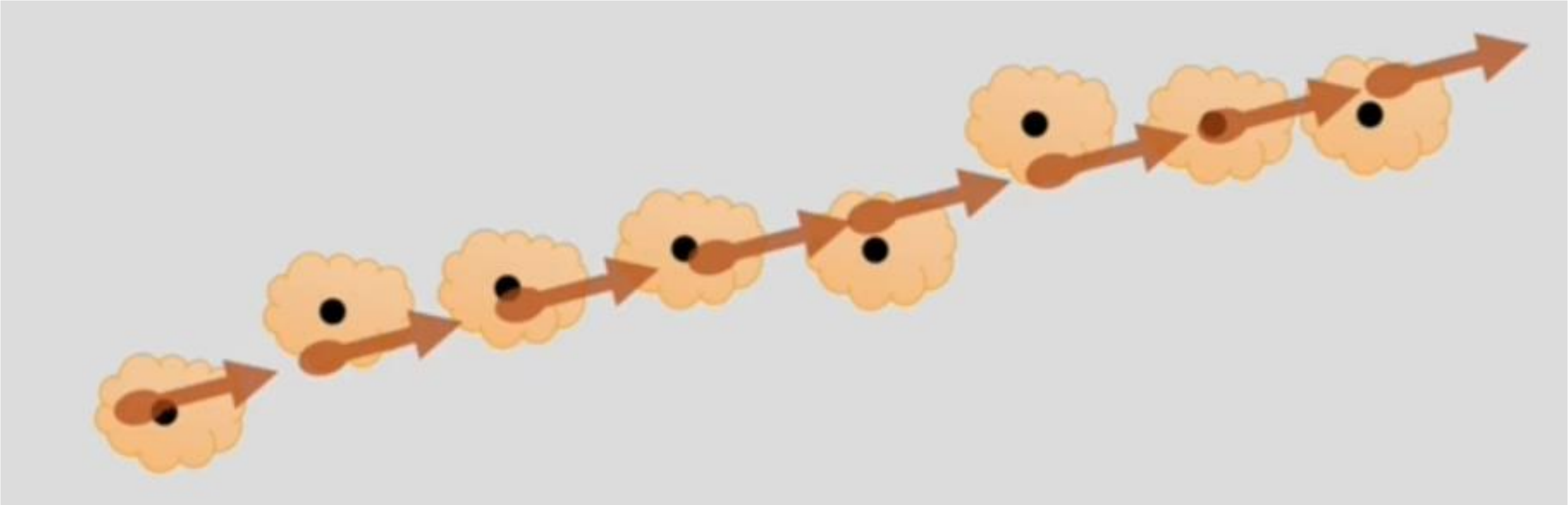
Машинное обучение в робототехнике

Фильтры для локализации роботов

Проблематика

- Имеется множество измерений движущегося объекта
- Все измерения зашумлены
- Где на самом деле находится объект?



Модель объекта

Движение описывается линейной дискретной системой

$$x_{t+1} = Ax_t + Bu_t$$
$$z_t = Cx_t$$

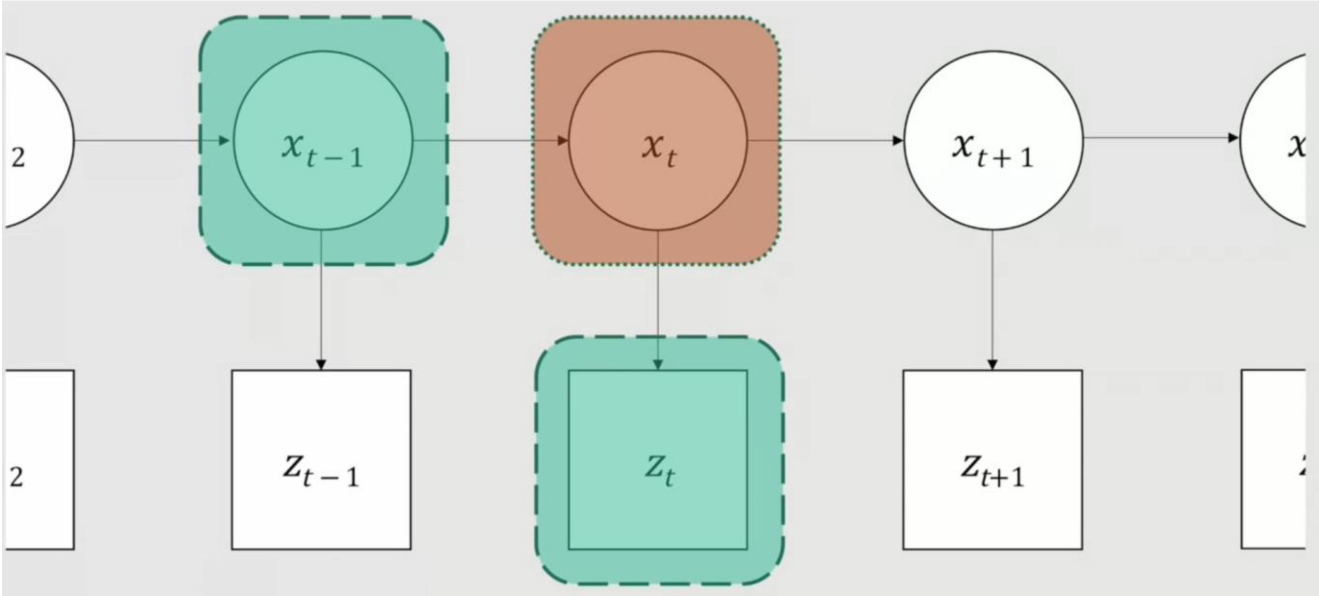
Вектор состояния:

$$x_{t+1}^T = [s \;\; ds/dt]$$

Динамика:

$$A = \begin{bmatrix} 1 & dt \\ 0 & 1 \end{bmatrix}$$

Модель объекта



- Требуется x_t
- Известен x_{t-1}
- Известен z_t

Вероятностная модель

- Предсказание с учетом состояния динамической модели
$$p(x_{t+1}|x_t)$$
- Учет зашумленности измерений
$$p(z_t|x_t)$$
- Моделирование x_t как гауссиана (мат. ожидание и ковариация)
$$p(x_t) = N(x_t, P_t)$$

Вероятностная модель

- Используем линейную динамическую модель

$$p(x_{t+1}|x_t) = Ap(x_t)$$
$$p(z_t|x_t) = Cp(x_t)$$

- Добавим шумы

$$p(x_{t+1}|x_t) = Ap(x_t) + v_m$$
$$p(z_t|x_t) = Cp(x_t) + v_o$$

- Гауссиан x_t:

$$p(x_{t+1}|x_t) = AN(x_t, P_t) + N(0, \Sigma_m)$$
$$p(z_t|x_t) = CN(x_t, P_t) + N(0, \Sigma_o)$$

Вероятностная модель

$$p(x_{t+1}|x_t) = AN(x_t, P_t) + N(0, \Sigma_m)$$
$$p(z_t|x_t) = CN(x_t, P_t) + N(0, \Sigma_o)$$

Выполним линейное преобразование распределений Гаусса:

$$p(x_{t+1}|x_t) = N(Ax_t, AP_tA^T) + N(0, \Sigma_m)$$
$$p(z_t|x_t) = N(Cx_t, CP_tC^T) + N(0, \Sigma_o)$$

Просуммируем гауссианы:

$$p(x_{t+1}|x_t) = N(Ax_t, AP_tA^T + \Sigma_m)$$
$$p(z_t|x_t) = N(Cx_t, CP_tC^T + \Sigma_o)$$

Максимизация апостериорной оценки

- Применить подход максимальной апостериорной (MAP) оценки по правилу Байеса
- Решить задачу максимизации
- Метод обновления параметров на основе фильтра Калмана

Байесовская фильтрация

- Правило Байеса:

$$p(\alpha|\beta) = \frac{p(\beta|\alpha)p(\alpha)}{p(\beta)}$$

- Вероятностная модель

$$p(x_{t+1}|x_t) = N(Ax_t, AP_tA^T + \Sigma_m)$$
$$p(z_t|x_t) = N(Cx_t, CP_tC^T + \Sigma_o)$$

Байесовская фильтрация

- Правило Байеса:

$$p(\alpha|\beta) = \frac{p(\beta|\alpha)p(\alpha)}{p(\beta)}$$

- Вероятностная модель

$$p(x_{t+1}|x_t) = N(Ax_t, AP_tA^T + \Sigma_m) \rightarrow \alpha$$ априорная

$$p(z_t|x_t) = N(Cx_t, CP_tC^T + \Sigma_o) \rightarrow \beta|\alpha$$ правдоподобие

постериорная $$p(x_t|z_t, x_{t-1}) = \frac{p(z_t|x_t, x_{t-1})p(x_t|x_{t-1})}{p(z_t)}$$

Байесовская фильтрация

- Постериорное распределение – гауссиан
- MAP оценивает «оптимальное» x_t
- Использование MAP дает новое математическое ожидание и дисперсию состояния

MAP

Задача:

$$\hat{x}_t = \arg\max_{x_t} p(x_t|z_t, x_{t-1})$$

$$\hat{x}_t = \arg\max_{x_t} \frac{p(z_t|x_t)p(x_t|x_{t-1})}{p(z_t)}$$

$$\hat{x}_t = \arg\max_{x_t} p(z_t|x_t)p(x_t|x_{t-1})$$

$$\hat{x}_t = \arg\max_{x_t} N(Cx_t, CP_tC^T + \Sigma_0)N(Ax_{t-1}, AP_{t-1}A^T + \Sigma_m)$$

MAP

$$\hat{x}_t = \arg\max_{x_t} N(Cx_t, CP_tC^T + \Sigma_0)N(Ax_{t-1}, AP_{t-1}A^T + \Sigma_m)$$

Введем замену переменных:

$$P = P_t = AP_{t-1}A^T + \Sigma_m$$
$$R = CP_tC^T + \Sigma_0$$

Возьмем логарифм и перейдем к задаче минимизации

$$\hat{x}_t = \arg\min_{x_t} \begin{matrix} (z_t - Cx_t)R^{-1}(z_t - Cx_t)^T + \\ +(x_t - Ax_{t-1})P^{-1}(x_t - Ax_{t-1})^T \end{matrix}$$

MAP

Аналитическое решение:

$$\hat{x}_t = \arg\min_{x_t} \begin{matrix} (z_t - Cx_t)R^{-1}(z_t - Cx_t)^T + \\ +(x_t - Ax_{t-1})P^{-1}(x_t - Ax_{t-1})^T \end{matrix}$$

$$0 = \frac{d}{dx_t}\begin{pmatrix} (z_t - Cx_t)R^{-1}(z_t - Cx_t)^T + \\ +(x_t - Ax_{t-1})P^{-1}(x_t - Ax_{t-1})^T \end{pmatrix}$$

MAP

Приведем подобные

$$(C^T R^{-1} C + P^{-1}) x_t = z_t^T R^{-1} C + P^{-1} A x_{t-1}$$

$$x_t = (C^T R^{-1} C + P^{-1})^{-1} (z_t^T R^{-1} C + P^{-1} A x_{t-1})$$

С учетом леммы об обратной матрице:

$$(C^T R^{-1} C + P^{-1})^{-1} = P - P C^T (R + C P C^T)^{-1} C P$$

Обозначим множитель Калмана:

$$K = P C^T (R + C P C^T)^{-1}$$

Фильтр Калмана

Результат:

$$K = PC^T R^{-1} - KCPC^T R$$
$$\hat{x}_t = Ax_{t-1} + K(z_t - CAx_{t-1})$$

Обновление ковариации:

$$\hat{P}_t = P - KCP$$

При наличии управления:

$$\hat{x}_t = Ax_{t-1} + K(z_t - CAx_{t-1}) + (I - KC)Bu_t$$

Фильтр Калмана

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

$$z_t = C_t x_t + \delta_t$$

R_t- коварияция ε_t

Q_t- коварияция δ_t

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

2: $\bar{\mu}_t = A_t\, \mu_{t-1} + B_t\, u_t$

3: $\bar{\Sigma}_t = A_t\, \Sigma_{t-1}\, A_t^T + R_t$

4: $K_t = \bar{\Sigma}_t\, C_t^T (C_t\, \bar{\Sigma}_t\, C_t^T + Q_t)^{-1}$

5: $\mu_t = \bar{\mu}_t + K_t(z_t - C_t\, \bar{\mu}_t)$

6: $\Sigma_t = (I - K_t\, C_t)\, \bar{\Sigma}_t$

7: return μ_t, Σ_t

Расширенный фильтр Калмана (EKF)

Нелинейная система

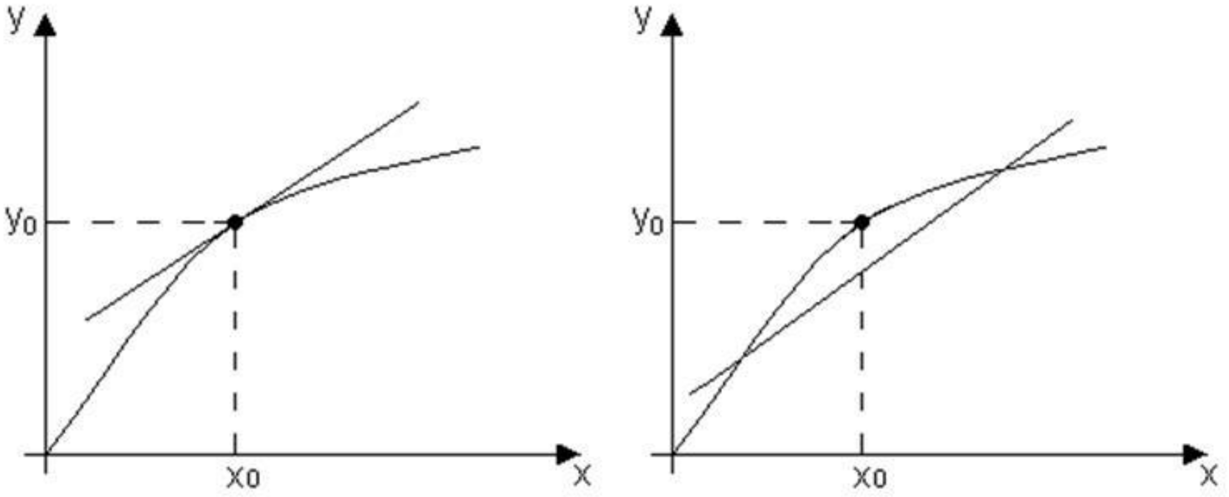
$$x_t = g(u_t, x_{t-1}) + \varepsilon_t$$
$$z_t = h(x_t) + \delta_t$$

EKF. Линеаризация

Нелинейность динамики

Обозначим

$$g'(u_t, x_{t-1}) = \frac{\partial g(u_t, x_{t-1})}{\partial x_{t-1}}$$



Разложим в ряд Тейлора

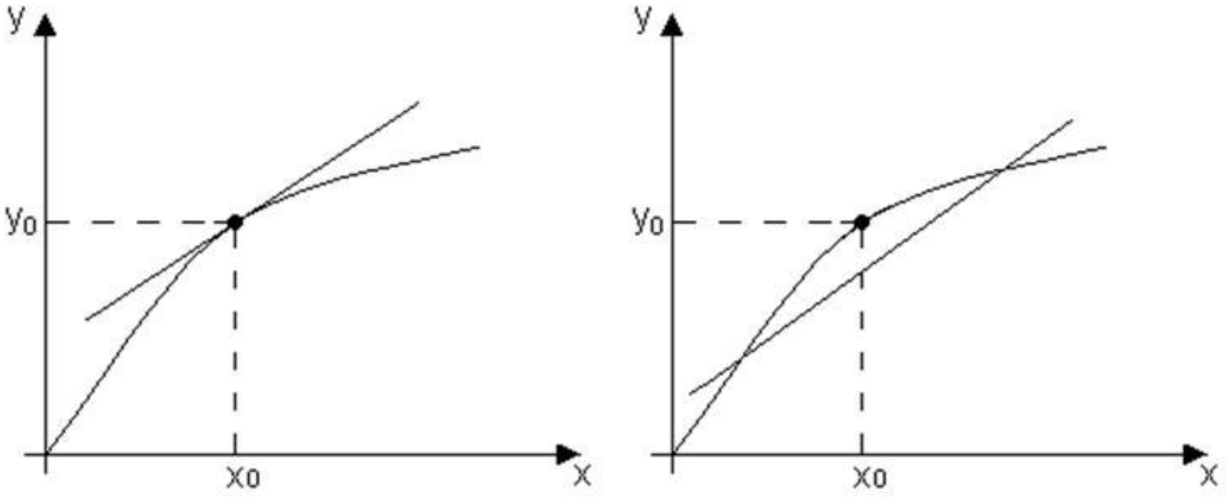
$$\mathrm{g}(u_t, x_{t-1}) \approx g(u_t, \mu_{t-1}) + g'(u_t, \mu_{t-1})(x_{t-1} - \mu_{t-1}) =$$
$$g(u_t, \mu_{t-1}) + G_t(x_{t-1} - \mu_{t-1})$$

EKF. Линеаризация

Нелинейность измерений

Обозначим

$$h'(x_t) = \frac{\partial h(x_t)}{\partial x_t}$$



Разложим в ряд Тейлора

$$h(x_t) \approx h(\bar{\mu}_t) + h'(x_t)(x_t - \bar{\mu}_t) = \\ h(\bar{\mu}_t) + H_t(x_{t-1} - \bar{\mu}_t)$$

EKF. Алгоритм

1: **Algorithm Extended_Kalman_filter**($\mu_{t-1}, \Sigma_{t-1}, u_t, z_t$)**:**
2: $\bar{\mu}_t = g(u_t, \mu_{t-1})$
3: $\bar{\Sigma}_t = G_t\ \Sigma_{t-1}\ G_t^T + R_t$
4: $K_t = \bar{\Sigma}_t\ H_t^T (H_t\ \bar{\Sigma}_t\ H_t^T + Q_t)^{-1}$
5: $\mu_t = \bar{\mu}_t + K_t(z_t - h(\bar{\mu}_t))$
6: $\Sigma_t = (I - K_t\ H_t)\ \bar{\Sigma}_t$
7: return μ_t, Σ_t

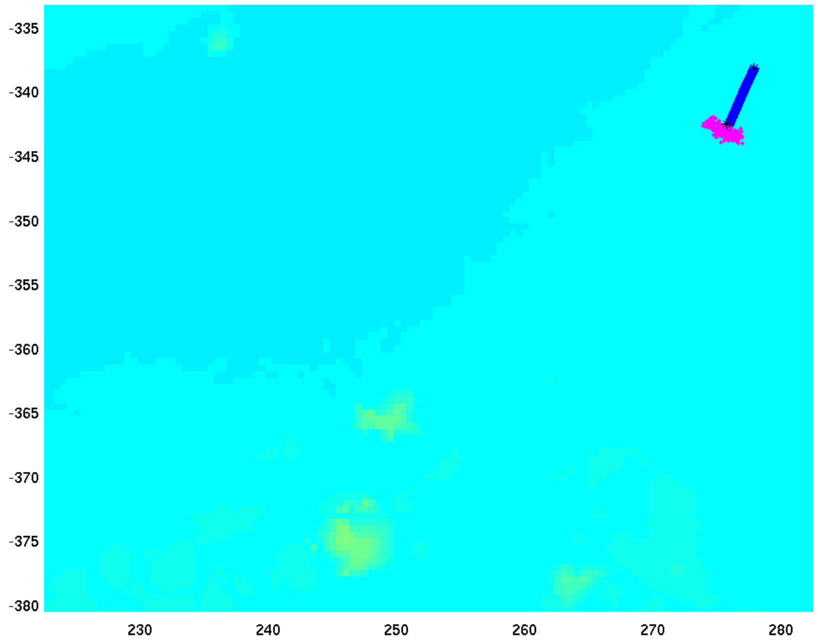
	Kalman filter	EKF
state prediction (Line 2)	$A_t\ \mu_{t-1} + B_t\ u_t$	$g(u_t, \mu_{t-1})$
measurement prediction (Line 5)	$C_t\ \bar{\mu}_t$	$h(\bar{\mu}_t)$

Фильтр частиц

Частицы:

$$\chi_t = x_t^{[1]}, x_2^{[t]}, \dots, x_t^{[m]}, \dots, x_t^{[M]}$$

$x_t^{[m]}$ $(1 \leq m \leq M)$ – гипотеза о состоянии x в момент времени t



Фильтр частиц

$$x_t^{[m]} \sim p(x_t | z_{1:t}, u_{1:t})$$

Algorithm Particle_filter($\mathcal{X}_{t-1}, u_t, z_t$):

$\bar{\mathcal{X}}_t = \mathcal{X}_t = \emptyset$

for $m = 1$ to M do

sample $x_t^{[m]} \sim p(x_t \mid u_t, x_{t-1}^{[m]})$

$w_t^{[m]} = p(z_t \mid x_t^{[m]})$

$\bar{\mathcal{X}}_t = \bar{\mathcal{X}}_t + \langle x_t^{[m]}, w_t^{[m]} \rangle$

endfor

for $m = 1$ to M do

draw i with probability $\propto w_t^{[i]}$

add $x_t^{[i]}$ to $\mathcal{X}_t$

endfor

return $\mathcal{X}_t$

Фильтр частиц

Альтернатива отсеву

```
Algorithm Particle_filter(𝒳_{t−1}, u_t, z_t):
    X̄_t = 𝒳_t = ∅
    for m = 1 to M do
        sample x_t^[m] ~ p(x_t | u_t, x_{t−1}^[m])
        w_t^[m] = p(z_t | x_t^[m])
        X̄_t = X̄_t + ⟨x_t^[m], w_t^[m]⟩
    endfor
    ~~for m = 1 to M do~~
        ~~draw i with probability ∝ w_t^[i]~~
        ~~add x_t^[i] to 𝒳_t~~
    ~~endfor~~
    return 𝒳_t
```

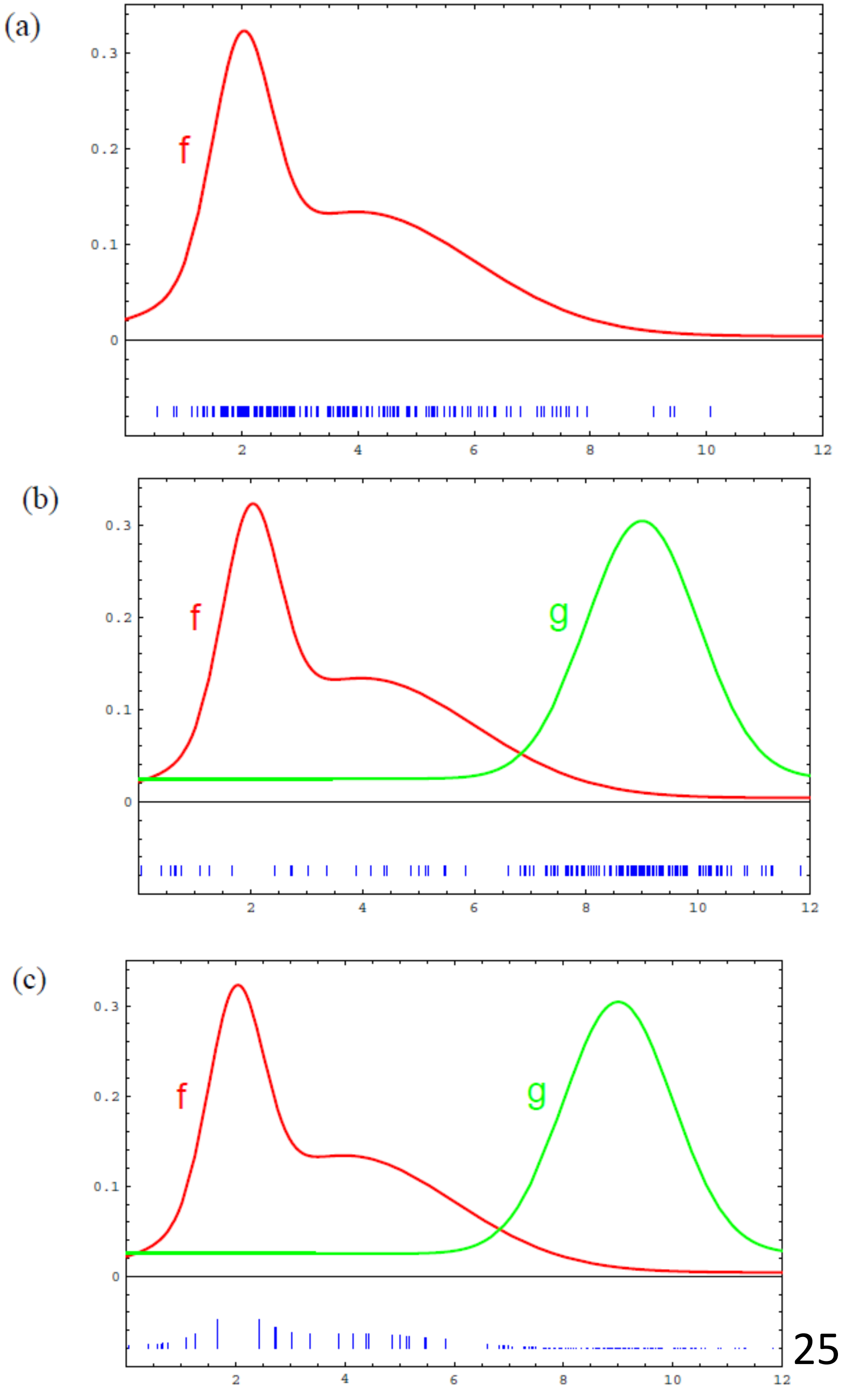
$$w_t^{[m]} = p\left(z_t \middle| x_t^{[m]}\right) w_{t-1}^{[m]}$$

Фильтр частиц. Веса выборки

$$\frac{1}{M}\sum_{m=1}^{M} I\left(x^{[m]} \in A\right) \rightarrow \int_A g(x)dx$$

$$w^{[m]} = \frac{f(x^{[m]})}{g(x^{[m]})}$$

$$\left[\sum_{m=1}^{M} w^{[m]}\right]^{-1} \sum_{m=1}^{M} I\left(x^{[m]} \in A\right)w^{[m]} \rightarrow \int_A f(x)dx$$



Фильтр частиц. Математика

$$x_{0:t}^{[m]} = x_0^{[m]}, x_1^{[m]}, \dots, x_t^{[m]}$$

$bel(x_{0:t}) = p(x_{0:t}|u_{1:t}, z_{1:t})$ вместо $bel(x_t) = p(x_{0:t}|u_{1:t}, z_{1:t})$

$$p(x_{0:t}|u_{1:t}, z_{1:t}) =^{Bayes}= \eta p(z_t|x_{0:t}, z_{1:t-1}, u_{1:t}) p(x_{0:t}|z_{1:t-1}, u_{1:t})$$

$$=^{Markov}= \eta p(z_t|x_t) p(x_{0:t}|z_{1:t-1}, u_{1:t})$$

$$= \eta p(z_t|x_t) p(x_t|x_{0:t-1}, z_{1:t-1}, u_{1:t})\, p(x_{0:t-1}|z_{1:t-1}, u_{1:t}) =$$

$$=^{Markov}= \eta p(z_t|x_t) p(x_t|x_{t-1}, u_t) p(x_{0:t-1}|u_{1:t-1}, z_{1:t-1})$$

Фильтр частиц. Математика

В момент времени $t-1$ предполагаемое $x_t^{[m]}$ в соответствии с алгоритмом:

$$p(x_t|x_{t-1},u_t)bel(x_{0:t-1}) = p(x_t|x_{t-1},u_t)p(x_{0:t-1}|z_{0:t-1},u_{0:t-1})$$

$$w^{[m]} = \frac{target\ distribution}{proposal\ distribution} = \frac{\eta p(z_t|x_t)p(x_t|x_{t-1},u_t)p(x_{0:t-1}|u_{1:t-1},z_{1:t-1})}{p(x_t|x_{t-1},u_t)p(x_{0:t-1}|z_{0:t-1},u_{0:t-1})} = \eta p(z_t|x_t)$$

Фильтр частиц. Математика

$$\eta w_t^{[m]} p(x_t|x_{t-1}, u_t) p(x_{0:t-1}|z_{o:t-1}, u_{0:t-1}) = bel(x_{o:t})$$

Фильтр частиц. Ошибки

- Конечное количество частиц
- Случайность повторной выборки

Фильтр частиц. Ошибки

- Конечное количество частиц
- Случайность отсева

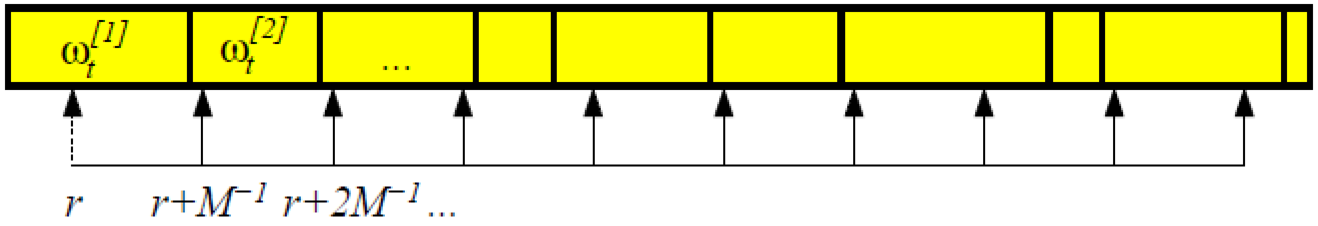
$$w_t^{[m]} \begin{cases} 1 \text{ при повторной выборке} \\ p\left(z_t \middle| x_t^{[m]}\right) w_{t-1}^m \text{ без повторной выборки} \end{cases}$$

Фильтр частиц. Ошибки

- Конечное количество частиц
- Случайность отсева

Algorithm Low_variance_sampler($\mathcal{X}_t, \mathcal{W}_t$):

$\bar{\mathcal{X}}_t = \emptyset$
$r = \mathrm{rand}(0; M^{-1})$
$c = w_t^{[1]}$
$i = 1$
for $m = 1$ to M do
  $u = r + (m-1) \cdot M^{-1}$
  while $u > c$
    $i = i + 1$
    $c = c + w_t^{[i]}$
  endwhile
  add $x_t^{[i]}$ to $\bar{\mathcal{X}}_t$
endfor
return $\bar{\mathcal{X}}_t$



$$i = \underset{j}{\operatorname{argmin}} \sum_{m=1}^{j} w_t^{[m]} \geq u$$

Фильтр частиц. Ошибки

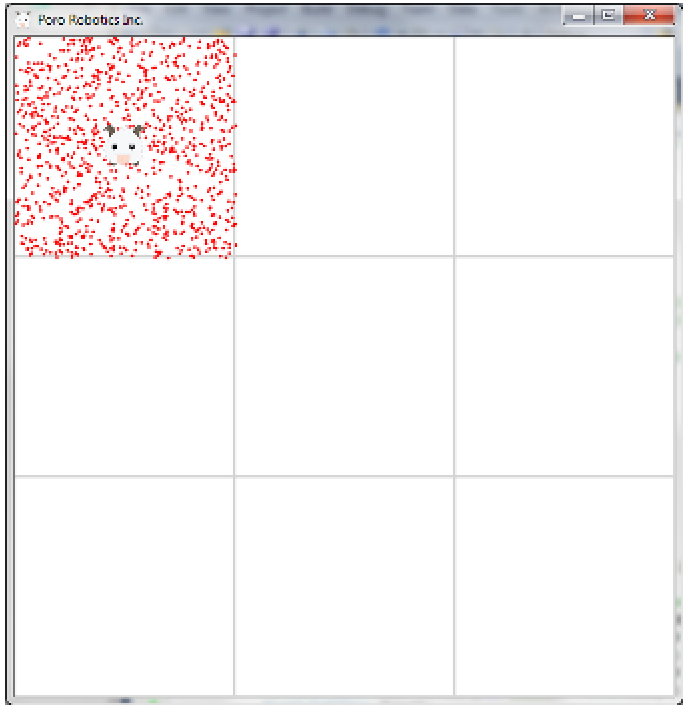
- Конечное количество частиц
- Случайность отсева
- Различие между предполагаемым и целевым распределением

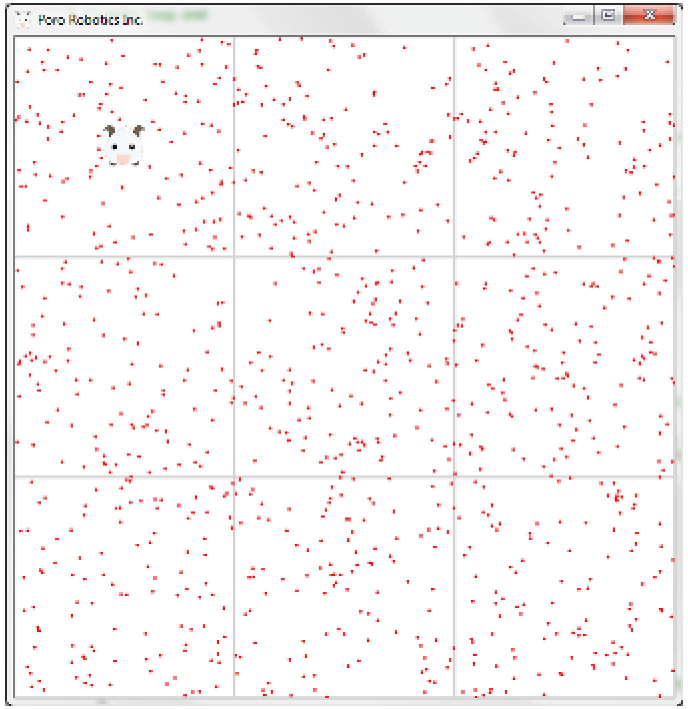
Фильтр частиц. Ошибки

- Конечное количество частиц
- Случайность отсева
- Различие между предполагаемым и целевым распределением
- Нехватка «хороших» частиц

Фильтр частиц. Пример

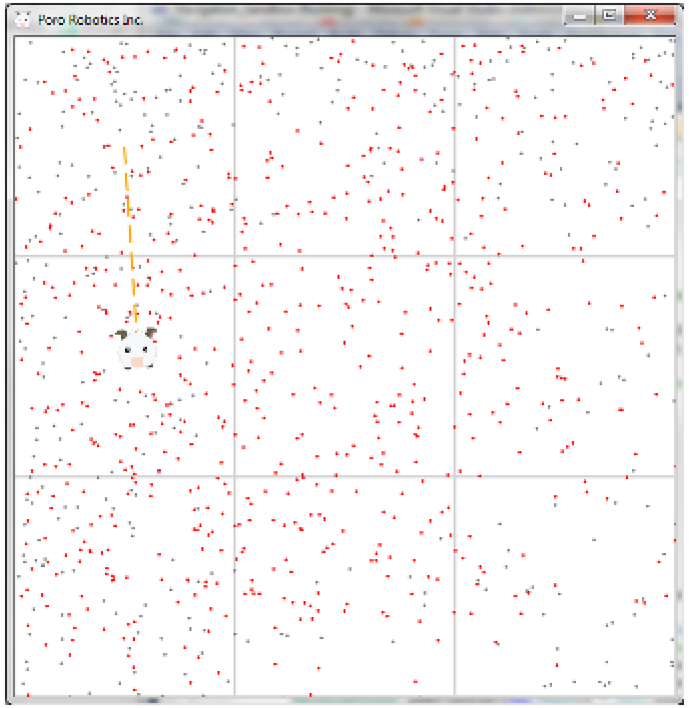
Инициализация





Фильтр частиц. Пример

Движение



Фильтр частиц. Пример

Измерение

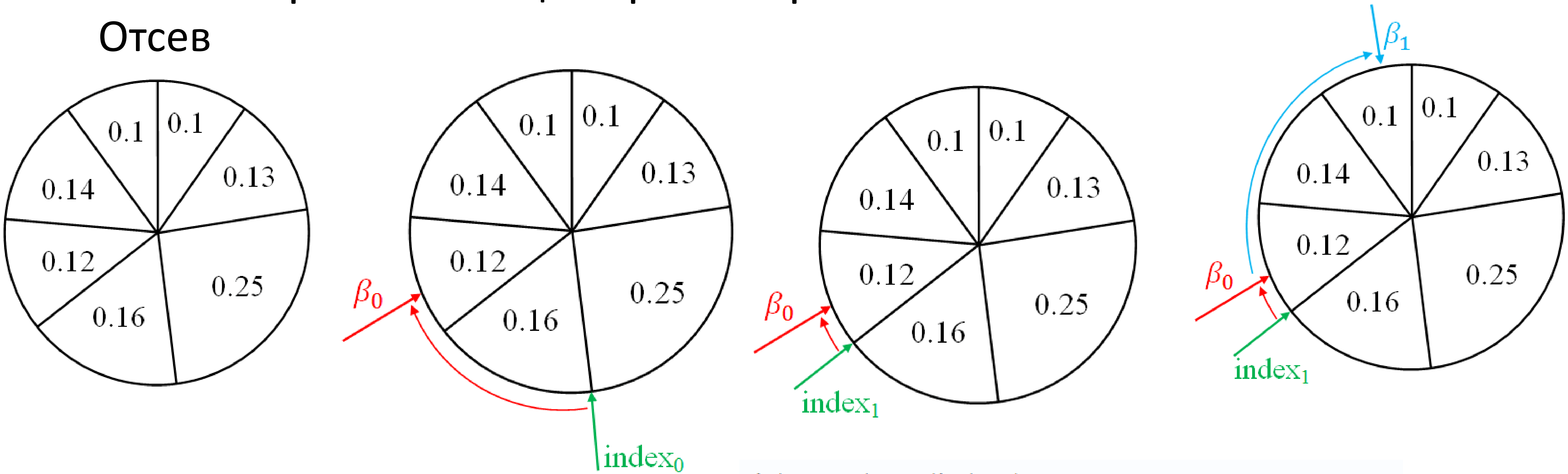
```
measurement = Sqrt((orientierX - robotX)^2 + (orientierY - robotY)^2)
```

$$f(\mu) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(\mu - \mu_0)^2}{\sigma^2}}$$

```
S = Sum(weight)
for i in range(N):
    weight[i] = weight[i]/S
```

Фильтр частиц. Пример

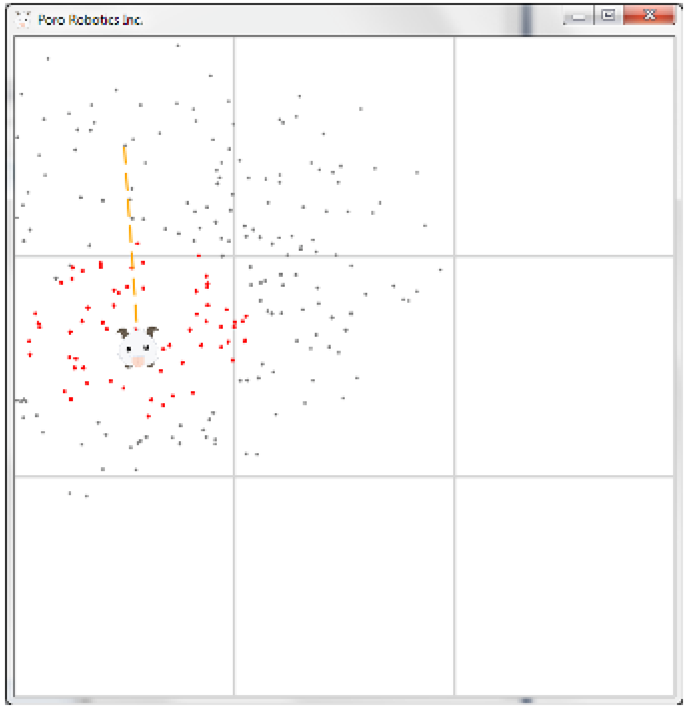
Отсев




```python
index = random.randint(0, 6)
betta = 0
for i in range(N):
    betta = betta + random.uniform(0, 2*max(weight))
    while betta > weight[index]:
        betta = betta - weight[index]
        index = (index + 1)%N # индекс изменяется в цикле от 0 до N
    newParticleList.append(particleList[index])
particleList = newParticleList
```

Фильтр частиц. Пример

Отсев



Фильтр частиц. Пример

Результат

```
estimateX = 0
estimateY = 0
for i in range(N):
    estimateX = estimateX + particleList[i].X*particleList[i].weight
    estimateY = estimateY + particleList[i].Y*particleList[i].weight
```