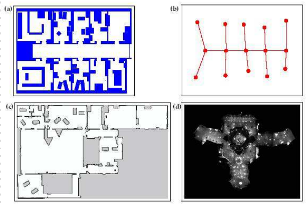
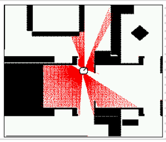
SCE5880
Algoritmos de Estimação para
Robótica Móvel

Localização I

Localização

Estimar a posição de um robô a partir de um mapa e de informações obtidas por sensores.





Localização: classificação

Tipo de problema

- Tracking
  - Posição inicial é conhecida
  - Busca local
  - Representação unimodal é apropriada

Localização: classificação

- Localização global
  - Posição inicial não é conhecida
  - Busca global
  - Representação multimodal é apropriada
- Kidnaped robot (teletransporte)
  - O robô pode ser movido para qualquer posição do ambiente
  - Exige solução robusta, capaz de se recuperar de falhas

Tipo de ambiente

- Estático
  - A única variável a ser estimada é a posição do robô
  - Formulação matemática concisa
- Dinâmico
  - Outras entidades, além do robô, se movem no ambiente
  - Problema complexo

Ambiente dinâmico - soluções

- Solução 1: estimar a posição dos objetos móveis
  - Aumento considerável da complexidade do problema
- Solução 2: identificar e filtrar os dados originados por entidades móveis
  - É necessário descartar informações, aumentando o tempo de convergência e diminuindo a precisão
  - Mantém a formulação de ambiente estático

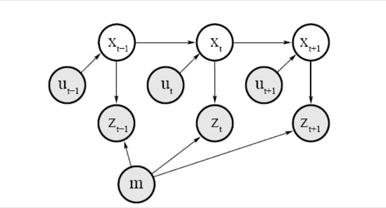
Tipo de localização

- Passiva
  - Módulo de localização apenas observa a operação do robô
- Ativa
  - Módulo de localização atua no controle do robô, de forma a facilitar a convergência na localização

Número de robôs

- Single robot
  - Todos os dados são obtidos pelo mesmo robô.
  - Não é necessário comunicação
- Multi-robot
  - Robôs podem trocar informações
  - Custo da comunicação
  - Complexidade na integração nos dados
  - Convergência mais rápida e robusta

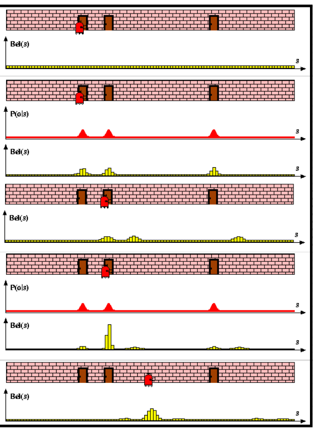
Localização de Markov



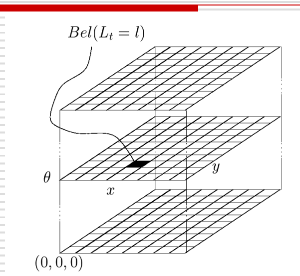
Localização de Markov

Algorithm Markov_localization($bel(x_{t-1}), u_t, z_t, m$):
  for all x_t do
    $\overline{bel}(x_t) = \int p(x_t \mid u_t, x_{t-1}, m)\ bel(x_{t-1})\ dx$
    $bel(x_t) = \eta\ p(z_t \mid x_t, m)\ \overline{bel}(x_t)$
  endfor
  return $bel(x_t)$

Malha de células

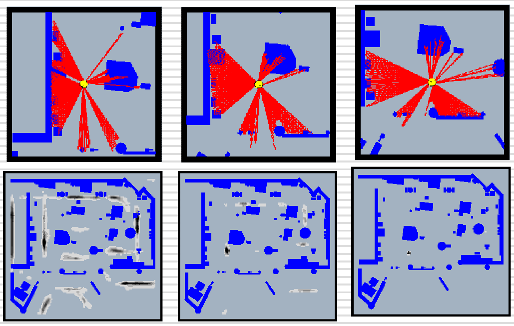


Células de ocupação

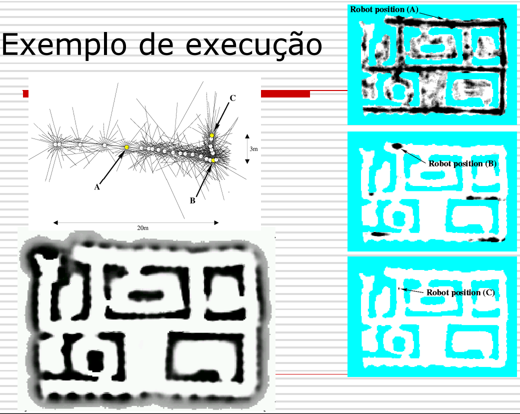


$$Bel(x_t = \langle x, y, \theta \rangle)$$

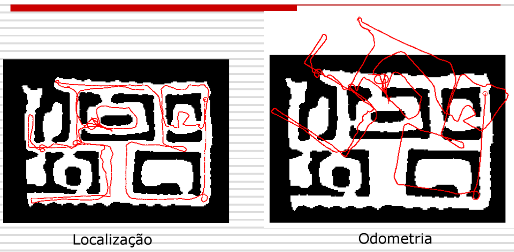
Exemplo de execução



Exemplo de execução

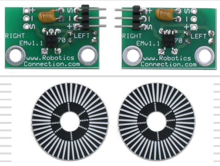


Localização x odometria

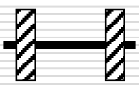


Modelo de movimentação probabilístico

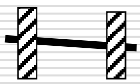
Cálculo de $p(x \mid x', u, m)$ baseado em informação odométrica.



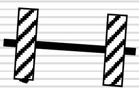
Possíveis causas para erros



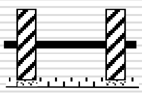
situação ideal



diâmetro diferente



desnível

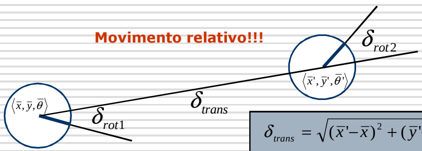


atrito

and many more …

Modelo de odômetro

- Robô se move de $\langle \bar{x}, \bar{y}, \bar{\theta} \rangle$ para $\langle \bar{x}', \bar{y}', \bar{\theta}' \rangle$.
- Informação odométrica $u = \langle \delta_{rot1}, \delta_{rot2}, \delta_{trans} \rangle$.



$$\delta_{trans} = \sqrt{(\bar{x}' - \bar{x})^2 + (\bar{y}' - \bar{y})^2}$$
$$\delta_{rot1} = \text{atan2}\,(\bar{y}' - \bar{y}, \bar{x}' - \bar{x}) - \bar{\theta}$$
$$\delta_{rot2} = \bar{\theta}' - \bar{\theta} - \delta_{rot1}$$

Calculando *x*, baseado em x′ e u

1. Algorithm **motion_model_odometry(x,x′,u)**
2. $\delta_{trans} = \sqrt{(\bar{x}'-\bar{x})^2 + (\bar{y}'-\bar{y})^2}$
3. $\delta_{rot1} = \text{atan2}(\bar{y}'-\bar{y}, \bar{x}'-\bar{x}) - \bar{\theta}$
4. $\delta_{rot2} = \bar{\theta}' - \bar{\theta} - \delta_{rot1}$
5. $\hat{\delta}_{trans} = \sqrt{(x'-x)^2 + (y'-y)^2}$
6. $\hat{\delta}_{rot1} = \text{atan2}(y'-y, x'-x) - \bar{\theta}$
7. $\hat{\delta}_{rot2} = \theta' - \theta - \hat{\delta}_{rot1}$
8. $p_1 = \text{prob}(\delta_{\text{rot1}} - \hat{\delta}_{\text{rot1}}, \alpha_1 |\hat{\delta}_{\text{rot1}}| + \alpha_2 \hat{\delta}_{\text{trans}})$
9. $p_2 = \text{prob}(\delta_{\text{trans}} - \hat{\delta}_{\text{trans}}, \alpha_3 \hat{\delta}_{\text{trans}} + \alpha_4 (|\hat{\delta}_{\text{rot1}}| + |\hat{\delta}_{\text{rot2}}|))$
10. $p_3 = \text{prob}(\delta_{\text{rot2}} - \hat{\delta}_{\text{rot2}}, \alpha_1 |\hat{\delta}_{\text{rot2}}| + \alpha_2 \hat{\delta}_{\text{trans}})$
11. return $p_1 \cdot p_2 \cdot p_3$

odometry values (u)

values of interest (x,x′)

Aproximação de distribuições probabilísticas

Algorithm prob_normal_distribution(a, b):

return $\frac{1}{\sqrt{2\pi b}} \, e^{-\frac{1}{2}\frac{a^2}{b}}$

Algorithm prob_triangular_distribution(a, b):

if $|a| > \sqrt{6b}$

return 0

else

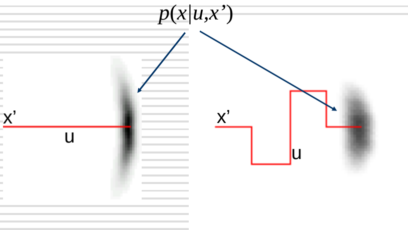
return $\frac{\sqrt{6b}-|a|}{6b}$

Probabilidade de *a*, dados média 0 e desvio padrão *b*

Aplicação

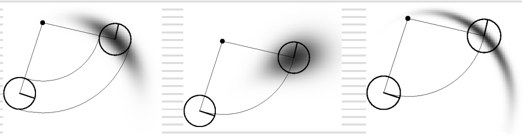
Aplicar o modelo repetidamente para movimentos curtos

$p(x|u,x')$



Modelo de odômetro

Exemplos:



Utilização do mapa

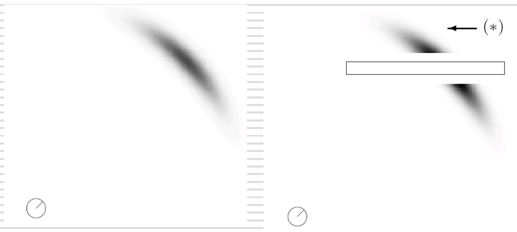
p(x| x′, u) ≠ p(x| x′, u, m)

p(x| x′, u, m) = η.p(x| x′, u).p(x|m)

p(x|m) = 1 se m estiver livre

p(x|m) = 0 se m estiver ocupado

Utilização do mapa



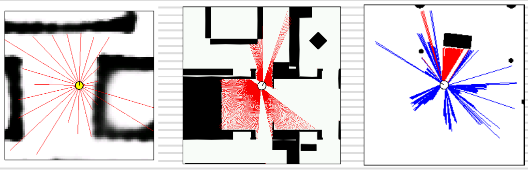
$p(x|u,x') \quad \neq \quad p(x|u,x',m)$

Modelo de percepção

Problema: calcular p(z|x)

Calcular p(z| x, m)

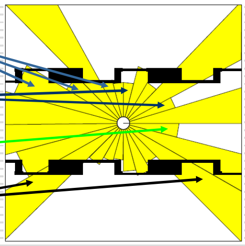
Sensores de distância



$$P(z \mid x, m) = \prod_{k=1}^{K} P(z_k \mid x, m)$$

Leituras de sensores de distância

1. Raios refletidos em obstáculos
2. Raios refletidos por pessoas/obst. móveis
3. Erros aleatórios
4. Distância máxima

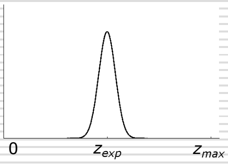


Modelo de raio-x

- Calcular a distância que deveria ser medida baseada no mapa e na posição do robô.
- Comparar com a distância lida do sensor

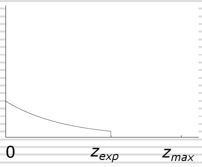
Modelo de raio-x

Measurement noise



$$P_{hit}(z \mid x, m) = \eta \frac{1}{\sqrt{2\pi b}} e^{-\frac{1}{2}\frac{(z - z_{exp})^2}{b}}$$

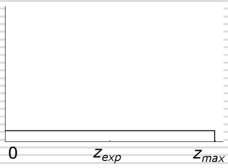
Unexpected obstacles



$$P_{unexp}(z \mid x, m) = \begin{cases} \eta\, \lambda\, e^{-\lambda z} & z < z_{exp} \\ 0 & otherwise \end{cases}$$

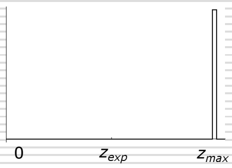
Modelo de raio-x

Random measurement



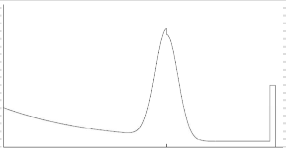
$$P_{rand}(z \mid x, m) = \eta \frac{1}{z_{max}}$$

Max range



$$P_{max}(z \mid x, m) = \eta \frac{1}{z_{small}}$$

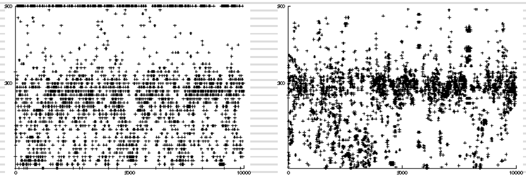
pdf resultante



$$P(z \mid x,m) = \begin{pmatrix} \alpha_{\text{hit}} \\ \alpha_{\text{unexp}} \\ \alpha_{\max} \\ \alpha_{\text{rand}} \end{pmatrix}^T \cdot \begin{pmatrix} P_{\text{hit}}(z \mid x,m) \\ P_{\text{unexp}}(z \mid x,m) \\ P_{\max}(z \mid x,m) \\ P_{\text{rand}}(z \mid x,m) \end{pmatrix}$$

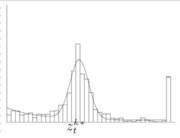
Dados obtidos de sensores

Dados obtidos em uma distância de 3m.

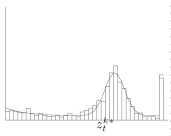


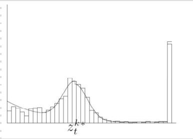
Sonar Laser

Aproximação

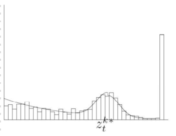


Laser



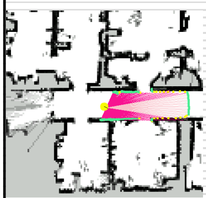


Sonar



300cm 400cm

Exemplo



z *P(z|x,m)*

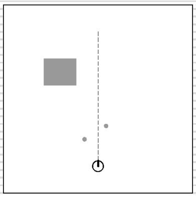
Modelo de raio-x

- Alto custo computacional
- Não tolera pequenos erros de localização

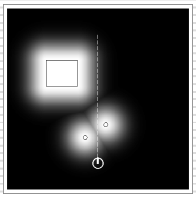
Modelo de mapa de probabilidade

- Pré-calcula a probabilidade de cada posição do mapa
- Compara o ponto obtido com os dados do sensor com o mapa de probabilidade e obtém p(z| x, m) diretamente

Exemplos

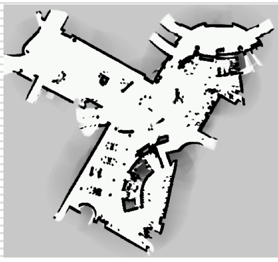


Mapa

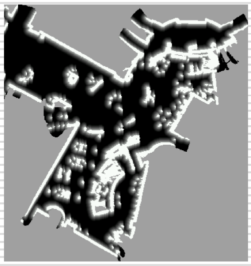


Mapa probabilidade

Exemplo



Occupancy grid map



Likelihood field

Mapa de likelihood

- Eficiente computacionalmente
- Comporta pequenos erros de localização
- Ignora a física do mapa e do sensor de distância

Landmarks

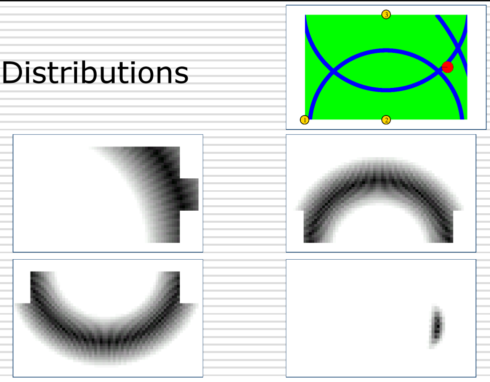
- Sensores ativos (radio, GPS, laser)
- Sensores passivos (cameras)
- Triangulação
  - Distância
  - Ângulo

Distance and Bearing

Modelo probabilístico

1. Algorithm **landmark_detection_model**(z,x,m):
   $z = \langle i, d, \alpha \rangle, x = \langle x, y, \theta \rangle$
2. $\hat{d} = \sqrt{(m_x(i) - x)^2 + (m_y(i) - y)^2}$
3. $\hat{a} = \operatorname{atan2}(m_y(i) - y, m_x(i) - x) - \theta$
4. $p_{\det} = \operatorname{prob}(\hat{d} - d, \varepsilon_d) \cdot \operatorname{prob}(\hat{\alpha} - \alpha, \varepsilon_\alpha)$
5. Return $p_{\det}$

Distributions



Próxima aula

Localização II

- Monte Carlo
- Filtro de Kalman