Recursive Estimation

Lecture 10
The Particle Filter (PF)

ETH Zurich

Spring 2021

Date compiled: May 7, 2021

Learning Objectives

Topic: The Particle Filter (PF)

Objectives

- You can explain the technique of *Monte Carlo sampling* and how it relates to the Particle Filter (PF).
- You can explain the *derivation of the PF*.
- You can *design and implement* a PF for a general nonlinear system and general noise distributions (today: prior update).
- You can assess for *what types of problems* a PF is suitable.

The Particle Filter

The *Particle Filter* (PF) is an approximation of the Bayesian state estimator for a general nonlinear system and general noise distributions.

Basic idea: approximately represent the state PDF by a (large) number of samples, which are called *particles*.

Essentially, in a region where the PDF is large, there is a large number of particles.

The basic recursive update mechanism is as follows:

- the particles are propagated through the process model,
- the particles are then weighted according to the measurement likelihood,
- a resampling step generates a new set of particles.

Outline

Model

We consider the nonlinear discrete-time system:

$$x(k) = q_{k-1}(x(k-1), v(k-1)), \quad k = 1, 2, \dots$$
$$z(k) = h_k(x(k), w(k)),$$

where $x(0), \{v(\cdot)\}$ and $\{w(\cdot)\}$ are mutually independent and can be discrete or continuous random variables with known PDFs.

We introduce the PF for a system without input $u(k-1)$, but the discussion directly extends to a system with an input (the known input $u(k-1)$ can be absorbed in the explicit time dependency of $q_{k-1}(\cdot)$).

Outline

Monte Carlo Sampling

Monte Carlo sampling refers to the basic technique of using a large number of samples (called particles) to approximate the PDF of a random variable.

It is beneficial, for example, for computing an approximation of the PDF of a function of the random variable: instead of using change of variables formulas, the function is simply evaluated at the particles.

Monte Carlo sampling is the underlying idea that is used in the PF.

Discrete Random Variables (1/3)

Consider a discrete random variable y, that takes values in $\mathcal{Y} = \{1, 2, 3, \ldots, \overline{Y}\}$, and its associated PDF p_y.

The aim is to approximate p_y by a large number of random samples from y.

We model these samples by the collection of N DRVs, $\{y^1, y^2, \ldots, y^N\}$, which are mutually independent and identically distributed according to the PDF p_y.

It is convenient to represent the N random samples of y with the DRVs s_i^n $(n = 1, 2, \ldots, N)$, which take the value 1 if $y^n = i$ and take the value 0 if $y^n \neq i$ $(i = 1, 2, \ldots, \overline{Y})$. These are thus defined as:

$$s_i^n := \begin{cases} 1 & \text{if } y^n = i, \\ 0 & \text{otherwise,} \end{cases}$$

$$i = 1, \ldots, \overline{Y}, \quad n = 1, \ldots, N.$$

Discrete Random Variables (2/3)

In a more compact form, we write:

$$s_i^n = \delta(i - y^n),$$

where δ is the Kronecker delta, defined as

$$\delta(x) = \begin{cases} 1 & \text{if } x = 0, \\ 0 & \text{otherwise.} \end{cases}$$

For a fixed i, the DRVs s_i^n, $n = 1, 2, \ldots, N$, are mutually independent and identically distributed.

Using the law of the unconscious statistician we can calculate the expected value of these DRVs:

$$\mathrm{E}\left[s_i^n\right] = \sum_{\bar{y}^n=1}^{\bar{Y}} \delta(i - \bar{y}^n) p_y(\bar{y}^n) = p_y(i)\,.$$

Discrete Random Variables (3/3)

Let the DRV s_i denote the average of s_i^n over $n = 1, 2, \ldots, N$,

$$s_i := \frac{1}{N} \sum_{n=1}^{N} s_i^n ,$$

which corresponds to the fraction of the s_i^n that take the value i.

By the *law of large numbers* (LLN) the average converges[1] to the expected value of s_i^n; that is,

$$s_i \xrightarrow{N \to \infty} \mathrm{E}\left[s_i^n\right] = p_y(i) \,.$$

We denote the values that the random variables s_i^n take by $\bar{s}_i^n$. We can therefore approximate the PDF of y with

$$\boxed{p_y(i) \approx \frac{1}{N} \sum_{n=1}^{N} \bar{s}_i^n .}$$

[1] In the sense of almost sure convergence, i.e. converging with probability 1.

Example (1/2)

Let the DRV y be the outcome of a fair coin toss, with $y = 1$ if the outcome is heads, and $y = 2$ if the outcome is tails.

We model three repeated coin tosses by the random variables y^1, y^2, and y^3, which are all independent and take values 1 (for heads) and 2 (for tails) with equal probability.

The random variables s_i^n, $i = 1, 2$ (for heads and tails), $n = 1, 2, 3$ (for the different tosses), are defined as above:

$$s_1^n := \delta(1 - y^n), \quad s_2^n := \delta(2 - y^n).$$

Example (2/2)

We throw the coin three times and the outcome is heads, heads, and tails.

Therefore the random variables s_i^n take the values,

$$\bar{s}_1^1 = 1, \bar{s}_2^1 = 0, \quad \bar{s}_1^2 = 1, \bar{s}_2^2 = 0, \quad \bar{s}_1^3 = 0, \bar{s}_2^3 = 1,$$

and we obtain,

$$p_y(1) \approx 1/3(\bar{s}_1^1 + \bar{s}_1^2 + \bar{s}_1^3) = 2/3,$$
$$p_y(2) \approx 1/3(\bar{s}_2^1 + \bar{s}_2^2 + \bar{s}_2^3) = 1/3,$$

as a (crude) approximation of $p_y(1)$, respectively $p_y(2)$.

By taking more and more samples, i.e. repeating the coin toss multiple times, the approximation converges to the true PDF.

Change of Variables (1/2)

Now consider a function $x = g(y)$, with $x \in \mathcal{X} := g(\mathcal{Y})$.

We have already discussed in class how to construct p_x from p_y (change of variables formula).

Now, we are interested in *approximating* p_x by a set of samples.

We define $x^n := g(y^n)$ and $j \in \mathcal{X}$ to be any value x can take and introduce:

$$r_j^n := \begin{cases} 1 & \text{if } x^n = g(y^n) = j, \\ 0 & \text{otherwise,} \end{cases}$$

which can be expressed in short hand notation as $r_j^n = \delta(j - x^n)$.

Change of Variables (2/2)

Analogous to the previous section, we can approximate p_x by:

$$p_x(j) \approx \frac{1}{N} \sum_{n=1}^{N} \bar{r}_j^n = \frac{1}{N} \sum_{n=1}^{N} \delta(j - \bar{x}^n) = \frac{1}{N} \sum_{n=1}^{N} \delta(j - g(\bar{y}^n)),$$

that is, by sampling y.

Instead of solving the change of variables, we simply need to evaluate the function $g(\cdot)$ at all samples from y.

Summary

For $y \in \mathcal{Y}$, it holds:

$$p_y(i) \approx \frac{1}{N} \sum_{n=1}^{N} \delta(i - \bar{y}^n), \quad i \in \mathcal{Y}.$$

For $x = g(y)$, with $x \in \mathcal{X} := g(\mathcal{Y})$, it holds:

$$p_x(j) \approx \frac{1}{N} \sum_{n=1}^{N} \delta(j - g(\bar{y}^n)), \quad j \in g(\mathcal{Y}).$$

Joint discrete random variables

The above is also true for joint discrete random variables,

$$p_x(\xi) \approx \frac{1}{N} \sum_{n=1}^{N} \delta(\xi - \bar{x}^n), \quad \forall \xi,$$

where ξ and $\bar{x}^n$ are now vectors, and $\delta(\cdot)$ refers to the vector version of the Kronecker delta (that is, δ is one if all entries of its vector argument are zero, otherwise δ is zero).

Moreover, we have only considered finite $\mathcal{X}$ and $\mathcal{Y}$, but the above holds for the infinite case as well.

Continuous random variables (1/5)

Consider the continuous random variable y with PDF p_y.

As before, we wish to approximate p_y, which will be done by binning the CRV y and applying the above sampling procedure.

Let Δy be the fixed bin size (can be generalized to non-fixed bin sizes) and let the CRVs y^n, $n = 1, 2, \ldots, N$ be independent and identically distributed with PDF p_y.

The CRVs y^n will be used to model the samples from y. We define the DRVs s_a^n as:

$$s_a^n := \begin{cases} 1 & \text{if } a \leq y^n < a + \Delta y, \\ 0 & \text{otherwise,} \end{cases}$$

where a is used to address the different bins (it takes discrete values spaced by Δy).

This can be expressed as $s_a^n = \int_a^{a+\Delta y} \delta(\xi - y^n)\, d\xi$, where, in this case, δ is the Dirac delta pulse.

Continuous random variables (2/5)

Analogous to the cases before, the expected value is given by,

$$\mathrm{E}\left[s_a^n\right] = \int_{-\infty}^{\infty} \left[\int_a^{a+\Delta y} \delta(\xi - \bar{y}^n)\, d\xi \right] p_y(\bar{y}^n)\, d\bar{y}^n,$$

where the inner integral is 1 for all $\bar{y}^n \in [a, a+\Delta y]$ (and 0 otherwise), so that,

$$\mathrm{E}\left[s_a^n\right] = \int_a^{a+\Delta y} p_y(\bar{y}^n)\; d\bar{y}^n = \Pr(a \le y < a + \Delta y).$$

As in the previous section, the LLN implies that:

$$\frac{1}{N} \sum_{n=1}^{N} s_a^n \xrightarrow{N\to\infty} \mathrm{E}\left[s_a^n\right] = \Pr(a \le y < a + \Delta y).$$

Continuous random variables (3/5)

$$\frac{1}{N}\sum_{n=1}^{N} s_a^n \xrightarrow{N\to\infty} \mathrm{E}\left[s_a^n\right] = \Pr(a \le y < a + \Delta y).$$

The left-hand side of this expression can be reformulated[2] as:

$$\frac{1}{N}\sum_{n=1}^{N} s_a^n = \frac{1}{N}\sum_{n=1}^{N} \int_a^{a+\Delta y} \delta(\xi - y^n)\, d\xi = \int_a^{a+\Delta y} \frac{1}{N}\sum_{n=1}^{N} \delta(\xi - y^n)\, d\xi.$$

Whereas the right-hand side can be restated as:

$$\Pr(a \le y < a + \Delta y) = \int_a^{a+\Delta y} p_y(\xi)\ d\xi.$$

[2]For the Fubini-Tonelli theorem, the order of the integration and the sum can be switched because the Dirac delta pulse is always non-negative.

Continuous random variables (4/5)

Thus, we conclude that

$$\int_a^{a+\Delta y} \frac{1}{N} \sum_{n=1}^{N} \delta(\xi - y^n)\, d\xi \xrightarrow{N\to\infty} \int_a^{a+\Delta y} p_y(\xi)\ d\xi.$$

A smooth and bounded p_y and a small Δy motivate the approximation:

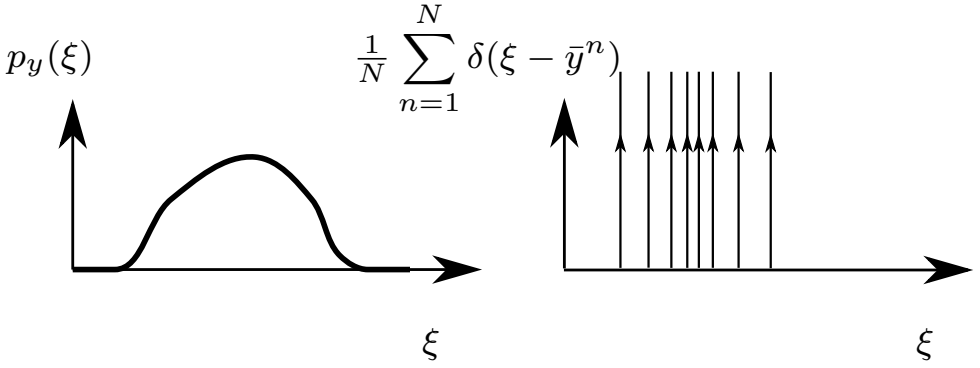
$$p_y(\xi) \approx \frac{1}{N} \sum_{n=1}^{N} \delta(\xi - \bar{y}^n), \quad \forall \xi,$$

where $\bar{y}^n$ denotes the values the CRV y^n takes.

This is an approximation in the sense that taking the integral on both sides yields similar results.

Continuous random variables (5/5)

The graph below shows a sketch of the approximation process, with the original PDF on the left and its approximation on the right.



Using the Dirac delta in the above approximation has the advantage that the same notation applies to discrete and continuous random variables (where δ is the Kronecker delta or the Dirac delta, respectively).

Example (1/3)

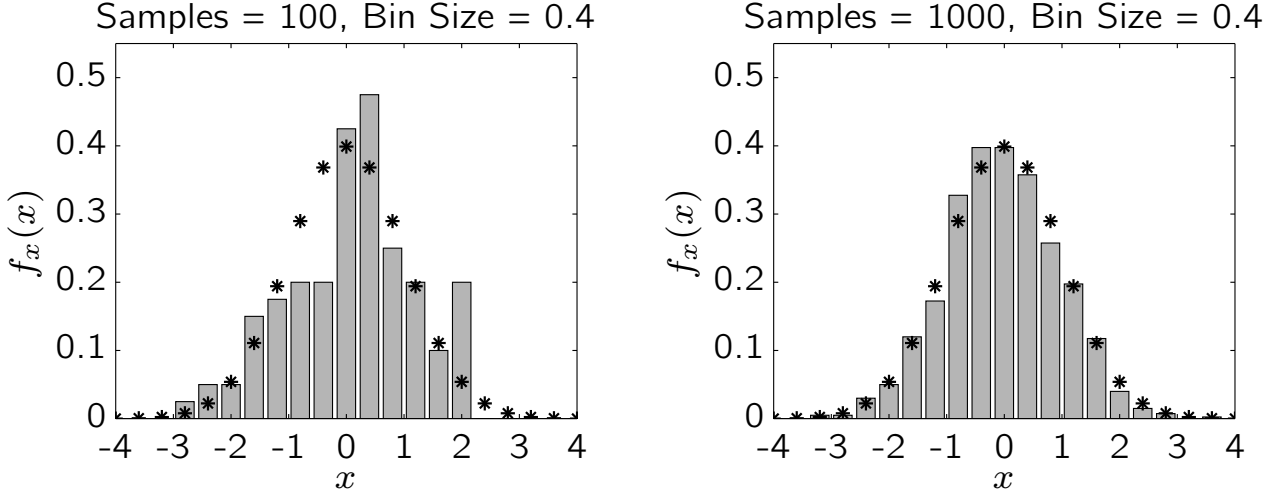
We approximate a Gaussian distribution with zero mean and unit variance by repeatedly drawing samples from this distribution.

The graphs in the next two slide show the result for different numbers of samples N and different bin sizes Δy.

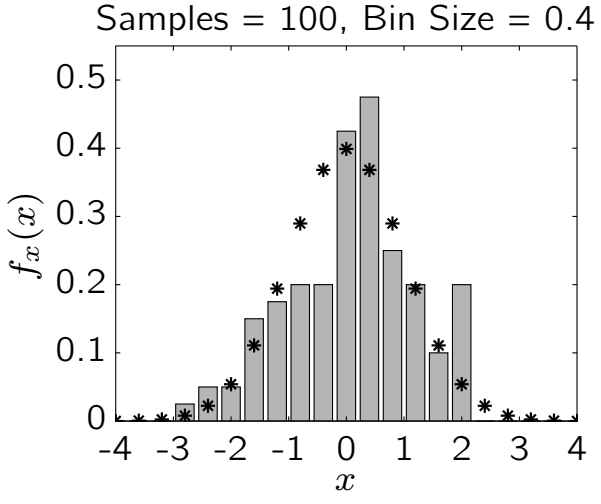
Example (2/3)

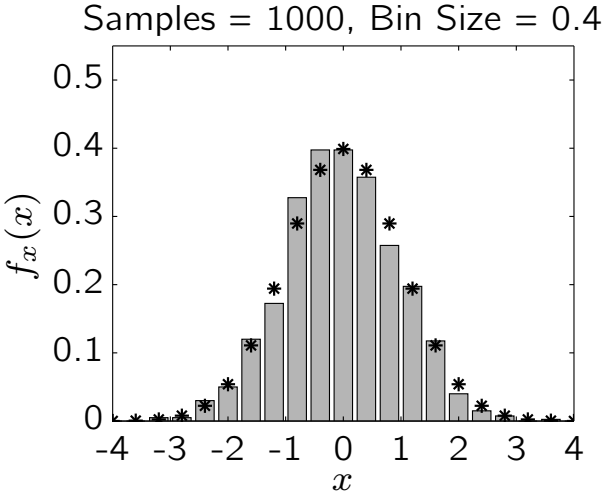
The exact value of the PDF is shown in black (asterisks) and the approximated value obtained from Monte Carlo sampling is indicated with gray bars, where the height of the bars represents the approximated probability of y being within a given bin.

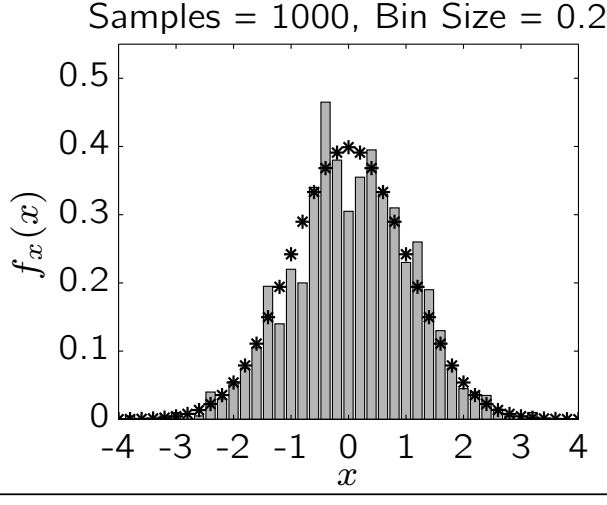
This can also be seen as a step-wise constant approximation of the probability density, as $\Pr(a \leq y < a + \Delta y) \approx p_y(a)\,\Delta y$, for small Δy.

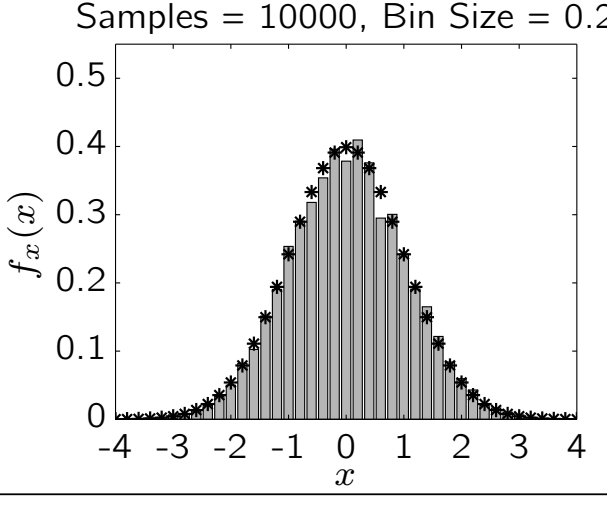


Example (3/3)


Samples = 100, Bin Size = 0.4
$f_x(x)$
0
0.1
0.2
0.3
0.4
0.5
-4 -3 -2 -1 0 1 2 3 4
x


Samples = 1000, Bin Size = 0.4
$f_x(x)$
0
0.1
0.2
0.3
0.4
0.5
-4 -3 -2 -1 0 1 2 3 4
x


Samples = 1000, Bin Size = 0.2
$f_x(x)$
0
0.1
0.2
0.3
0.4
0.5
-4 -3 -2 -1 0 1 2 3 4
x


Samples = 10000, Bin Size = 0.2
$f_x(x)$
0
0.1
0.2
0.3
0.4
0.5
-4 -3 -2 -1 0 1 2 3 4
x

Change of Variables

Using the binning approach, the same procedure as for discrete random variables can be applied.

The results are summarized here.

For $y \in \mathcal{Y}$, it holds:

$$p_y(\xi) \approx \frac{1}{N} \sum_{n=1}^{N} \delta(\xi - \bar{y}^n), \quad \forall \xi.$$

For $x := g(y)$, it holds:

$$p_x(\xi) \approx \frac{1}{N} \sum_{n=1}^{N} \delta(\xi - g(\bar{y}^n)), \quad \forall \xi. \tag{1}$$

Joint Continuous Random Variables

The above can also be extended for joint continuous random variables,

$$p_x(\xi) \approx \frac{1}{N} \sum_{n=1}^{N} \delta(\xi - \bar{x}^n), \quad \forall \xi,$$

where ξ and $\bar{x}^n$ are now vectors, and $\delta(\cdot)$ refers to the vector version of the Dirac delta pulse.

Outline

Recall: Bayesian State Estimation (1/2)

The Bayesian tracking algorithm (also called Bayesian state estimator) computes $p_{x(k)|z(1:k)}$.

The objective of the PF is to *approximate* $p_{x(k)|z(1:k)}$.

We introduce auxiliary variables, just like we did for the Kalman filter.

We define new random variables, $x_{\mathrm{p}}(k)$, $x_{\mathrm{m}}(k)$, and $z_{\mathrm{m}}(k)$, and we distinguish again between the random variable $z(k)$ and the value $\bar{z}(k)$ that $z(k)$ takes (the actual measurement at time k).

Init: $x_{\mathrm{m}}(0) := x(0)$

S1: $x_{\mathrm{p}}(k) := q_{k-1}(x_{\mathrm{m}}(k-1), v(k-1))$

S2: $z_{\mathrm{m}}(k) := h_k(x_{\mathrm{p}}(k), w(k))$

$x_{\mathrm{m}}(k)$ defined via its PDF

$$p_{x_{\mathrm{m}}(k)}(\xi) := p_{x_{\mathrm{p}}(k)|z_{\mathrm{m}}(k)}(\xi|\bar{z}(k)), \quad \forall \xi$$

$\Big\}$ $k = 1, 2, \ldots$.

Recall: Bayesian State Estimation (1/2)

Analogously to the proof for linear systems, it can be shown that for all ξ and $k = 1, 2, \ldots$:

$$p_{x_{\mathrm{p}}(k)}(\xi) = p_{x(k)|z(1:k-1)}(\xi|\bar{z}(1{:}k{-}1))\,,$$

$$p_{x_{\mathrm{m}}(k)}(\xi) = p_{x(k)|z(1:k)}(\xi|\bar{z}(1{:}k))\,.$$

The prior update step will be presented with the next slides, while the posterior update step will be presented in the next lecture.

Outline

Prior Update (S1) (1/2)

Given the PDF $p_{x_{\mathrm{m}}(k-1)}$ of $x_{\mathrm{m}}(k–1)$, we construct the PDF $p_{x_{\mathrm{p}}(k)}$ of $x_{\mathrm{p}}(k)$.

We *approximate* $p_{x_{\mathrm{m}}(k-1)}$ and $p_{x_{\mathrm{p}}(k)}$ by Monte Carlo sampling.

Let:

$$p_{x_{\mathrm{m}}(k-1)}(\xi) \approx \frac{1}{N}\sum_{n=1}^{N}\delta(\xi - \bar{x}_{\mathrm{m}}^{n}(k–1)), \quad \forall \xi,$$

where $\{\bar{x}_{\mathrm{m}}^{n}(k–1)\}$ are N particles that approximate the PDF of $x_{\mathrm{m}}(k–1)$.

These particles are Monte Carlo samples of $p_{x_{\mathrm{m}}(k-1)}$.

We can therefore form an approximation of:

$$p_{x_{\mathrm{p}}(k)}(\xi) \approx \frac{1}{N}\sum_{n=1}^{N}\delta(\xi - \bar{x}_{\mathrm{p}}^{n}(k)), \quad \forall \xi,$$

where

$$\bar{x}_{\mathrm{p}}^{n}(k) := q_{k-1}(\bar{x}_{\mathrm{m}}^{n}(k–1), \bar{v}^{n}(k–1)), \qquad \text{for } n = 1, 2, \ldots, N,$$

and $\bar{v}^{n}(k–1)$ are Monte Carlo samples of $p_{v(k-1)}$.

Prior Update (S1) (2/2)

This is straightforward and intuitive.

We simply propagate N particles through the process dynamics.

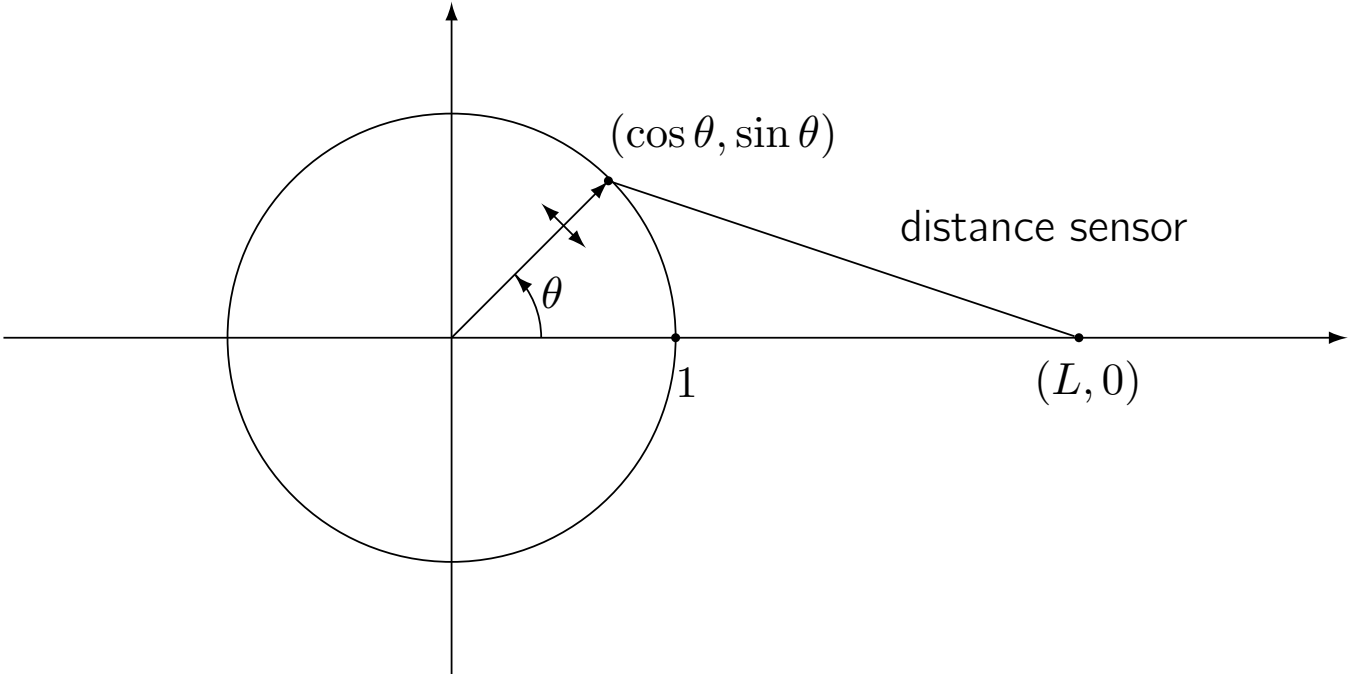
This is like a parallel simulation.

Provided N is "large" *and* $\{\bar{x}_{\mathrm{m}}^n(k-1)\}$ is a "good" representation of $p_{x_{\mathrm{m}}(k-1)}$, we can conclude that $\{\bar{x}_{\mathrm{p}}^n(k)\}$ will be a "good" representation of $p_{x_{\mathrm{p}}(k)}$.

Outline

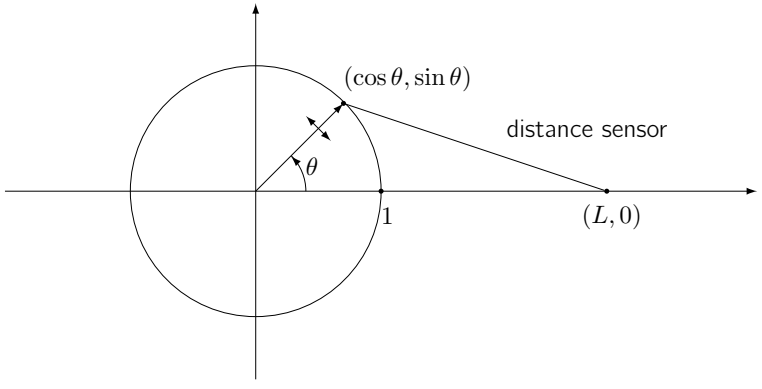
Example of Prior Update (1/4)

We revisit an old example, slightly modified. We apply the PF prior update step and do not consider any measurements yet.



- $\theta(k)$: location of an object on a circle, $\theta(k) \in [0, 2\pi)$.

Example of Prior Update (2/4)



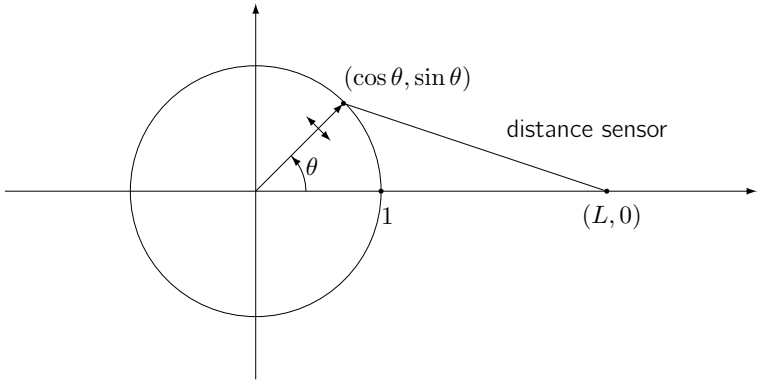
- Dynamics:

$$\theta(k) = \text{mod}\left(\theta(k{-}1) + s(k{-}1),\, 2\pi\right),$$

where $s(k{-}1)$ is a CRV uniformly distributed on $[-\overline{s} + b, \overline{s} + b]$, and b is a CRV (not time-dependent) uniformly distributed on $[-\overline{s}, \overline{s}]$.

The modulo operation $\text{mod}(y, 2\pi)$ adds or subtracts multiples of 2π to the argument y such that the result is in the interval $[0, 2\pi)$.

Example of Prior Update (3/4)



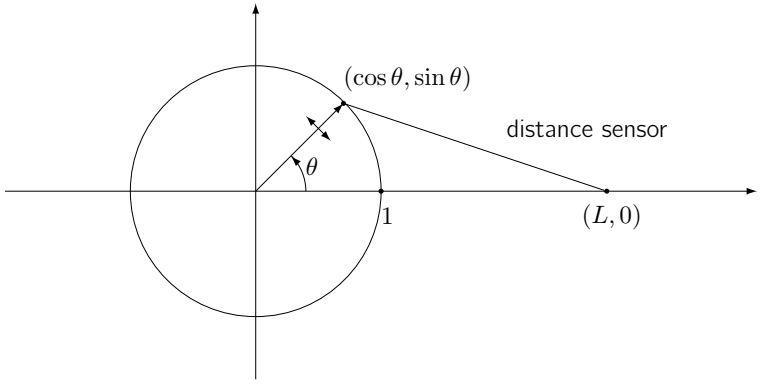
- Introducing the states $x_1(k) := b$ and $x_2(k) := \theta(k)$, we can rewrite the dynamics in standard form:

$$x_1(k) = x_1(k\text{–}1),$$
$$x_2(k) = \text{mod}\left(x_2(k\text{–}1) + x_1(k\text{–}1) + v(k\text{–}1),\, 2\pi\right),$$

  where $x_1(0)$ is uniformly distributed on $[-\overline{s}, \overline{s}]$ and $v(k\text{–}1)$ is uniformly distributed on $[-\overline{s}, \overline{s}]$.

  The distribution of $x_2(0)$ reflects our initial knowledge of the object's location.

Example of Prior Update (4/4)



- Note that $x(k)$ is now a CRV.

  When we discussed the example in the context of Bayesian tracking, we assumed that the object moves in discrete steps, which allowed us to implement the Bayesian tracking algorithm directly.

  Here, we use a PF to track the object's location. As a first step, we consider the PF without measurements.

  Next lecture, we add measurements.

Implementation and Simulation Results

- Parameter values (known to the PF): the object's initial location $\theta(0) = \frac{\pi}{2}$, $\overline{s} = 0.03$.
- We use $N = 1000$ particles. Each particle $x^n(k)$ is a vector with two elements: the first corresponding to the bias b and the second corresponding to the location $\theta(k)$.
- The histogram of the object's location spreads out over time: because of the process noise, our knowledge of the object's location becomes less accurate. This is amplified when increasing $\overline{s}$.
- The bias histogram does not change. Without any measurement, there is no way to infer information about the bias, and the process dynamics do not alter $x_1(k)$.
- If we do not know $\theta(0)$ and initialize the PF with particles drawn from a uniform distribution on $[0, 2\pi)$, we basically have no idea where the object is.