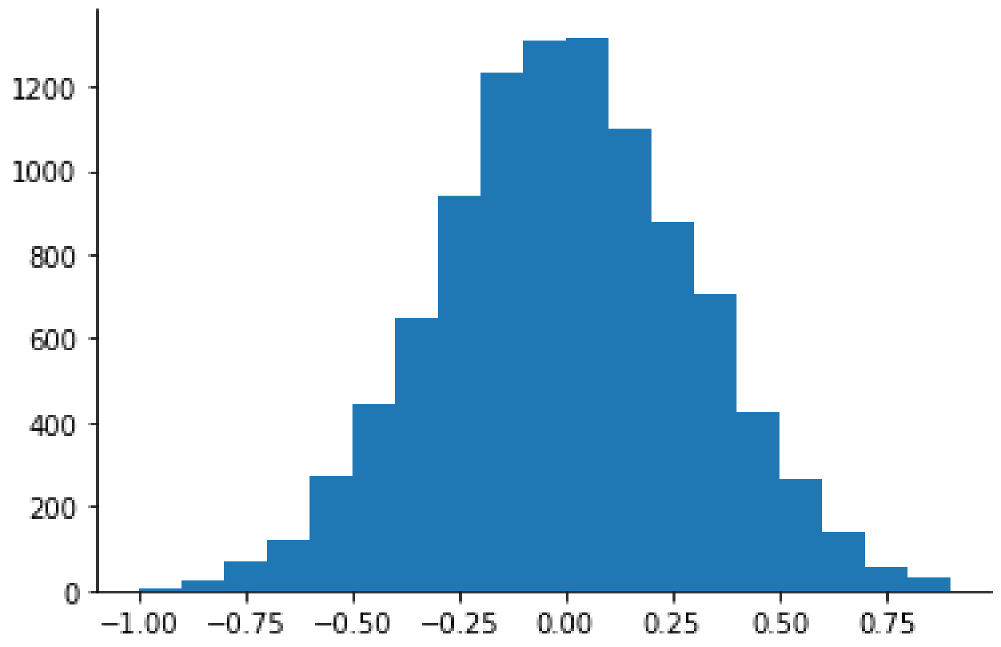
The Gaussian functions plotted above are *probability distribution functions*. PDFs describe weights for sampling values from the real number line. The greater the value of the PDF at a given x value, the likelier that x value is to be sampled.

Formally, the likelihood that a value in the range $[x_0, x_1]$ is drawn is equal to the area under the PDF in that range. Since probability distributions must sum to 1, this means that the area under every PDF is 1.

In the case of the Gaussian, note that is non-zero even for very large x values. This means that with a Gaussian distribution with $\mu = 0$ and $\sigma = 1$, it's still technically possible (but extremely unlikely) to draw a very large value such as 1,000.



As a demonstration, we draw 10,000 samples from the distribution, then plot them as a histogram with bins of size 0.1. Note how the histogram approximates the PDF function, given we take enough samples.

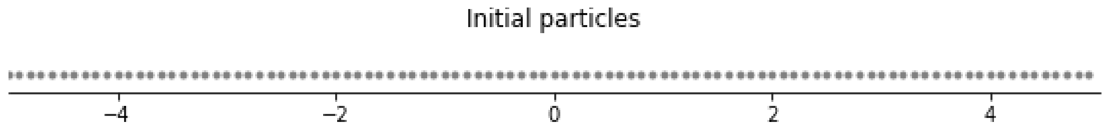


4 Generating our initial particles

Before getting the first measurement, we know nothing about the target's current location, except perhaps for a bounding area where it is guaranteed to be.

For this exercise, let's start by assuming the target is somewhere in the range $[-5, 5]$.

If we have no measurements, every value in that range is equally likely, so we generate particles which evenly cover the space.



5 The particle filter cycle

On each timestep we do the following:

- *weight* our existing particles based on the latest measurement
- *resample* our particles according to their weights

- optionally *fuzz* our particles by adding a small amount of noise to each
- *move* our particles according to the motion model

6 Resampling after the first measurement

The first measurement is our first piece of information about the target's location. We update our model by resampling our current set of particles.

To resample, we define our own weighting function to calculate how likely each particle is to represent the target's true location.

We have some flexibility defining this function, but in general it should:

- give large weights particles near the measured location, to improve our estimate
- give small weights (or zero weight) to particles far from the measured location

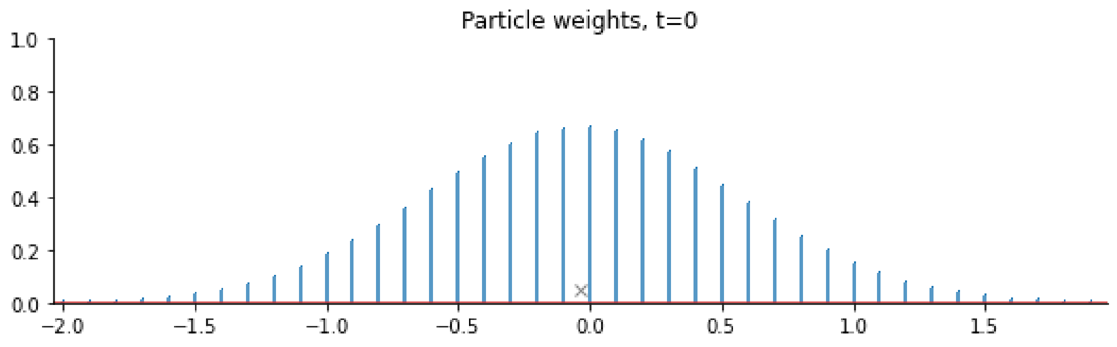
What counts as "near" and "far" depends on how noisy the measurements are.

As an example, below are two possible weighting functions:

- a Gaussian PDF centered on the measurement with $\sigma_{weighting} > \sigma_{measurement}$
- a uniform distribution within $w > \sigma_{measurement}$ of the measurement

Using a wider, flatter PDF than the original measurement noise makes our model more robust to outlier measurements.

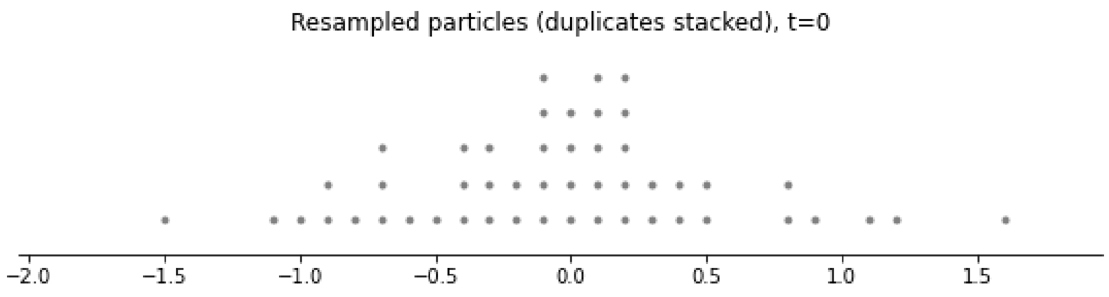
For the rest of this document, we will use the Gaussian PDF with $\sigma_{weighting} = 2\sigma_{measurement}$.



Our weight distribution is centered on the measurement, giving higher weights to particles close to the measurement and lower weights to particles farther away.

Note that the weight distribution has the same general shape as the measurement noise but is substantially wider and flatter.

Using these weights, we take a weighted sample from our current set of particles to generate a new, rebalanced set of particles. Each particle may be sampled zero, one or multiple times. Particles with high weights are likely to be sampled more than once, while particles with low weights may not be sampled at all.



7 Fuzzing the particles

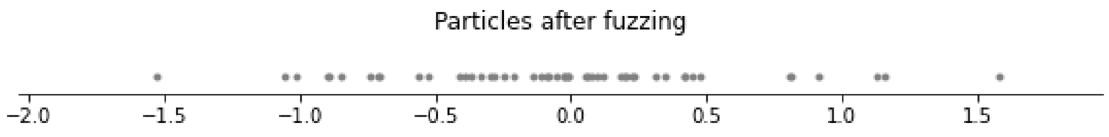
”Fuzzing” here means adding a small amount of noise to some or all particles. This does two things:

- makes duplicate samples slightly different from each other
- moves some particles closer to the true position (but also moves others farther away)

Fuzzing is not strictly neccessary in a particle filter and requires tuning if used. Details to consider are:

- how much noise to add
- how many particles to fuzz

In this example, we add noise sampled from a uniform distribution over $[-0.05, 0.05]$ to every particle.

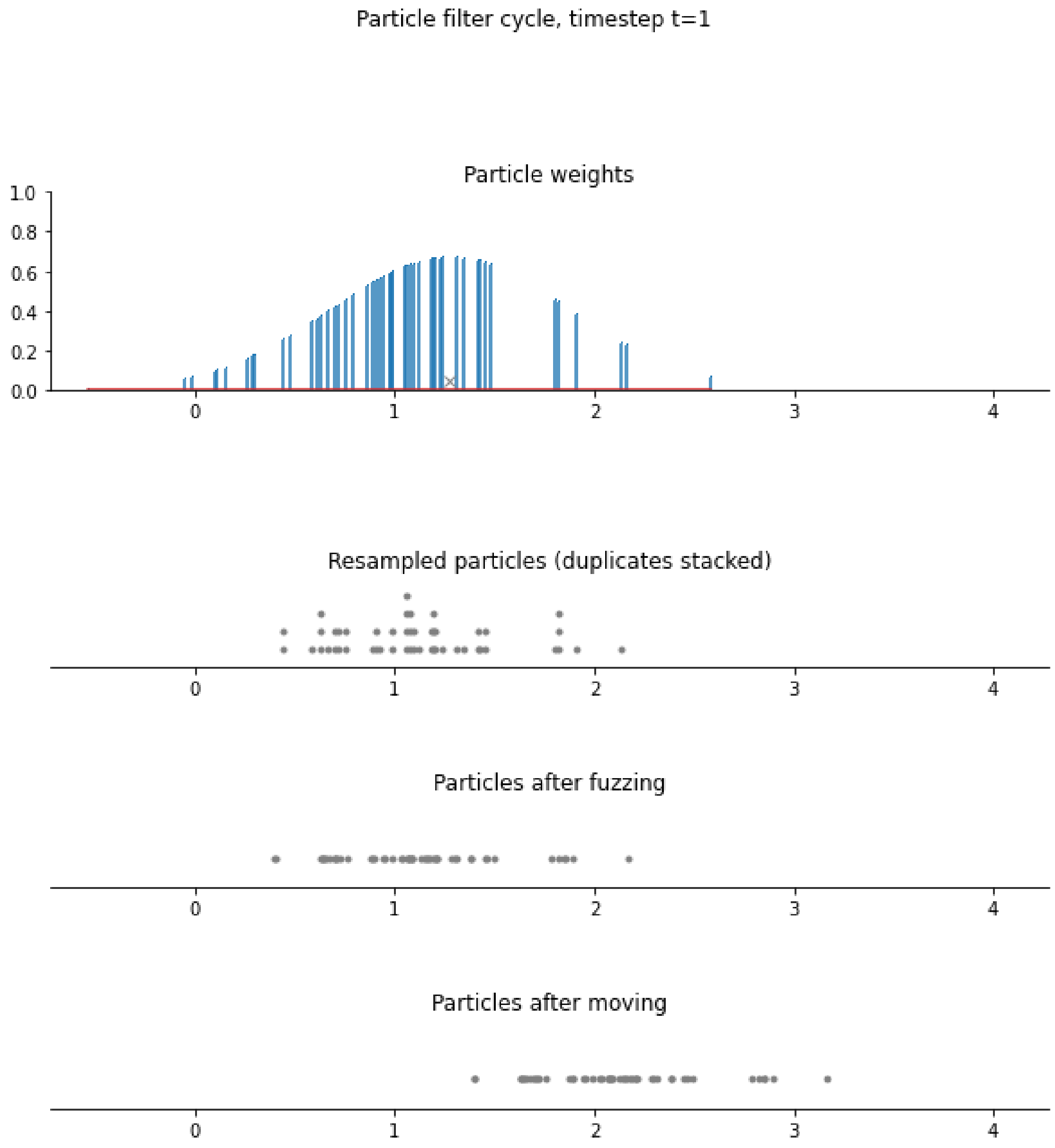


8 Moving the particles

We move the particles by applying the motion model. In this example, this means adding 1.0 to each particle.

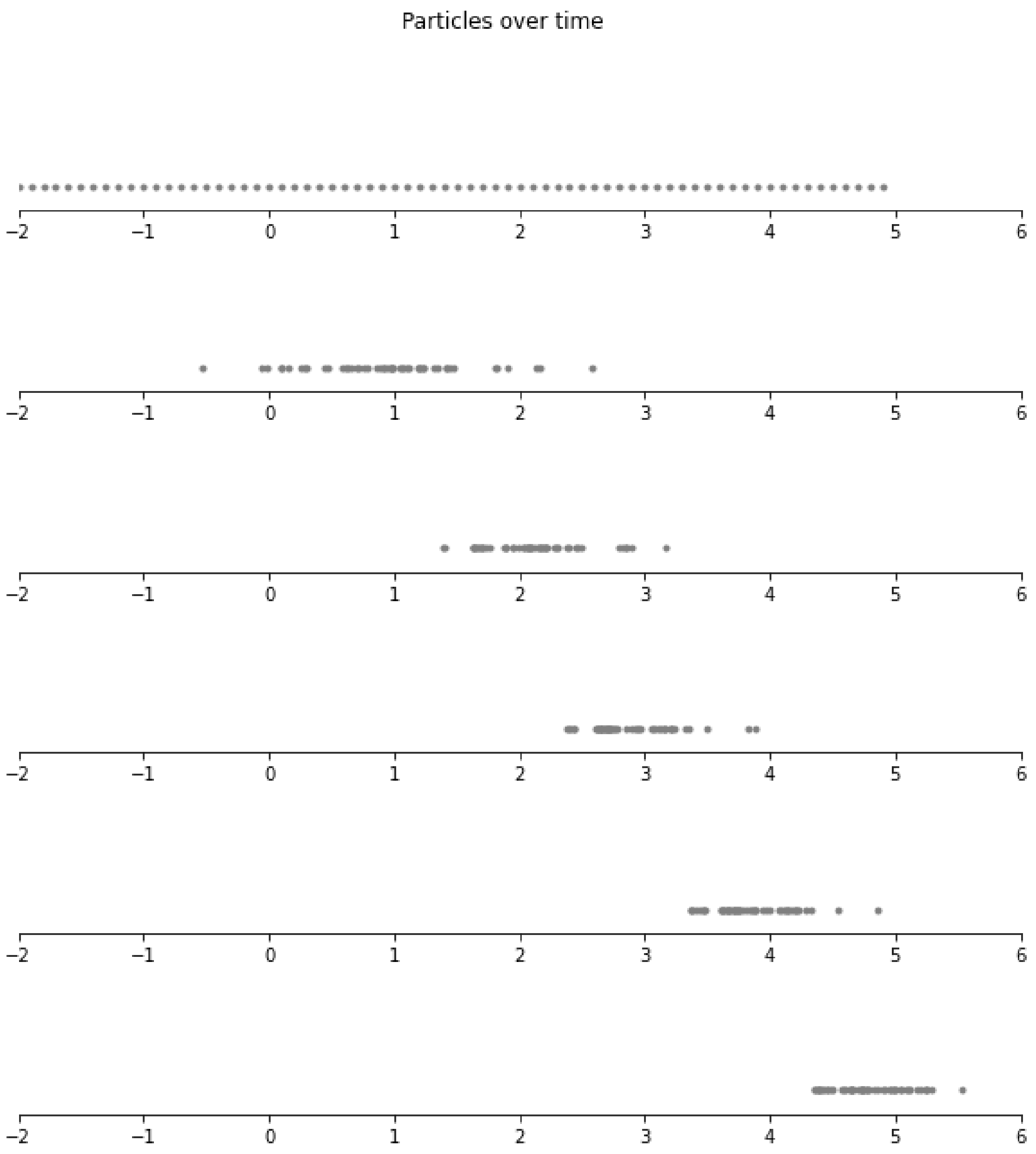
9 Repeat the cycle

For the next measurement, we repeat the weight-resample-fuzz-move cycle.



10 Convergence over time

Next, we run the particle filter through the remaining timesteps and observe the particles on each step. Note how the set of particles becomes more compact with each timestep.



11 Estimating location from particles

Some applications require that we convert the particle set into a single estimate of the target's location. As with weighting and fuzzing, there are is some flexiblity in deciding how to aggregate the particles into one location. Two possibilities are:

- averaging all particles
- randomly selecting one particle