概率机器人（Probabilistic Robotics）

吉建民

USTC
jianmin@ustc.edu.cn

2021 年 5 月 6 日

Used Materials

Disclaimer: 本课件大量采用了 Jana Kosecka’s Autonomous Robotics 课件及其他网络课程课件，也采用了 GitHub 中开源代码，以及部分网络博客内容

Table of Contents

基础
    概率论基础

递归状态估计
    机器人环境交互
    贝叶斯滤波

高斯滤波（Gaussian Filter）
    卡尔曼滤波
    扩展卡尔曼滤波
    无迹卡尔曼滤波（Unscented Kalman Filter）

非参数滤波（Nonparametric Filters）
    离散贝叶斯滤波（Discrete Bayes Filter）
    粒子滤波（Particle Filter）

概率机器人

- 机器人中的不确定性：
  - 环境不确定性：动态环境难以预测
  - 感知不确定性：传感器物理限制
  - 行动不确定性：执行结构噪音
  - 模型误差：真实世界的近似模型
  - 算法误差：近似算法
- 概率机器人：Explicit representation of uncertainty using the calculus of probability theory
  - Perception = state estimation
  - Action = utility optimization

Table of Contents

基础
    概率论基础

递归状态估计
    机器人环境交互
    贝叶斯滤波

高斯滤波（Gaussian Filter）
    卡尔曼滤波
    扩展卡尔曼滤波
    无迹卡尔曼滤波（Unscented Kalman Filter）

非参数滤波（Nonparametric Filters）
    离散贝叶斯滤波（Discrete Bayes Filter）
    粒子滤波（Particle Filter）

样本空间和事件

- 随机试验：事先不能完全预知其结果的试验
  - 抛掷骰子是一个随机试验
- 样本空间：随机试验的所有可能结果组成的集合，记为 Ω
  - 掷骰子的样本空间 $\Omega = \{1, 2, 3, 4, 5, 6\}$
- 原子事件（样本点）：样本空间中的点，即随机试验的可能结果，记为 ω
- 事件：样本空间的子集，记为 A, B, ...
  - $A = \{1, 3, 5\}$ 表示“掷出结果为奇数”这一事件
  - Ω 本身为必然事件，$\emptyset$ 为不可能事件
- 若两事件 $A \cap B = \emptyset$，称为互斥事件（不相容事件）
- 若两事件 $A \cap B = \emptyset$ 且 $A \cup B = \Omega$，称为互补事件

概率

- 概率测度：给样本空间中的每一个事件 A 赋予一个数值（概率） $P(A) \in [0, 1]$
- 概率测度（形式化）是一个从样本空间 Ω 的幂集 2^{Ω} 到区间 $[0, 1]$ 的映射 $P: 2^{\Omega} \to [0, 1]$，且满足以下三个 Kolmogorov 公理：

  (1) $P(\Omega) = 1$;（规范性）

  (2) $P(A) \geq 0, \forall A \in 2^{\Omega}$;（非负性）

  (3) $P(A \cup B) = P(A) + P(B), \forall A, B \in 2^{\Omega}, A \cap B = \emptyset$.（有限可加性）

- $P(A)$ 称为事件 A 的概率

随机变量和概率函数

- 随机变量是定义在样本空间 Ω 上的函数，记为 X, Y, Z
- 随机变量的取值随试验结果而定，记为 x, y, z
- 随机变量 X 的所有可能取值的集合称为其值域（状态空间），记为 Ω_X
- 设 X 为一随机变量，x 是它的一个取值，在样本空间 Ω 中，所有使 X 取值为 x 的原子事件组成一个事件，记为 $\Omega_{X=x} = \{w \in \Omega \mid X(\omega) = x\}$，简记为 “$X = x$”
- 事件 “$X = x$” 的概率 $P(X = x) = P(\Omega_{X=x})$ 依赖于 X 的取值 x，让 x 在 Ω_X 上变动，$P(X = x)$ 就称为 Ω_X 的一个取值于 $[0, 1]$ 的函数，称为随机变量 X 的概率质量函数 (probability mass function)，记为 $P(X)$
- 根据概率测度的定义

$$P(X = x) \geq 0, \forall x \in \Omega_X \text{ 简记为 } P(X) \geq 0$$

$$\sum_{x \in \Omega_X} P(X = x) = 1 \text{ 简记为 } \sum_{x} P(X = x) = 1.$$

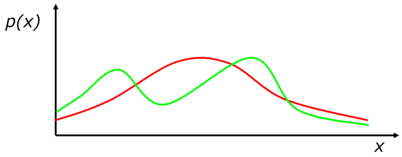
连续随机变量（Continuous Random Variables）

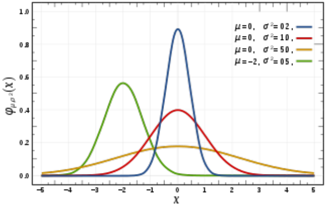
- X 是随机变量，并且取值是连续的
- $p(X = x)$ or $p(x)$ 为概率密度函数（probability density function）

$$P(x \in (a, b)) = \int_a^b p(x)\,\mathrm{d}x$$

- 例如：高斯分布（正态分布），均值为 μ，标准差为 σ

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$





联合概率分布（Joint Probability）

- 对多个随机变量 $X_1, \ldots, X_n$，用联合概率分布 $P(X_1, \ldots, X_n)$ 来描述各变量所有可能的状态组合的概率。
- 联合分布是定义在所有变量状态空间的笛卡尔乘积上的函数：
  - $P(X_1, \ldots, X_n): \otimes_{i=1}^{n} \Omega_{X_i} \to [0, 1]$
  - $\Sigma_{X_1, \ldots, X_n} P(X_1, \ldots, X_n) = 1$
- 联合分布通常表示为一张表，包含 $\Pi_{i=1}^{n} |\Omega_{X_i}|$ 个状态组合及其概率值。例，香港租房市场

	public	private	others
low	0.17	0.01	0.02
medium	0.44	0.03	0.01
upper medium	0.09	0.07	0.01
high	0	0.14	0.01

- 记 $\mathbf{X} = \{X_1, \ldots, X_n\}$，$\mathbf{Y}$ 是 $\mathbf{X}$ 的真子集（$\mathbf{Y} \subset \mathbf{X}$），$\mathbf{Z} = \mathbf{X} \setminus \mathbf{Y}$。则相对于 $P(\mathbf{X})$，$\mathbf{Y}$ 的边缘分布 $P(\mathbf{Y})$ 定义为 $P(\mathbf{Y}) = \Sigma_{\mathbf{Z}} P(X_1, \ldots, X_n)$，称为边缘化

条件概率分布（Conditional Probability）

- 条件概率：
$$P(A \mid B) = \frac{P(A \cap B)}{P(B)}$$
- 条件概率分布：
$$P(X = x \mid Y = y) = \frac{P(X = x, \ Y = y)}{P(Y = y)}$$
固定 y，让 x 在 Ω_X 上变动，得到函数 $P(X \mid Y = y)$ （在给定 $Y = y$ 时变量 X 的条件概率分布）；
$P(X \mid Y) = \{P(X \mid Y = y) \mid y \in \Omega_Y\}$ （给定 Y 时变量 X 的条件概率分布）
$$P(\mathbf{X} \mid \mathbf{Y}) = \frac{P(\mathbf{X}, \mathbf{Y})}{P(\mathbf{Y})}$$
- 链规则：
$$P(X_1, X_2, \ldots, X_n) = P(X_1) P(X_2 \mid X_1) \cdots P(X_n \mid X_1, \ldots, X_{n-1})$$

条件独立（Conditional Independence）

- 事件 A 与 B 相互独立： $P(A \cap B) = P(A)P(B)$
  - 当 $P(A) > 0$ 时， $P(B) = P(B \mid A)$.
- 事件 A 与 B 在给定 C 时相互条件独立：

$$P(A \cap B \mid C) = P(A \mid C)P(B \mid C)$$

  - 当 $P(B \cap C) > 0$ 时， $P(A \mid C) = P(A \mid B \cap C)$.
- 两个变量 X 和 Y 相互独立，记为 $X \perp Y$:

$$P(X, Y) = P(X)P(Y)$$

  - 若 $P(Y = y) > 0$，则 $P(X) = P(X \mid Y = y)$.
- 三个随机变量 X, Y 和 Z，设 $P(Z = z) > 0, \forall z \in \Omega_Z$， X 和 Y 在给定 Z 时相互条件独立，记为 $X \perp Y \mid Z$:

$$P(X, Y \mid Z) = P(X \mid Z)P(Y \mid Z)$$

.
  - 若 $P(Y = y, Z = z) > 0$，
    则 $P(X \mid Y = y, Z = z) = P(X \mid Z = z)$.

一些规则

Discrete case

$$\sum_x P(x) = 1$$

$$P(x) = \sum_y P(x, y)$$

$$P(x) = \sum_y P(x \mid y) P(y)$$

Continuous case

$$\int p(x)\, dx = 1$$

$$p(x) = \int p(x, y)\, dy$$

$$p(x) = \int p(x \mid y)\, p(y)\, dy$$

贝叶斯公式（Bayes Formula）

- 在考虑证据 $E = e$ 之前，对事件 $H = h$ 的概率估计 $P(H = h)$ 称为先验概率；而在考虑证据之后，对 $H = h$ 的概率估计 $P(H = h \mid E = e)$ 称为后验概率
- 贝叶斯定理（贝叶斯规则、公式）

$$P(H = h \mid E = e) = \frac{P(H = h)P(E = e \mid H = h)}{P(E = e)}$$

$$P(X \mid E = e) = \frac{P(X)P(E = e \mid X)}{P(E = e)}$$

$$P(x, y) = P(x \mid y)P(y) = P(y \mid x)P(x)$$

$$\Rightarrow$$

$$\boxed{P(x \mid y) = \frac{P(y \mid x)\ P(x)}{P(y)} = \frac{\text{likelihood} \cdot \text{prior}}{\text{evidence}}}$$

贝叶斯公式（Bayes Formula）

$$P(x \mid y) = \frac{P(y \mid x)P(x)}{P(y)} = \frac{P(y \mid x)P(x)}{\sum_{x'} P(y \mid x')P(x')}$$

$$P(x \mid y) = \frac{P(y \mid x)P(x)}{P(y)} = \frac{P(y \mid x)P(x)}{\int P(y \mid x')P(x')\,\mathrm{d}x'}$$

- 如果 x 是一个希望由 y 推测出来的数值，则概率 $P(x)$ 称为先验概率分布（prior probability distribution）
- y 称为数据（data），也就是传感器测量值
- $P(x)$ 总结了在综合数据 y 之前已经有的关于 x 的信息
- 概率 $P(x \mid y)$ 称为在 X 上的后验概率分布（posterior probability distribution）
- 贝叶斯准则利用“逆”条件概率 $P(y \mid x)$ 和先验概率 $P(x)$ 计算后验概率 $P(x \mid y)$
- $P(y \mid x)$ 称为生成模型（generative model），表示变量 X 如何引起检测数据 Y

归一化（Normalization）

$$P(x|y) = \frac{P(y|x)\ P(x)}{P(y)} = \eta\ P(y|x)\,P(x)$$

$$\eta = P(y)^{-1} = \frac{1}{\sum_x P(y|x)P(x)}$$

Algorithm:

$$\forall x : \mathrm{aux}_{x|y} = P(y|x)\ P(x)$$

$$\eta = \frac{1}{\sum_x \mathrm{aux}_{x|y}}$$

$$\forall x : P(x|y) = \eta\ \mathrm{aux}_{x|y}$$

Table of Contents

基础

概率论基础

递归状态估计

机器人环境交互

贝叶斯滤波

高斯滤波（Gaussian Filter）

卡尔曼滤波

扩展卡尔曼滤波

无迹卡尔曼滤波（Unscented Kalman Filter）

非参数滤波（Nonparametric Filters）

离散贝叶斯滤波（Discrete Bayes Filter）

粒子滤波（Particle Filter）

Table of Contents

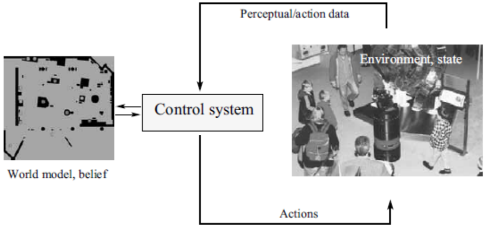
基础
    概率论基础

递归状态估计
    机器人环境交互
    贝叶斯滤波

高斯滤波（Gaussian Filter）
    卡尔曼滤波
    扩展卡尔曼滤波
    无迹卡尔曼滤波（Unscented Kalman Filter）

非参数滤波（Nonparametric Filters）
    离散贝叶斯滤波（Discrete Bayes Filter）
    粒子滤波（Particle Filter）

机器人环境交互



- 环境状态（state）：$x_{t_1:t_2} = x_{t_1}, x_{t_1+1}, \ldots, x_{t_2}$
- 环境传感器测量（measurement）：$z_{t_1:t_2} = z_{t_1}, z_{t_1+1}, \ldots, z_{t_2}$
- 控制行动（control action）：$u_{t_1:t_2} = u_{t_1}, u_{t_1+1}, \ldots, u_{t_2}$

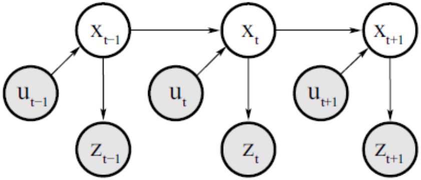
概率生成法则

- 状态概率生成法则： $P(x_t \mid x_{0:t-1}, z_{1:t-1}, u_{1:t})$
- 如果状态 x 是完整的，则：
  - 状态转移概率： $P(x_t \mid x_{0:t-1}, z_{1:t-1}, u_{1:t}) = P(x_t \mid x_{t-1}, u_t)$
  - 测量概率： $P(z_t \mid x_{0:t-1}, z_{1:t-1}, u_{1:t}) = P(z_t \mid x_t)$
- 置信度（belief）：已知条件下对状态估计的概率分布

$$bel(x_t) = P(x_t \mid z_{1:t}, u_{1:t})$$

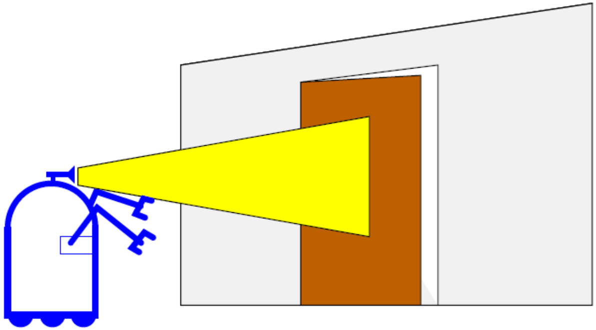
$$\overline{bel}(x_t) = P(x_t \mid z_{1:t-1}, u_{1:t}) \qquad \text{prediction}$$

- 由 $\overline{bel}(x_t)$ 计算 $bel(x_t)$ 为修正（correction）或测量更新（measurement update）



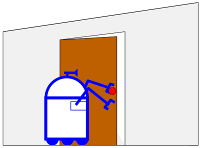
状态估计

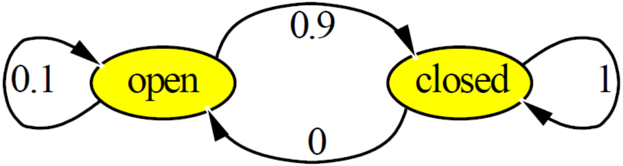
- Suppose a robot obtains measurement z
- What is $P(open \mid z)$?



状态估计

- $P(x \mid u, x')$ for $u =$ "close door"





If the door is open, the action "close door" succeeds in 90% of all cases

Table of Contents

基础
- 概率论基础

递归状态估计
- 机器人环境交互
- 贝叶斯滤波

高斯滤波（Gaussian Filter）
- 卡尔曼滤波
- 扩展卡尔曼滤波
- 无迹卡尔曼滤波（Unscented Kalman Filter）

非参数滤波（Nonparametric Filters）
- 离散贝叶斯滤波（Discrete Bayes Filter）
- 粒子滤波（Particle Filter）

Recursive Bayesian Updating

给定观察 $z_1, \ldots, z_n$， **估计状态** x

$$P(x \mid z_1, \ldots, z_n) = \frac{P(z_n \mid x, z_1, \ldots, z_{n-1}) P(x \mid z_1, \ldots, z_{n-1})}{P(z_n \mid z_1, \ldots, z_{n-1})}$$

- Markov Assumption: z_n is independent of $z_1, \ldots, z_{n-1}$ if we know x

$$\begin{aligned} P(x \mid z_1, \ldots, z_n) &= \frac{P(z_n \mid x) P(x \mid z_1, \ldots, z_{n-1})}{P(z_n \mid z_1, \ldots, z_{n-1})} \\ &= \eta P(z_n \mid x) P(x \mid z_1, \ldots, z_{n-1}) \\ &= \eta_{1,\ldots,n} \left(\prod_{i=1,\ldots,n} P(z_i \mid x) \right) P(x) \end{aligned}$$

Integrating the Outcome of Actions

- Continuous case:

$$P(x \mid u) = \int P(x \mid u, x') P(x') \, \mathrm{d}x'$$

- Discrete case:

$$P(x \mid u) = \sum P(x \mid u, x') P(x')$$

Bayes Filters: Framework

- Given:
  - Stream of observations z and action data u:

$$d_t = \{u_1, z_1, \ldots, u_t, z_t\}$$

  - Sensor model $P(z \mid x)$
  - Action model $P(x \mid u, x')$
  - Prior probability of the system state $P(x)$
- Wanted:
  - Estimate of the state X of a dynamical system
  - The posterior of the state is also called Belief:

$$bel(x_t) = P(x_t \mid u_1, z_1, \ldots, u_t, z_t)$$

Markov Assumption

$$\begin{tikzcd}
 & u_{t-1} \arrow[dr] & & u_t \arrow[dr] & & u_{t+1} \arrow[dr] & & & \\
\cdots \arrow[rr] & & x_{t-1} \arrow[rr] \arrow[d] & & x_t \arrow[rr] \arrow[d] & & x_{t+1} \arrow[r] \arrow[d] & {} & \cdots \\
 & & z_{t-1} & & z_t & & z_{t+1} & &
\end{tikzcd}$$

$$P(z_t \mid x_{0:t}, z_{1:t-1}, u_{1:t}) = P(z_t \mid x_t)$$
$$P(x_t \mid x_{1:t-1}, z_{1:t-1}, u_{1:t}) = P(x_t \mid x_{t-1}, u_t)$$

Underlying Assumptions

- Static world
- Independent noise
- Perfect model, no approximation errors

Bayes Filters

$$
\begin{aligned}
& bel(x_t) \\
&= P(x_t \mid z_{1:t}, u_{1:t}) \\
&= \eta P(z_t \mid x_t, z_{1:t-1}, u_{1:t}) P(x_t \mid z_{1:t-1}, u_{1:t}) && \text{E} \\
&= \eta P(z_t \mid x_t) P(x_t \mid z_{1:t-1}, u_{1:t}) && \text{Ma} \\
&= \eta P(z_t \mid x_t) \overline{bel}(x_t) \\
&= \eta P(z_t \mid x_t) \int P(x_t \mid x_{t-1}, z_{1:t-1}, u_{1:t}) p(x_{t-1} \mid z_{1:t-1}, u_{1:t}) \, \mathrm{d}x_{t-1} && \text{Total} \\
&= \eta P(z_t \mid x_t) \int P(x_t \mid u_t, x_{t-1}) P(x_{t-1} \mid z_{1:t-1}, u_{1:t}) \, \mathrm{d}x_{t-1} && \text{Ma} \\
&= \eta P(z_t \mid x_t) \int P(x_t \mid u_t, x_{t-1}) P(x_{t-1} \mid z_{1:t-1}, u_{1:t-1}) \, \mathrm{d}x_{t-1} && \text{Ma} \\
&= \eta P(z_t \mid x_t) \int P(x_t \mid u_t, x_{t-1}) bel(x_{t-1}) \, \mathrm{d}x_{t-1}
\end{aligned}
$$

Bayes Filter Algorithm

$$Bel(x_t) = \eta P(z_t \mid x_t) \int P(x_t \mid u_t, x_{t-1}) Bel(x_{t-1}) dx_{t-1}$$

```
1.  Algorithm Bayes_filter($Bel(x)$, $d$):
2.  $\eta$=0
3.  If $d$ is a perceptual data item $z$ then
4.      For all $x$ do
5.          $Bel'(x) = P(z \mid x) Bel(x)$
6.          $\eta = \eta + Bel'(x)$
7.      For all $x$ do
8.          $Bel'(x) = \eta^{-1} Bel'(x)$
9.  Else if $d$ is an action data item $u$ then
10.     For all $x$ do
11.         $Bel'(x) = \int P(x \mid u, x') Bel(x') dx'$
12. Return $Bel'(x)$
```

Bayes Filters are Familiar

- Prediction

$$\overline{bel}(x_t) = \int P(x_t \mid u_t, x_{t-1})\, bel(x_{t-1})\, \mathrm{d}x_{t-1}$$

- Correction

$$bel(x_t) = \eta P(z_t \mid x_t)\, \overline{bel}(x_t)$$

$$\boxed{Bel(x_t) = \eta P(z_t \mid x_t) \int P(x_t \mid u_t, x_{t-1}) Bel(x_{t-1}) dx_{t-1}}$$

- Kalman filters
- Particle filters
- Hidden Markov models
- Dynamic Bayesian networks
- Partially Observable Markov Decision Processes (POMDPs)

Table of Contents

基础
- 概率论基础

递归状态估计
- 机器人环境交互
- 贝叶斯滤波

高斯滤波（Gaussian Filter）
- 卡尔曼滤波
- 扩展卡尔曼滤波
- 无迹卡尔曼滤波（Unscented Kalman Filter）

非参数滤波（Nonparametric Filters）
- 离散贝叶斯滤波（Discrete Bayes Filter）
- 粒子滤波（Particle Filter）

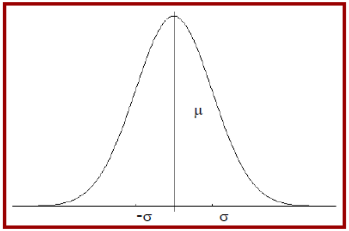
高斯分布

Gaussians

$p(x) \sim N(\mu, \sigma^2)$:

$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}}$$

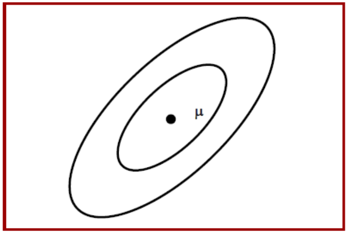
Univariate



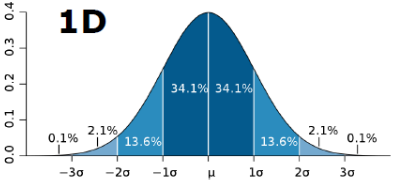
$p(\mathbf{x}) \sim N(\mu, \Sigma)$:

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{d/2} |\Sigma|^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\mu)^t \Sigma^{-1} (\mathbf{x}-\mu)}$$

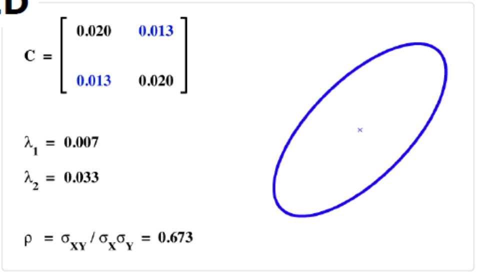
Multivariate



高斯分布



2D



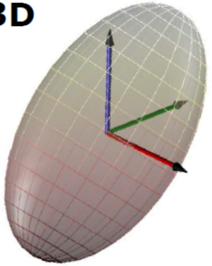
$$C = \begin{bmatrix} 0.020 & 0.013 \\ 0.013 & 0.020 \end{bmatrix}$$

$\lambda_1 = 0.007$

$\lambda_2 = 0.033$

$\rho = \sigma_{XY} / \sigma_X \sigma_Y = 0.673$

3D



Properties of Gaussians

- Univariate case

$$\left.\begin{array}{l} X \sim N(\mu,\sigma^2) \\ Y = aX + b \end{array}\right\} \Rightarrow \quad Y \sim N(a\mu + b, a^2\sigma^2)$$

$$\left.\begin{array}{l} X_1 \sim N(\mu_1,\sigma_1^{\;2}) \\ X_2 \sim N(\mu_2,\sigma_2^{\;2}) \end{array}\right\} \Rightarrow p(X_1)\cdot p(X_2) \sim N\left(\frac{\sigma_2^{\;2}}{\sigma_1^{\;2}+\sigma_2^{\;2}}\mu_1 + \frac{\sigma_1^{\;2}}{\sigma_1^{\;2}+\sigma_2^{\;2}}\mu_2, \quad \frac{1}{\sigma_1^{\;-2}+\sigma_2^{\;-2}}\right)$$

Properties of Gaussians

- Multivariate case

$$\boxed{\left.\begin{array}{r} X \sim N(\mu,\Sigma) \\ Y = AX + B \end{array}\right\} \Rightarrow \quad Y \sim N(A\mu + B, A\Sigma A^T)}$$

$$\boxed{\left.\begin{array}{l} X_1 \sim N(\mu_1,\Sigma_1) \\ X_2 \sim N(\mu_2,\Sigma_2) \end{array}\right\} \Rightarrow p(X_1)\cdot p(X_2) \sim N\left(\frac{\Sigma_2}{\Sigma_1+\Sigma_2}\mu_1 + \frac{\Sigma_1}{\Sigma_1+\Sigma_2}\mu_2, \quad \frac{1}{\Sigma_1^{-1}+\Sigma_2^{-1}}\right)}$$

(where division "–" denotes matrix inversion)

- We **stay Gaussian** as long as we start with Gaussians and perform only **linear transformations**

Table of Contents

基础
- 概率论基础

递归状态估计
- 机器人环境交互
- 贝叶斯滤波

高斯滤波（Gaussian Filter）
- 卡尔曼滤波
- 扩展卡尔曼滤波
- 无迹卡尔曼滤波（Unscented Kalman Filter）

非参数滤波（Nonparametric Filters）
- 离散贝叶斯滤波（Discrete Bayes Filter）
- 粒子滤波（Particle Filter）

Discrete Kalman Filter

Estimates the state x of a discrete-time controlled process that is governed by the linear stochastic difference equation

$$\boxed{x_t = A_t x_{t-1} + B_t u_t + \varepsilon_t}$$

with a measurement

$$\boxed{z_t = C_t x_t + \delta_t}$$

Components of a Kalman Filter

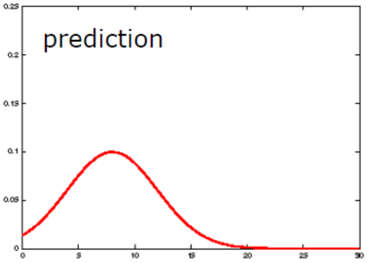
$\boxed{A_t}$ Matrix ($n \times n$) that describes how the state evolves from t-1 to t without controls or noise.

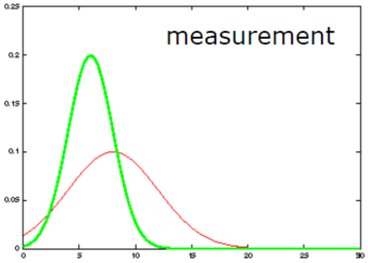
$\boxed{B_t}$ Matrix ($n \times l$) that describes how the control u_t changes the state from t-1 to t.

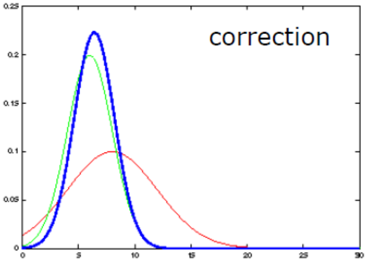
$\boxed{C_t}$ Matrix ($k \times n$) that describes how to map the state x_t to an observation z_t.

$\boxed{\varepsilon_t}$ $\boxed{\delta_t}$ Random variables representing the process and measurement noise that are assumed to be independent and normally distributed with covariance Q_t and R_t respectively.

Kalman Filter Updates in 1D

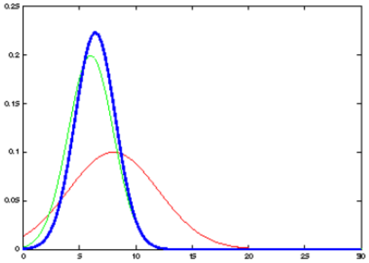






It's a weighted mean!

Kalman Filter Updates in 1D



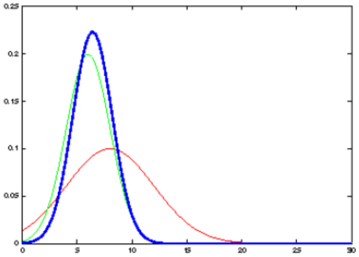
How to get the blue one?

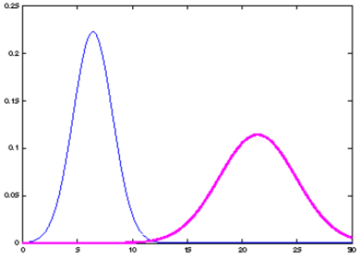
Kalman correction step

$$bel(x_t) = \begin{cases} \mu_t = \bar{\mu}_t + K_t(z_t - \bar{\mu}_t) \\ \sigma_t^2 = (1 - K_t)\bar{\sigma}_t^2 \end{cases} \quad \text{with} \quad K_t = \frac{\bar{\sigma}_t^2}{\bar{\sigma}_t^2 + \bar{\sigma}_{obs,t}^2}$$

$$bel(x_t) = \begin{cases} \mu_t = \bar{\mu}_t + K_t(z_t - C_t\bar{\mu}_t) \\ \Sigma_t = (I - K_t C_t)\bar{\Sigma}_t \end{cases} \quad \text{with} \quad K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + R_t)^{-1}$$

Kalman Filter Updates in 1D





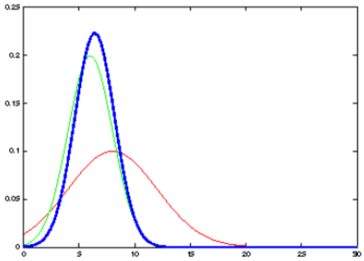
$$\overline{bel}(x_t) = \begin{cases} \bar{\mu}_t = a_t \mu_{t-1} + b_t u_t \\ \bar{\sigma}_t^2 = a_t^2 \sigma_t^2 + \sigma_{act,t}^2 \end{cases}$$

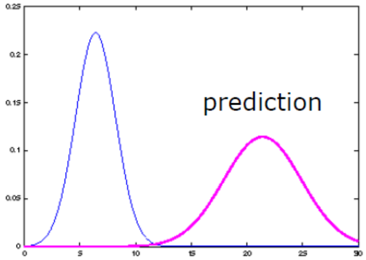
$$\overline{bel}(x_t) = \begin{cases} \bar{\mu}_t = A_t \mu_{t-1} + B_t u_t \\ \bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + Q_t \end{cases}$$

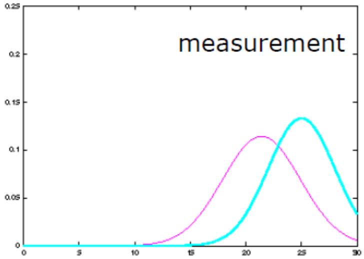
How to get the magenta one?

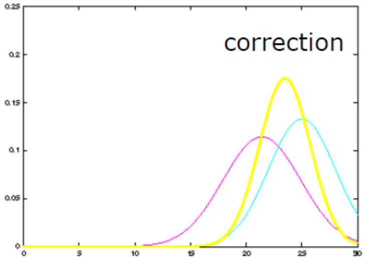
State prediction step

Kalman Filter Updates









Linear Gaussian Systems: Initialization

Initial belief is normally distributed:

$$\boxed{bel(x_0) = N(x_0; \mu_0, \Sigma_0)}$$

Linear Gaussian Systems: Dynamics

Dynamics are linear functions of the state and the control plus additive noise:

$$\boxed{x_t = A_t x_{t-1} + B_t u_t + \varepsilon_t}$$

$$\boxed{p(x_t \mid u_t, x_{t-1}) = N\left(x_t; A_t x_{t-1} + B_t u_t, Q_t\right)}$$

$$\boxed{\begin{array}{ll} \overline{bel}(x_t) = \int p(x_t \mid u_t, x_{t-1}) & bel(x_{t-1})\, dx_{t-1} \\ \qquad\Downarrow & \qquad\Downarrow \\ \quad \sim N\left(x_t; A_t x_{t-1} + B_t u_t, Q_t\right) & \sim N\left(x_{t-1}; \mu_{t-1}, \Sigma_{t-1}\right) \end{array}}$$

Linear Gaussian Systems: Dynamics

$$\overline{bel}(x_t) = \int p(x_t \mid u_t, x_{t-1}) \qquad bel(x_{t-1})\, dx_{t-1}$$

$$\Downarrow \qquad\qquad\qquad\qquad \Downarrow$$

$$\sim N(x_t; A_t x_{t-1} + B_t u_t, Q_t) \quad \sim N(x_{t-1}; \mu_{t-1}, \Sigma_{t-1})$$

$$\Downarrow$$

$$\overline{bel}(x_t) = \eta \int \exp\left\{-\frac{1}{2}(x_t - A_t x_{t-1} - B_t u_t)^T Q_t^{-1} (x_t - A_t x_{t-1} - B_t u_t)\right\}$$

$$\exp\left\{-\frac{1}{2}(x_{t-1} - \mu_{t-1})^T \Sigma_{t-1}^{-1} (x_{t-1} - \mu_{t-1})\right\} dx_{t-1}$$

$$\overline{bel}(x_t) = \begin{cases} \bar{\mu}_t = A_t \mu_{t-1} + B_t u_t \\ \bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + Q_t \end{cases}$$

Linear Gaussian Systems: Observations

Observations are a linear function of the state plus additive noise:

$$\boxed{z_t = C_t x_t + \delta_t}$$

$$\boxed{p(z_t \mid x_t) = N\left(z_t; C_t x_t, R_t\right)}$$

$$\boxed{\begin{array}{llll} bel(x_t) = & \eta & p(z_t \mid x_t) & \overline{bel}(x_t) \\ & & \Downarrow & \Downarrow \\ & & \sim N(z_t; C_t x_t, R_t) & \sim N\left(x_t; \bar{\mu}_t, \bar{\Sigma}_t\right) \end{array}}$$

Linear Gaussian Systems: Observations

$$
\begin{array}{lccc}
bel(x_t) = & \eta & p(z_t \mid x_t) & \overline{bel}(x_t) \\
 & & \Downarrow & \Downarrow \\
 & & \sim N(z_t; C_t x_t, R_t) & \sim N\left(x_t; \bar{\mu}_t, \bar{\Sigma}_t\right) \\
 & & \Downarrow &
\end{array}
$$

$$
bel(x_t) = \eta \exp\left\{-\frac{1}{2}(z_t - C_t x_t)^T R_t^{-1} (z_t - C_t x_t)\right\} \exp\left\{-\frac{1}{2}(x_t - \bar{\mu}_t)^T \bar{\Sigma}_t^{-1} (x_t - \bar{\mu}_t)\right\}
$$

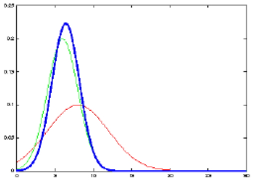
$$
bel(x_t) = \begin{cases} \mu_t = \bar{\mu}_t + K_t (z_t - C_t \bar{\mu}_t) \\ \Sigma_t = (I - K_t C_t) \bar{\Sigma}_t \end{cases} \quad \text{with} \quad K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + R_t)^{-1}
$$

Kalman Filter Algorithm

```
1. Algorithm **Kalman_filter**( $\mu_{t-1}$, $\Sigma_{t-1}$, $u_t$, $z_t$):

2.     Prediction:
3.         $\bar{\mu}_t = A_t \mu_{t-1} + B_t u_t$
4.         $\bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + Q_t$

5.     Correction:
6.         $K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + R_t)^{-1}$
7.         $\mu_t = \bar{\mu}_t + K_t (z_t - C_t \bar{\mu}_t)$
8.         $\Sigma_t = (I - K_t C_t) \bar{\Sigma}_t$

9.     Return $\mu_t$, $\Sigma_t$
```

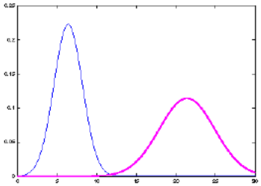
The Prediction-Correction-Cycle

Prediction

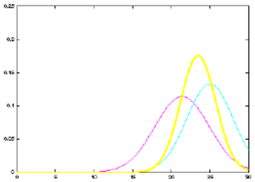


$$\overline{bel}(x_t) = \begin{cases} \bar{\mu}_t = a_t \mu_{t-1} + b_t u_t \\ \bar{\sigma}_t^2 = a_t^2 \sigma_t^2 + \sigma_{act,t}^2 \end{cases}$$

$$\overline{bel}(x_t) = \begin{cases} \bar{\mu}_t = A_t \mu_{t-1} + B_t u_t \\ \bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + Q_t \end{cases}$$

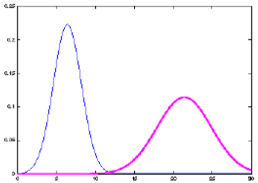


The Prediction-Correction-Cycle

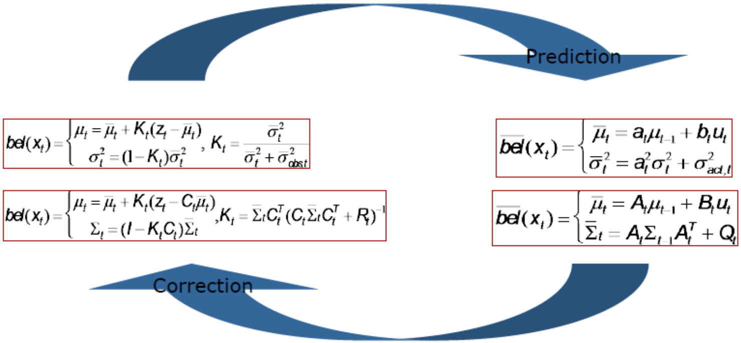


$$bel(x_t) = \begin{cases} \mu_t = \bar{\mu}_t + K_t(z_t - \bar{\mu}_t) \\ \sigma_t^2 = (1-K_t)\bar{\sigma}_t^2 \end{cases}, K_t = \frac{\bar{\sigma}_t^2}{\bar{\sigma}_t^2 + \sigma_{obs,t}^2}$$

$$bel(x_t) = \begin{cases} \mu_t = \bar{\mu}_t + K_t(z_t - C_t\bar{\mu}_t) \\ \Sigma_t = (I - K_t C_t)\bar{\Sigma}_t \end{cases}, K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + R_t)^{-1}$$



The Prediction-Correction-Cycle



$$\overline{bel}(x_t) = \begin{cases} \bar{\mu}_t = a_t \mu_{t-1} + b_t u_t \\ \bar{\sigma}_t^2 = a_t^2 \sigma_t^2 + \sigma_{act,t}^2 \end{cases}$$

$$\overline{bel}(x_t) = \begin{cases} \bar{\mu}_t = A_t \mu_{t-1} + B_t u_t \\ \bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + Q_t \end{cases}$$

$$bel(x_t) = \begin{cases} \mu_t = \bar{\mu}_t + K_t(z_t - \bar{\mu}_t) \\ \sigma_t^2 = (1 - K_t)\bar{\sigma}_t^2 \end{cases}, K_t = \frac{\bar{\sigma}_t^2}{\bar{\sigma}_t^2 + \sigma_{obs,t}^2}$$

$$bel(x_t) = \begin{cases} \mu_t = \bar{\mu}_t + K_t(z_t - C_t \bar{\mu}_t) \\ \Sigma_t = (I - K_t C_t)\bar{\Sigma}_t \end{cases}, K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + R_t)^{-1}$$

Kalman Filter Summary

- Only two parameters describe belief about the state of the system
- **Highly efficient:** Polynomial in the measurement dimensionality k and state dimensionality n:

$$O(k^{2.376} + n^2)$$

- **Optimal for linear Gaussian systems**!
- However: Most robotics systems are **nonlinear**!
- Can only model unimodal beliefs

Table of Contents

基础
- 概率论基础

递归状态估计
- 机器人环境交互
- 贝叶斯滤波

高斯滤波（Gaussian Filter）
- 卡尔曼滤波
- 扩展卡尔曼滤波
- 无迹卡尔曼滤波（Unscented Kalman Filter）

非参数滤波（Nonparametric Filters）
- 离散贝叶斯滤波（Discrete Bayes Filter）
- 粒子滤波（Particle Filter）

Nonlinear Dynamic Systems

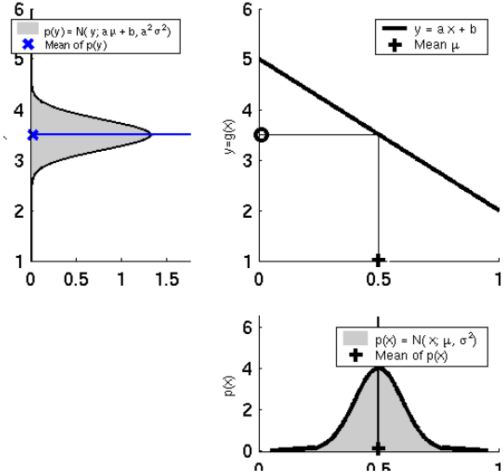
- Most realistic robotic problems involve nonlinear functions

$$\boxed{\xcancel{x_t = A_t x_{t-1} + B_t u_t + \varepsilon_t}} \qquad \xcancel{\boxed{z_t = C_t x_t + \delta_t}}$$

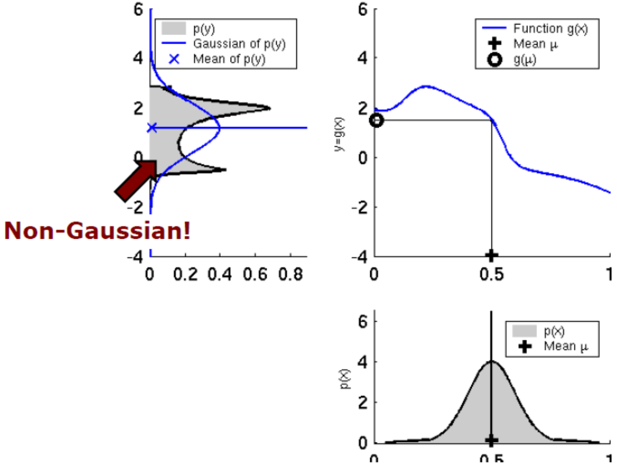
$$\Downarrow \qquad\qquad\qquad\qquad \Downarrow$$

$$\boxed{x_t = g(u_t, x_{t-1})} \qquad \boxed{z_t = h(x_t)}$$

Linearity Assumption Revisited



Non-Linear Function



Non-Gaussian Distributions

- The non-linear functions lead to non-Gaussian distributions
- Kalman filter is not applicable anymore!

What can be done to resolve this?

Local linearization!

EKF Linearization: First Order Taylor Expansion

- Prediction:

$$g(u_t, x_{t-1}) \approx g(u_t, \mu_{t-1}) + \frac{\partial g(u_t, \mu_{t-1})}{\partial x_{t-1}} (x_{t-1} - \mu_{t-1})$$

$$g(u_t, x_{t-1}) \approx g(u_t, \mu_{t-1}) + G_t\, (x_{t-1} - \mu_{t-1})$$

- Correction:

$$h(x_t) \approx h(\bar{\mu}_t) + \frac{\partial h(\bar{\mu}_t)}{\partial x_t} (x_t - \bar{\mu}_t)$$

$$h(x_t) \approx h(\bar{\mu}_t) + H_t\, (x_t - \bar{\mu}_t)$$

Jacobian matrices

Reminder: Jacobian Matrix

- It is a **non-square matrix** $n \times m$ in general
- Given a vector-valued function

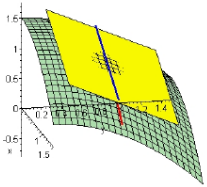
$$f(\mathbf{x}) = \begin{bmatrix} f_1(\mathbf{x}) \\ f_2(\mathbf{x}) \\ \vdots \\ f_m(\mathbf{x}) \end{bmatrix}$$

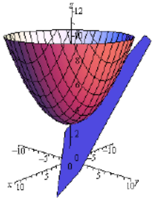
- The **Jacobian matrix** is defined as

$$\mathbf{F_x} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \cdots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}$$

Reminder: Jacobian Matrix

- It is the orientation of the tangent plane to the vector-valued function at a given point





- Generalizes the gradient of a scalar valued function

EKF Linearization: First Order Taylor Expansion

- Prediction:

$$g(u_t, x_{t-1}) \approx g(u_t, \mu_{t-1}) + \frac{\partial g(u_t, \mu_{t-1})}{\partial x_{t-1}} (x_{t-1} - \mu_{t-1})$$

$$g(u_t, x_{t-1}) \approx g(u_t, \mu_{t-1}) + G_t \, (x_{t-1} - \mu_{t-1})$$

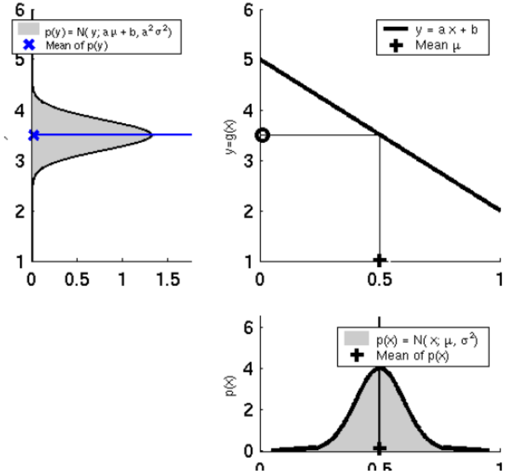
- Correction:

$$h(x_t) \approx h(\bar{\mu}_t) + \frac{\partial h(\bar{\mu}_t)}{\partial x_t} (x_t - \bar{\mu}_t)$$

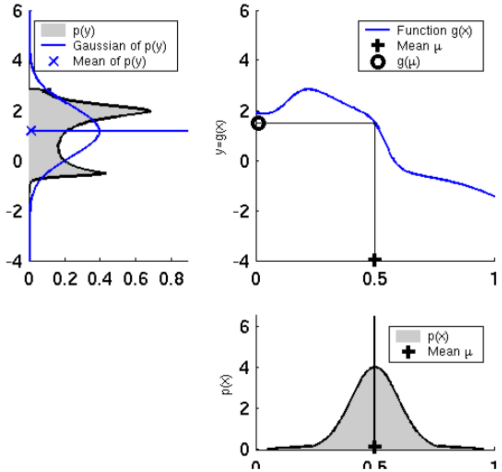
$$h(x_t) \approx h(\bar{\mu}_t) + H_t \, (x_t - \bar{\mu}_t)$$

Linear function!

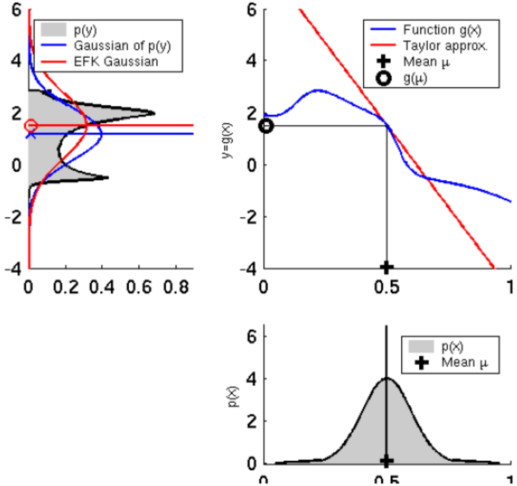
Linearity Assumption Revisited



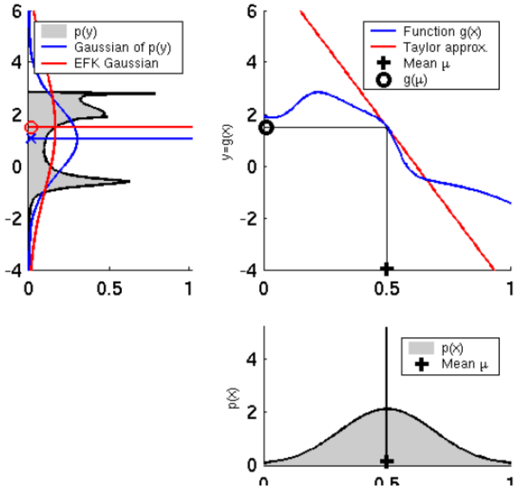
Non-Linear Function



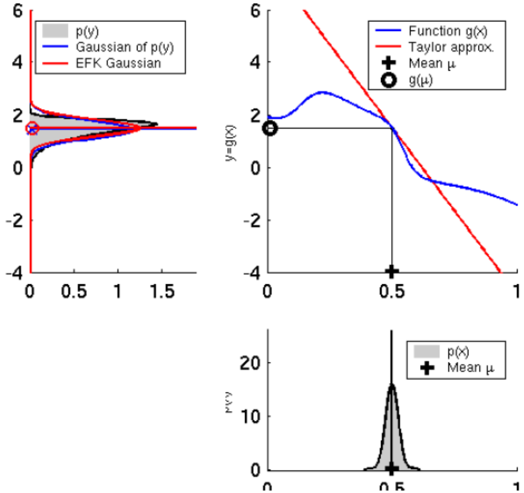
EKF Linearization (1)



EKF Linearization (2)



EKF Linearization (3)



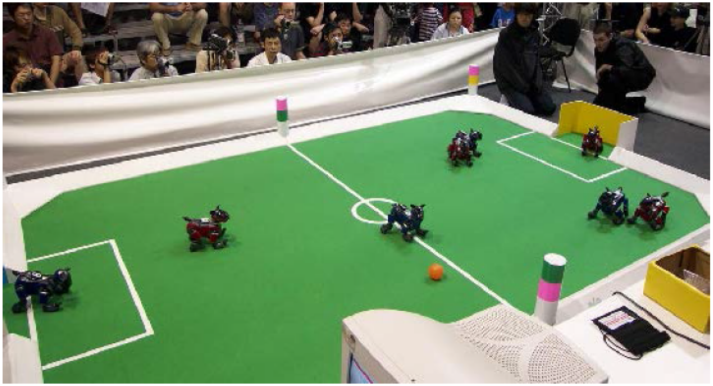
EKF Algorithm

```
1. **Extended_Kalman_filter**( $\mu_{t-1}$, $\Sigma_{t-1}$, $u_t$, $z_t$):

2.    Prediction:
3.    $\bar{\mu}_t = g(u_t, \mu_{t-1})$    $\longleftarrow$    $\bar{\mu}_t = A_t \mu_{t-1} + B_t u_t$
4.    $\bar{\Sigma}_t = G_t \Sigma_{t-1} G_t^T + Q_t$    $\longleftarrow$    $\bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + Q_t$

5.    Correction:
6.    $K_t = \bar{\Sigma}_t H_t^T (H_t \bar{\Sigma}_t H_t^T + R_t)^{-1}$    $\longleftarrow$    $K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + R_t)^{-1}$
7.    $\mu_t = \bar{\mu}_t + K_t (z_t - h(\bar{\mu}_t))$    $\longleftarrow$    $\mu_t = \bar{\mu}_t + K_t (z_t - C_t \bar{\mu}_t)$
8.    $\Sigma_t = (I - K_t H_t) \bar{\Sigma}_t$    $\longleftarrow$    $\Sigma_t = (I - K_t C_t) \bar{\Sigma}_t$

9.    Return $\mu_t$, $\Sigma_t$
```

$$H_t = \frac{\partial h(\bar{\mu}_t)}{\partial x_t} \qquad G_t = \frac{\partial g(u_t, \mu_{t-1})}{\partial x_{t-1}}$$

Example: EKF Localization

- EKF localization with landmarks (point features)



1. **EKF_localization** ($\mu_{t-1}, \Sigma_{t-1}, u_t, z_t, m$):

Prediction:

3. $G_t = \frac{\partial g(u_t, \mu_{t-1})}{\partial \mu_{t-1}} = \begin{pmatrix} \frac{\partial x'}{\partial \mu_{t-1,x}} & \frac{\partial x'}{\partial \mu_{t-1,y}} & \frac{\partial x'}{\partial \mu_{t-1,\theta}} \\ \frac{\partial y'}{\partial \mu_{t-1,x}} & \frac{\partial y'}{\partial \mu_{t-1,y}} & \frac{\partial y'}{\partial \mu_{t-1,\theta}} \\ \frac{\partial \theta'}{\partial \mu_{t-1,x}} & \frac{\partial \theta'}{\partial \mu_{t-1,y}} & \frac{\partial \theta'}{\partial \mu_{t-1,\theta}} \end{pmatrix}$ Jacobian of g w.r.t location

5. $V_t = \frac{\partial g(u_t, \mu_{t-1})}{\partial u_t} = \begin{pmatrix} \frac{\partial x'}{\partial v_t} & \frac{\partial x'}{\partial \omega_t} \\ \frac{\partial y'}{\partial v_t} & \frac{\partial y'}{\partial \omega_t} \\ \frac{\partial \theta'}{\partial v_t} & \frac{\partial \theta'}{\partial \omega_t} \end{pmatrix}$ Jacobian of g w.r.t control

1. $Q_t = \begin{pmatrix} (\alpha_1 |v_t| + \alpha_2 |\omega_t|)^2 & 0 \\ 0 & (\alpha_3 |v_t| + \alpha_4 |\omega_t|)^2 \end{pmatrix}$ Motion noise

2. $\bar{\mu}_t = g(u_t, \mu_{t-1})$ Predicted mean

3. $\bar{\Sigma}_t = G_t \Sigma_{t-1} G_t^T + V_t Q_t V_t^T$ Predicted covariance (V maps Q into state space)

1. **EKF_localization** ($\mu_{t-1}, \Sigma_{t-1}, u_t, z_t, m$):

Correction:

3. $\hat{z}_t = \begin{pmatrix} \sqrt{(m_x - \bar{\mu}_{t,x})^2 + (m_y - \bar{\mu}_{t,y})^2} \\ \operatorname{atan2}(m_y - \bar{\mu}_{t,y}, m_x - \bar{\mu}_{t,x}) - \bar{\mu}_{t,\theta} \end{pmatrix}$ Predicted measurement mean (depends on observation type)

5. $H_t = \frac{\partial h(\bar{\mu}_t, m)}{\partial x_t} = \begin{pmatrix} \frac{\partial r_t}{\partial \bar{\mu}_{t,x}} & \frac{\partial r_t}{\partial \bar{\mu}_{t,y}} & \frac{\partial r_t}{\partial \bar{\mu}_{t,\theta}} \\ \frac{\partial \phi_t}{\partial \bar{\mu}_{t,x}} & \frac{\partial \phi_t}{\partial \bar{\mu}_{t,y}} & \frac{\partial \phi_t}{\partial \bar{\mu}_{t,\theta}} \end{pmatrix}$ Jacobian of *h* w.r.t location

6. $R_t = \begin{pmatrix} \sigma_r^2 & 0 \\ 0 & \sigma_r^2 \end{pmatrix}$

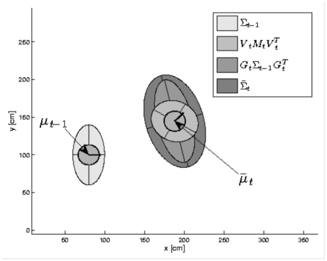
7. $S_t = H_t \bar{\Sigma}_t H_t^T + R_t$ Innovation covariance

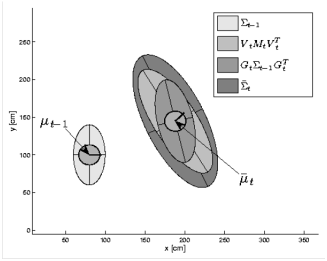
8. $K_t = \bar{\Sigma}_t H_t^T S_t^{-1}$ Kalman gain

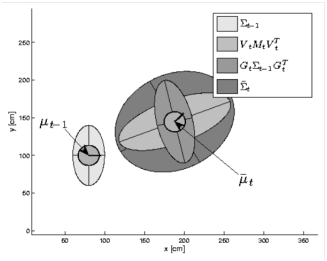
9. $\mu_t = \bar{\mu}_t + K_t (z_t - \hat{z}_t)$ Updated mean

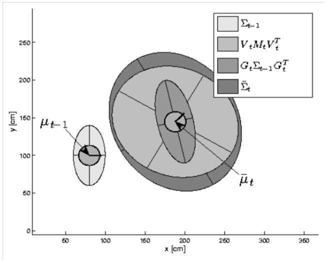
10. $\Sigma_t = (I - K_t H_t) \bar{\Sigma}_t$ Updated covariance

EKF Prediction Step

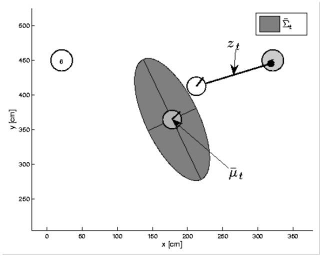


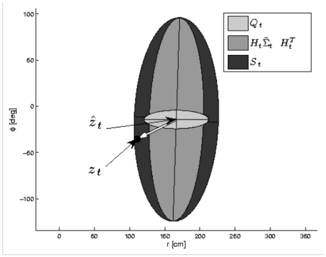


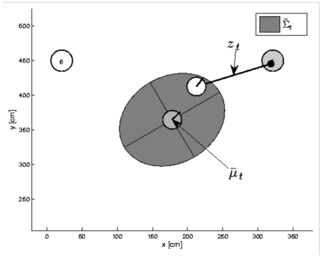


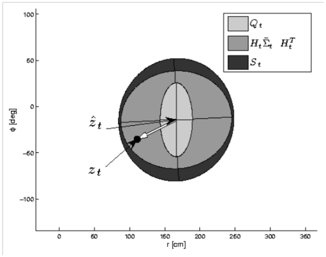


EKF Observation Prediction Step

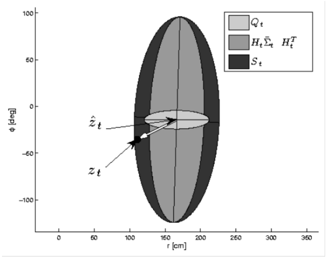


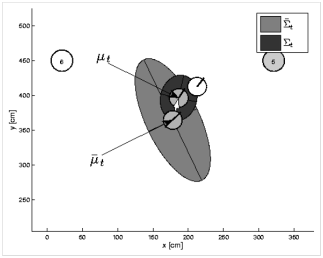


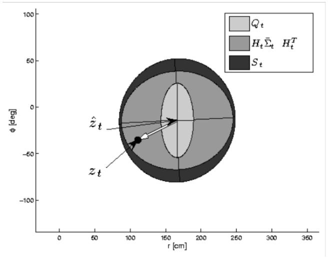


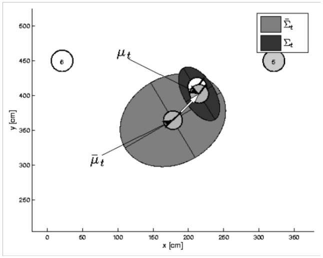


EKF Correction Step

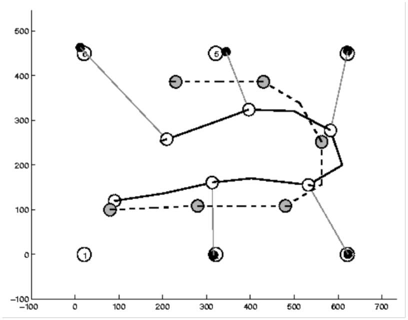


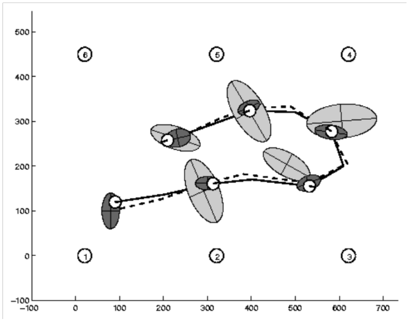




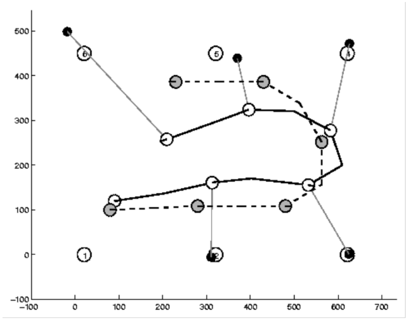


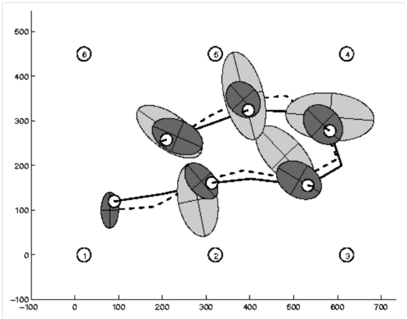
Estimation Sequence (1)



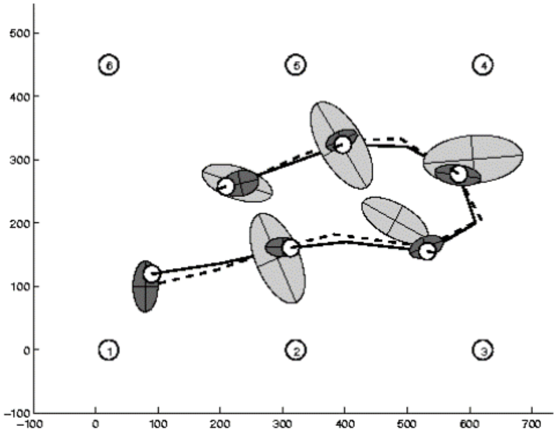


Estimation Sequence (2)





Comparison to GroundTruth



EKF Summary

- Ad-hoc solution to deal with non-linearities
- Performs local linearization in each step
- Works well in practice for moderate non-linearities
- Example: landmark localization
- There exist better ways for dealing with non-linearities such as the unscented Kalman filter called UKF

Table of Contents

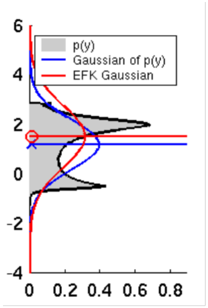
基础
概率论基础

递归状态估计
机器人环境交互
贝叶斯滤波

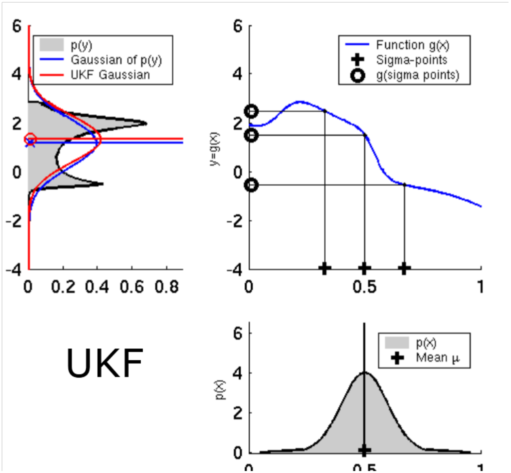
高斯滤波（Gaussian Filter）
卡尔曼滤波
扩展卡尔曼滤波
无迹卡尔曼滤波（Unscented Kalman Filter）

非参数滤波（Nonparametric Filters）
离散贝叶斯滤波（Discrete Bayes Filter）
粒子滤波（Particle Filter）

Linearization via Unscented Transform

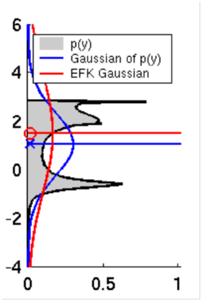


EKF

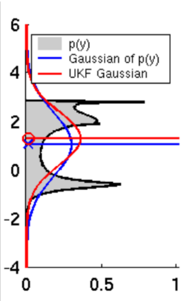


UKF

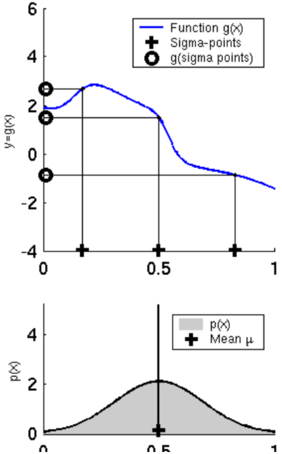
UKF Sigma-Point Estimate (2)



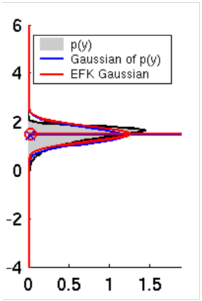
EKF



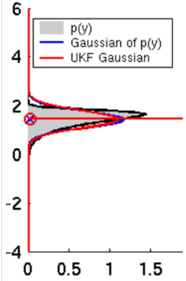
UKF



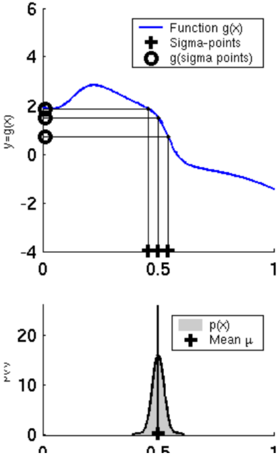
UKF Sigma-Point Estimate (3)



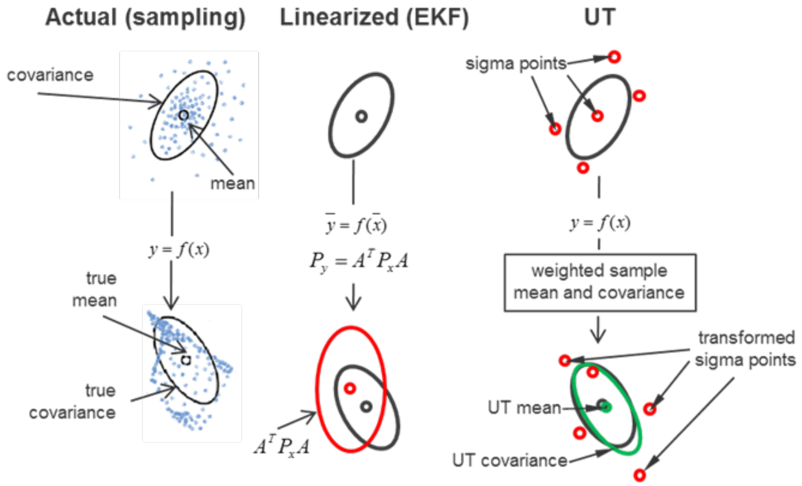
EKF



UKF



EKF vs. UKF



Actual (sampling)
Linearized (EKF)
UT
covariance
mean
$y = f(x)$
true mean
true covariance
$\bar{y} = f(\bar{x})$
$P_y = A^T P_x A$
$A^T P_x A$
sigma points
$y = f(x)$
weighted sample mean and covariance
transformed sigma points
UT mean
UT covariance

UKF_localization ($\mu_{t-1}, \Sigma_{t-1}, u_t, z_t, m$):

Prediction:

$$M_t = \begin{pmatrix} (\alpha_1 |v_t| + \alpha_2 |\omega_t|)^2 & 0 \\ 0 & (\alpha_3 |v_t| + \alpha_4 |\omega_t|)^2 \end{pmatrix}$$ Motion noise

$$Q_t = \begin{pmatrix} \sigma_r^2 & 0 \\ 0 & \sigma_r^2 \end{pmatrix}$$ Measurement noise

$$\mu_{t-1}^a = \begin{pmatrix} \mu_{t-1}^T & (0\ 0)^T & (0\ 0)^T \end{pmatrix}$$ Augmented state mean

$$\Sigma_{t-1}^a = \begin{pmatrix} \Sigma_{t-1} & 0 & 0 \\ 0 & M_t & 0 \\ 0 & 0 & Q_t \end{pmatrix}$$ Augmented covariance

$$\chi_{t-1}^a = \begin{pmatrix} \mu_{t-1}^a & \mu_{t-1}^a + \gamma \sqrt{\Sigma_{t-1}^a} & \mu_{t-1}^a - \gamma \sqrt{\Sigma_{t-1}^a} \end{pmatrix}$$ Sigma points

$$\bar{\chi}_t^x = g\left(u_t + \chi_t^u, \chi_{t-1}^x\right)$$ Prediction of sigma points

$$\bar{\mu}_t = \sum_{i=0}^{2L} w_m^i \, \chi_{i,t}^x$$ Predicted mean

$$\bar{\Sigma}_t = \sum_{i=0}^{2L} w_c^i \left(\chi_{i,t}^x - \bar{\mu}_t\right)\left(\chi_{i,t}^x - \bar{\mu}_t\right)^T$$ Predicted covariance

UKF_localization (μ_{t-1}, Σ_{t-1}, u_t, z_t, m):

Correction:

$\bar{Z}_t = h(\chi_t^x) + \chi_t^z$	Measurement sigma points
$\hat{z}_t = \sum_{i=0}^{2L} w_m^i \bar{Z}_{i,t}$	Predicted measurement mean
$S_t = \sum_{i=0}^{2L} w_c^i (\bar{Z}_{i,t} - \hat{z}_t)(\bar{Z}_{i,t} - \hat{z}_t)^T$	Pred. measurement covariance
$\Sigma_t^{x,z} = \sum_{i=0}^{2L} w_c^i (\bar{\chi}_{i,t}^x - \bar{\mu}_t)(\bar{Z}_{i,t} - \hat{z}_t)^T$	Cross-covariance
$K_t = \Sigma_t^{x,z} S_t^{-1}$	Kalman gain
$\mu_t = \bar{\mu}_t + K_t(z_t - \hat{z}_t)$	Updated mean
$\Sigma_t = \bar{\Sigma}_t - K_t S_t K_t^T$	Updated covariance

1. **EKF_localization** ($\mu_{t-1}, \Sigma_{t-1}, u_t, z_t, m$):

Correction:

3. $\hat{z}_t = \begin{pmatrix} \sqrt{(m_x - \bar{\mu}_{t,x})^2 + (m_y - \bar{\mu}_{t,y})^2} \\ \operatorname{atan2}(m_y - \bar{\mu}_{t,y}, m_x - \bar{\mu}_{t,x}) - \bar{\mu}_{t,\theta} \end{pmatrix}$ Predicted measurement mean

5. $H_t = \frac{\partial h(\bar{\mu}_t, m)}{\partial x_t} = \begin{pmatrix} \frac{\partial r_t}{\partial \bar{\mu}_{t,x}} & \frac{\partial r_t}{\partial \bar{\mu}_{t,y}} & \frac{\partial r_t}{\partial \bar{\mu}_{t,\theta}} \\ \frac{\partial \varphi_t}{\partial \bar{\mu}_{t,x}} & \frac{\partial \varphi_t}{\partial \bar{\mu}_{t,y}} & \frac{\partial \varphi_t}{\partial \bar{\mu}_{t,\theta}} \end{pmatrix}$ Jacobian of *h* w.r.t location

6. $Q_t = \begin{pmatrix} \sigma_r^2 & 0 \\ 0 & \sigma_r^2 \end{pmatrix}$

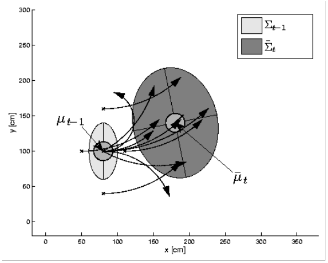
7. $S_t = H_t \bar{\Sigma}_t H_t^T + Q_t$ Pred. measurement covariance

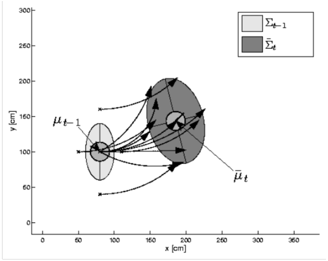
8. $K_t = \bar{\Sigma}_t H_t^T S_t^{-1}$ Kalman gain

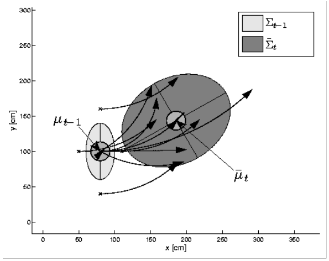
9. $\mu_t = \bar{\mu}_t + K_t (z_t - \hat{z}_t)$ Updated mean

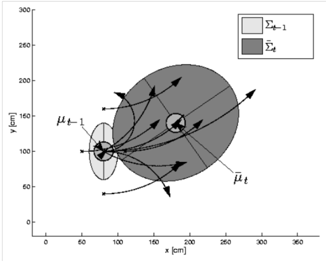
10. $\Sigma_t = (I - K_t H_t) \bar{\Sigma}_t$ Updated covariance

UKF Prediction Step

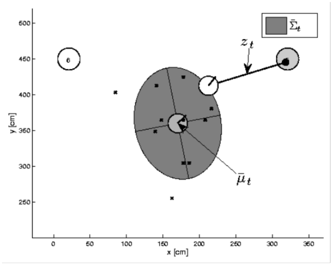


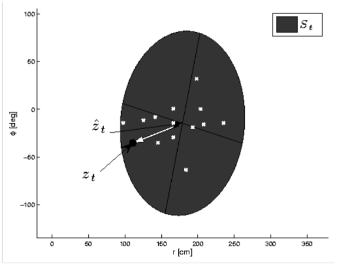


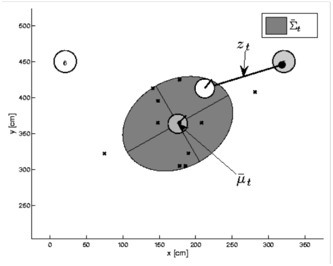


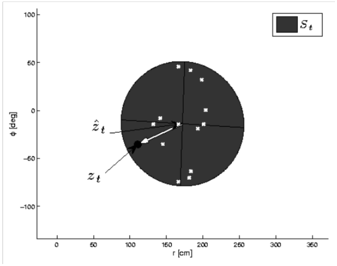


UKF Observation Prediction Step

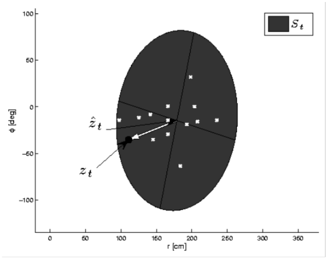


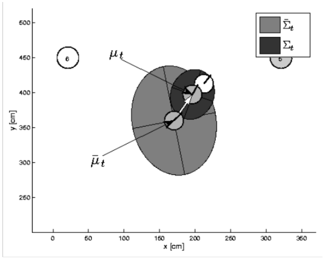


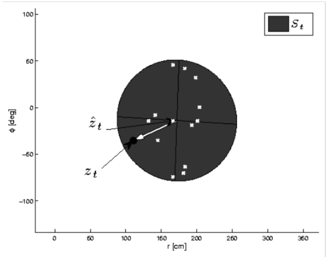


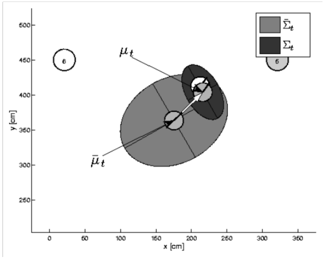


UKF Correction Step

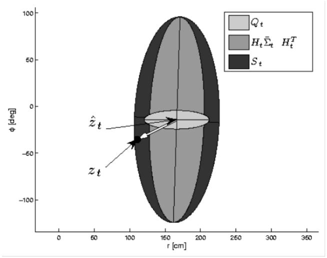


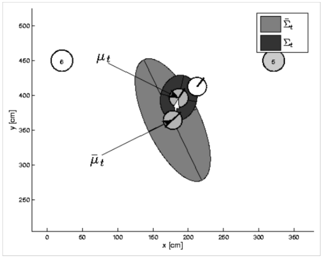


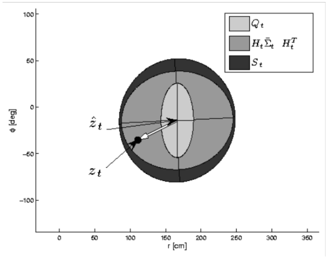


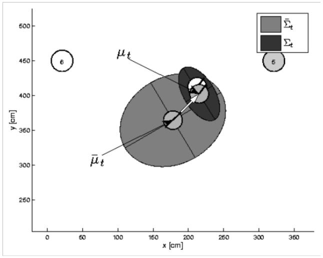


EKF Correction Step

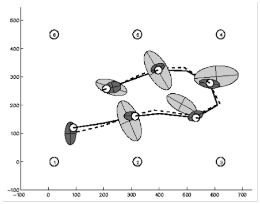




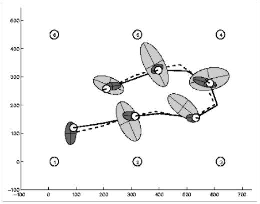




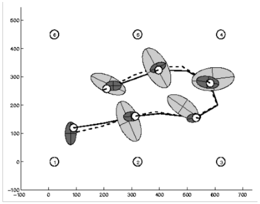
Estimation Sequence



EKF

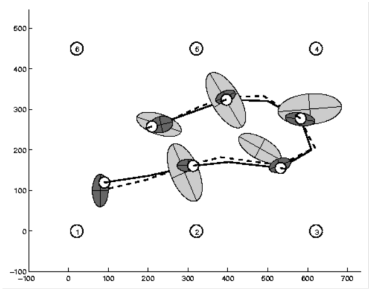


PF

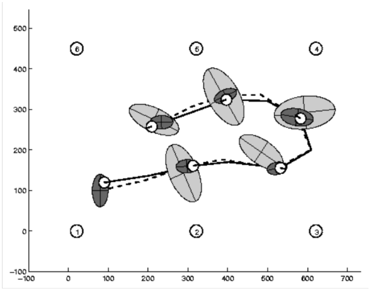


UKF

Estimation Sequence

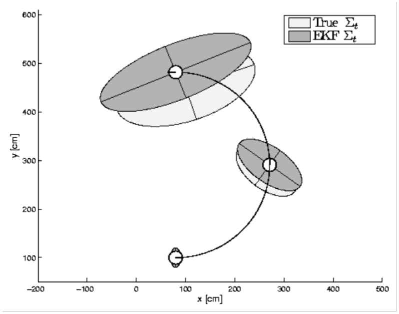


EKF

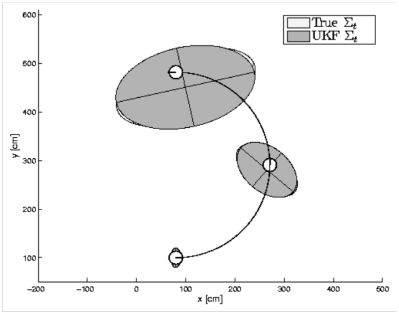


UKF

Prediction Quality



EKF



UKF

UKF Summary

- Highly efficient: Same complexity as EKF, with a constant factor slower in typical practical applications
- Better linearization than EKF: Accurate in first two terms of Taylor expansion (EKF only first term)
- Derivative-free: No Jacobians needed
- Still not optimal!

Table of Contents

基础
- 概率论基础

递归状态估计
- 机器人环境交互
- 贝叶斯滤波

高斯滤波（Gaussian Filter）
- 卡尔曼滤波
- 扩展卡尔曼滤波
- 无迹卡尔曼滤波（Unscented Kalman Filter）

非参数滤波（Nonparametric Filters）
- 离散贝叶斯滤波（Discrete Bayes Filter）
- 粒子滤波（Particle Filter）

非参数滤波

- 不同于高斯滤波，非参数滤波不依赖确定的后验函数，通过有限数量的值来近似后验，每一个值大致与状态空间的一个区域有关
  - 离散化：对状态空间进行分解，每一个值与状态空间的一个紧凑子区域的后验密度的累积概率相关
  - 采样：随机采样后验分布来近似状态空间
- 离散贝叶斯滤波，在连续空间下又称为直方图滤波 (Histogram Filter)：将状态空间分解为有限多个区域，并用直方图表示后验，一个直方图分配给一个区域一个单一的累积概率
- 粒子滤波：用有限多个样本表示后验
- 非参数滤波的表达能力是以增加计算复杂性为代价的

Table of Contents

基础
概率论基础

递归状态估计
机器人环境交互
贝叶斯滤波

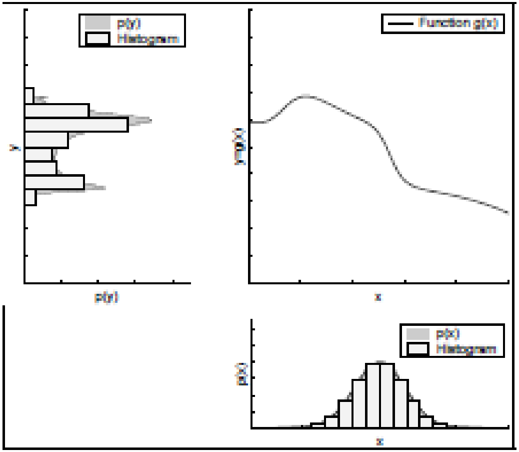
高斯滤波（Gaussian Filter）
卡尔曼滤波
扩展卡尔曼滤波
无迹卡尔曼滤波（Unscented Kalman Filter）

非参数滤波（Nonparametric Filters）
离散贝叶斯滤波（Discrete Bayes Filter）
粒子滤波（Particle Filter）

Discrete Bayes Filter Algorithm

```
1.   Algorithm **Discrete_Bayes_filter**( Bel(x),d ):
2.   $\eta$=0
3.   If d is a perceptual data item z then
4.       For all x do
5.           $Bel'(x) = P(z \mid x) Bel(x)$
6.           $\eta = \eta + Bel'(x)$
7.       For all x do
8.           $Bel'(x) = \eta^{-1} Bel'(x)$
9.   Else if d is an action data item u then
10.      For all x do
11.          $Bel'(x) = \sum_{x'} P(x \mid u, x')\, Bel(x')$
12.  Return Bel'(x)
```

连续情况



直方图滤波

- 直方图滤波将连续状态空间分解成有限区域：

$$\mathrm{dom}(X_t) = x_{1,t} \cup x_{2,t} \cup \cdots \cup x_{k,t}$$

  其中 X_t 为描述机器人状态在时刻 t 的随机变量。函数 $\mathrm{dom}(X_t)$ 为状态空间
- 离散贝叶斯滤波为每一个区域 $x_{k,t}$ 分配一个概率 $p_{k,t}$
- 后验成为一个分段常数概率密度函数，它在区域 $x_{k,t}$ 中的每一个状态 x_t 分配了相同的概率

$$p(x_t) = \frac{p_{k,t}}{|x_{k,t}|}$$

  其中 $|x_{k,t}|$ 为区域 $x_{k,t}$ 的绝对值
- 利用 $x_{k,t}$ 的平均状态进行探究

$$\hat{x}_{k,t} = |x_{k,t}|^{-1} \int_{x_{k,t}} x_t \, \mathrm{d}x_t$$

$$P(z_t \mid x_{k,t}) \approx P(z_t \mid \hat{x}_{k,t})$$

$$p(x_{k,t} \mid u_t, x_{i,t-1}) \approx \eta \, |x_{k,t}| \, P(\hat{x}_{k,t} \mid u_t, \hat{x}_{i,t-1})$$

Discrete Bayes Filter Summary

- Discrete filters are an alternative way for implementing Bayes Filters
- They are based on histograms for representing the density.
- They have huge memory and processing requirements
- Can easily recover from localization errors
- Their accuracy depends on the resolution of the grid.
- Special approximations need to be made to make this approach having dynamic memory and computational requirements.

Table of Contents

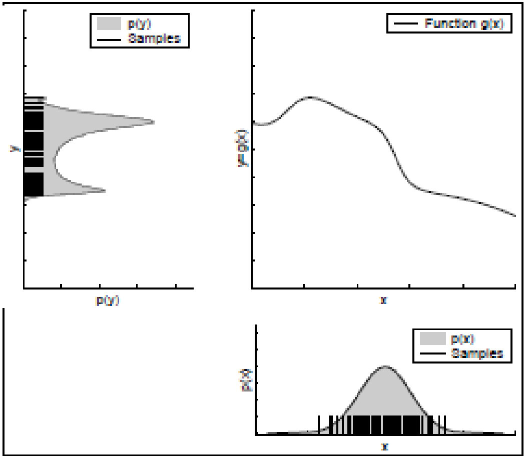
基础
概率论基础

递归状态估计
机器人环境交互
贝叶斯滤波

高斯滤波（Gaussian Filter）
卡尔曼滤波
扩展卡尔曼滤波
无迹卡尔曼滤波（Unscented Kalman Filter）

非参数滤波（Nonparametric Filters）
离散贝叶斯滤波（Discrete Bayes Filter）
粒子滤波（Particle Filter）

粒子表示法



Mathematical Description

- Set of weighted samples

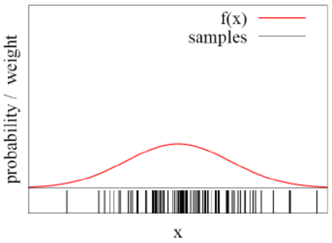
$$S = \left\{ \left\langle \underset{\substack{\uparrow \\ \text{State hypothesis}}}{s^{[i]}}, \underset{\substack{\uparrow \\ \text{Importance weight}}}{w^{[i]}} \right\rangle \mid i = 1, \ldots, N \right\}$$

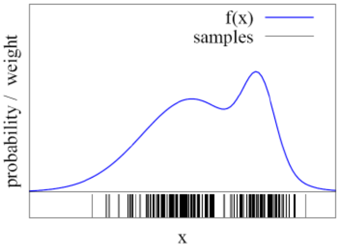
- The samples represent the posterior

$$p(x) = \sum_{i=1}^{N} w_i \cdot \delta_{s^{[i]}}(x)$$

Function Approximation

- Particle sets can be used to approximate functions





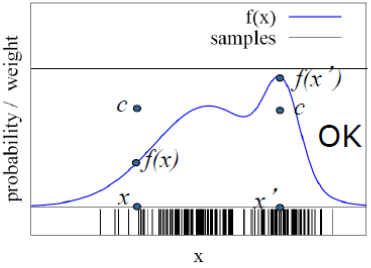
- The more particles fall into an interval, the higher the probability of that interval

- How to draw samples from a function/distribution?

Rejection Sampling

- Let us assume that $f(x)<1$ for all x
- Sample x from a uniform distribution
- Sample c from [0,1]
- if $f(x) > c$ keep the sample

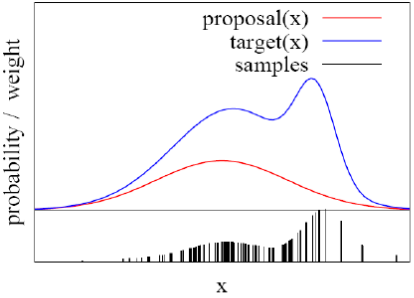
  otherwise reject the sample



Importance Sampling Principle

- We can even use a different distribution g to generate samples from f
- By introducing an importance weight w, we can account for the "differences between g and f"
- $w = f / g$
- f is called target
- g is called proposal
- Pre-condition:

  $f(x)>0 \rightarrow g(x)>0$



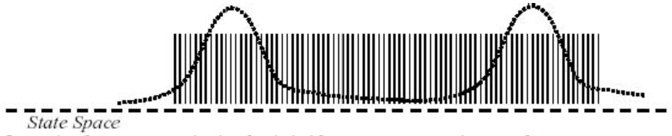
Importance Sampling

$$\boxed{\text{Target distribution f: } p(x \mid z_1, z_2, \ldots, z_n) = \frac{\prod_k p(z_k \mid x) \quad p(x)}{p(z_1, z_2, \ldots, z_n)}}$$

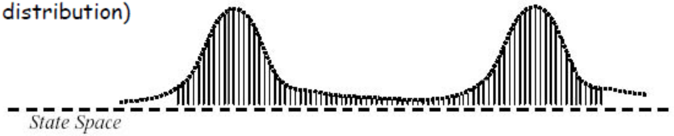
$$\boxed{\text{Sampling distribution g: } p(x \mid z_l) = \frac{p(z_l \mid x) p(x)}{p(z_l)}}$$

$$\boxed{\text{Importance weights w: } \frac{f}{g} = \frac{p(x \mid z_1, z_2, \ldots, z_n)}{p(x \mid z_l)} = \frac{p(z_l) \prod_{k \neq l} p(z_k \mid x)}{p(z_1, z_2, \ldots, z_n)}}$$

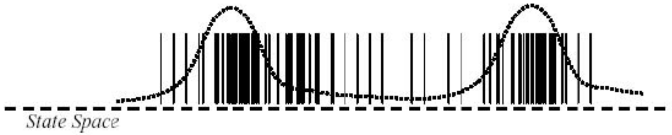
Importance Sampling with Resampling



Sample from prior belief q(x) (for instance, the uniform distribution)

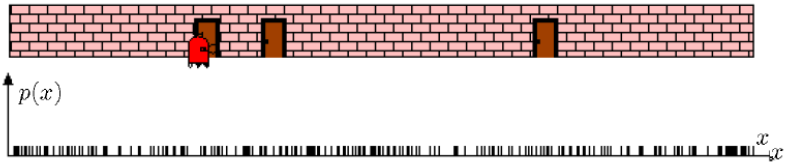


Compute importance weights, w(x) = p(x) /q(x)



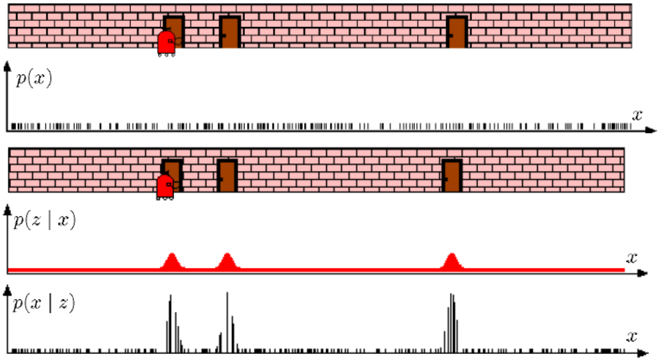
Resample particles according to importance weights to get p(x)
Samples with high weights chosen many times; density reflects pdf

Particle Filters



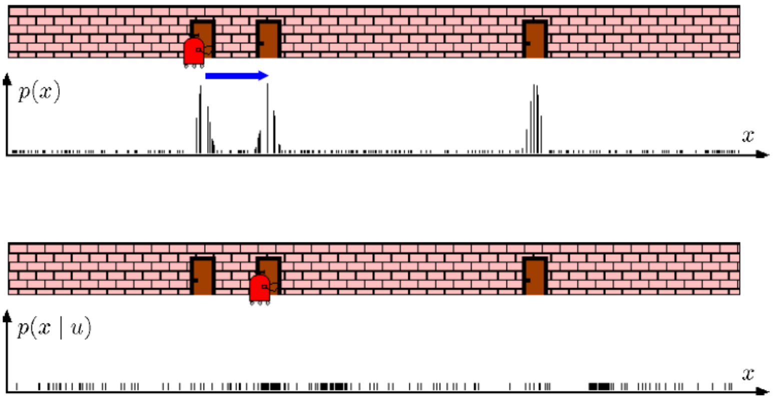
Sensor Information: Importance Sampling

$$\boxed{\begin{array}{lll} Bel(x) & \leftarrow & \alpha\, p(z \mid x)\, Bel^{-}(x) \\ w & \leftarrow & \dfrac{\alpha\, p(z \mid x)\, Bel^{-}(x)}{Bel^{-}(x)} \;=\; \alpha\, p(z \mid x) \end{array}}$$



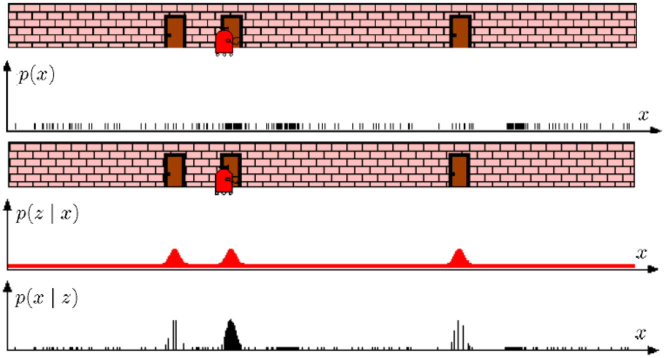
Robot Motion

$$\boxed{Bel^{-}(x) \leftarrow \int p(x \mid u, x')\, Bel(x')\, \mathrm{d}x'}$$



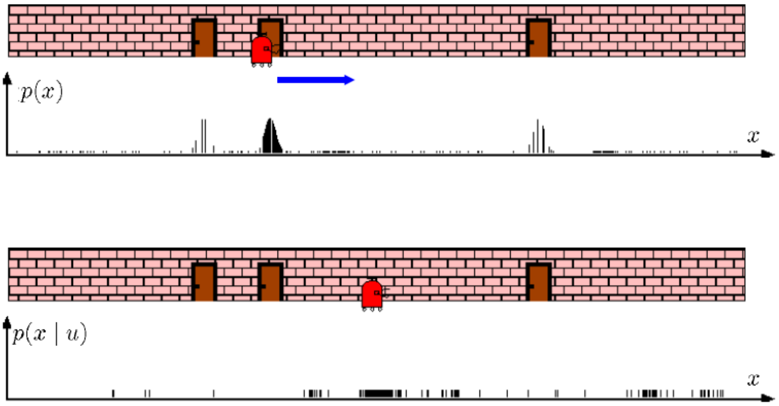
Sensor Information: Importance Sampling

$$\boxed{\begin{array}{lcll} Bel(x) & \leftarrow & \alpha\, p(z \mid x)\, Bel^{-}(x) & \\ w & \leftarrow & \dfrac{\alpha\, p(z \mid x)\, Bel^{-}(x)}{Bel^{-}(x)} & = \alpha\, p(z \mid x) \end{array}}$$



Robot Motion

$$Bel^{-}(x) \leftarrow \int p(x \mid u, x')\, Bel(x')\, \mathrm{d}x'$$



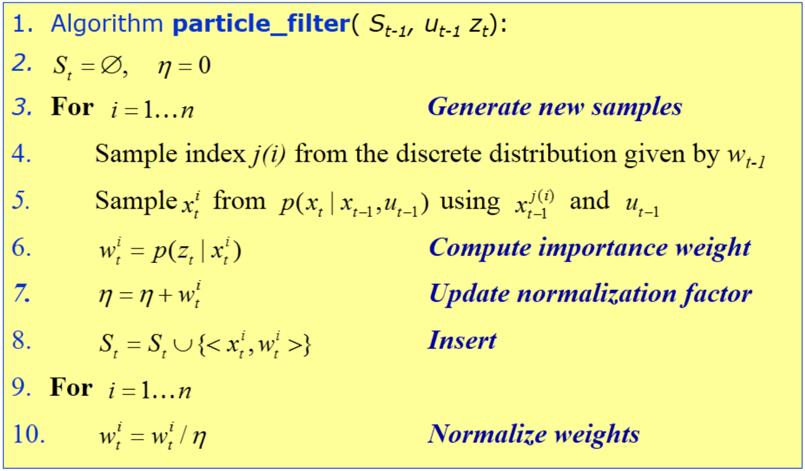
Particle Filter Algorithm

- Sample the next generation for particles using the proposal distribution
- Compute the importance weights :

  $$\textit{weight} = \textit{target distribution} \,/\, \textit{proposal distribution}$$

- Resampling: “Replace unlikely samples by more likely ones”

Particle Filter Algorithm



```
1.  Algorithm particle_filter( $S_{t-1}$, $u_{t-1}$ $z_t$):
2.  $S_t = \varnothing, \quad \eta = 0$
3.  For $i = 1 \ldots n$                                   Generate new samples
4.        Sample index $j(i)$ from the discrete distribution given by $w_{t-1}$
5.        Sample $x_t^i$ from $p(x_t \mid x_{t-1}, u_{t-1})$ using $x_{t-1}^{j(i)}$ and $u_{t-1}$
6.        $w_t^i = p(z_t \mid x_t^i)$                      Compute importance weight
7.        $\eta = \eta + w_t^i$                            Update normalization factor
8.        $S_t = S_t \cup \{< x_t^i, w_t^i >\}$            Insert
9.  For $i = 1 \ldots n$
10.       $w_t^i = w_t^i / \eta$                           Normalize weights
```

Particle Filter Algorithm

$$\boxed{Bel(x_t) = \eta\, p(z_t \mid x_t) \int p(x_t \mid x_{t-1}, u_t)\, Bel(x_{t-1})\, dx_{t-1}}$$

draw x^i_{t-1} from $Bel(x_{t-1})$

draw x^i_t from $p(x_t \mid x^i_{t-1}, u_t)$

Importance factor for x^i_t:

$$w^i_t = \frac{\text{target distribution}}{\text{proposal distribution}}$$

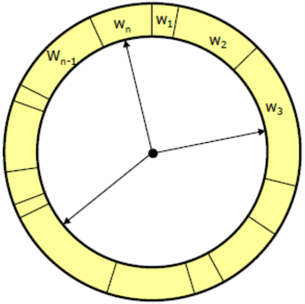
$$= \frac{\eta\, p(z_t \mid x_t)\, p(x_t \mid x_{t-1}, u_t)\, Bel\,(x_{t-1})}{p(x_t \mid x_{t-1}, u_t)\, Bel\,(x_{t-1})}$$

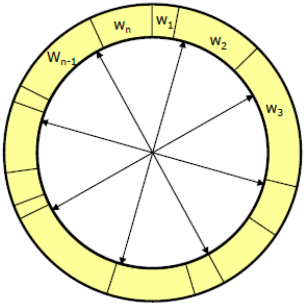
$$\propto p(z_t \mid x_t)$$

Resampling

- **Given**: Set S of weighted samples.
- **Wanted** : Random sample, where the probability of drawing x_i is given by w_i.
- Typically done n times with replacement to generate new sample set S'.

Resampling

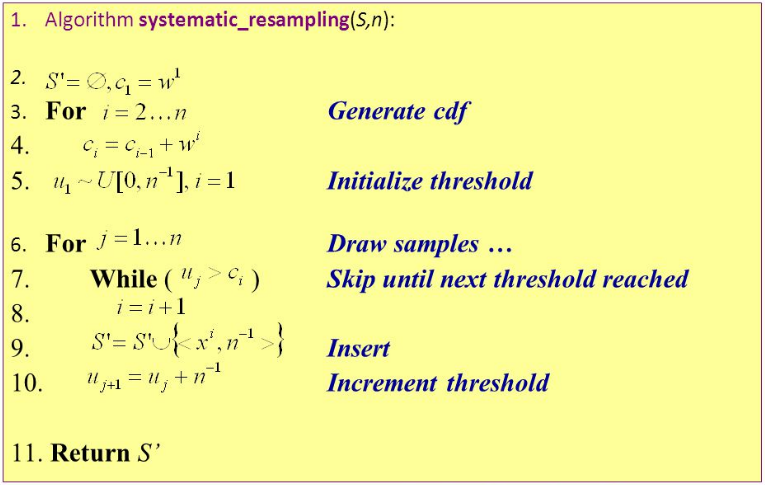




- Roulette wheel
- Binary search, n log n

- Stochastic universal sampling
- Systematic resampling
- Linear time complexity
- Easy to implement, low variance

Resampling Algorithm



```
1.  Algorithm systematic_resampling(S,n):

2.  $S' = \emptyset, c_1 = w^1$
3.  For $i = 2 \ldots n$                      Generate cdf
4.      $c_i = c_{i-1} + w^i$
5.  $u_1 \sim U[0, n^{-1}], i = 1$             Initialize threshold

6.  For $j = 1 \ldots n$                      Draw samples …
7.      While ( $u_j > c_i$ )                 Skip until next threshold reached
8.          $i = i + 1$
9.      $S' = S' \cup \{< x^i, n^{-1} >\}$    Insert
10.     $u_{j+1} = u_j + n^{-1}$              Increment threshold

11. Return S'
```

Also called **stochastic universal sampling**

Particle Filters Summary

- Particle filters are an implementation of recursive Bayesian filtering
- They represent the posterior by a set of weighted samples
- They can model non-Gaussian distributions
- Proposal to draw new samples
- Weight to account for the differences between the proposal and the target
- Monte Carlo filter, Survival of the fittest, Condensation, Bootstrap filter

小结

- 概率机器人：采用概率方法显示的处理机器人中的不确定性
- 贝叶斯滤波是动态环境下状态估计的主要概率工具
- 当观察模型和动态模型都是线性的，并且不确定性为高斯分布，则贝叶斯滤波可以简化为卡尔曼滤波
- 对于非线性模型的情况，可以采用扩展卡尔曼滤波 (EKF) 或无迹卡尔曼滤波 (UKF)，通常 UKF 的效果更好
- 对于高度非线性和非高斯分布的情况，可以采用粒子滤波，实际效果比 KF, EKF, UKF 效果好很多，但计算代价也更大