



# Robótica Móvel Inteligente: Da Simulação às Aplicações no Mundo Real

Denis Fernando Wolf  
Eduardo do Valle Simões  
Fernando Santos Osório  
Onofre Trindade Junior

Universidade de São Paulo – USP - ICMC  
Grupo de Sist. Embarcados, Evolutivos e Robóticos  
LRM – Laboratório de Robótica Móvel  
INCT – Sistemas Embarcados Críticos

**INCT**  
*SEC*

Instituto Nacional de Ciência e Tecnologia  
em **Sistemas Embarcados Críticos**



# LRM

*Laboratório de Robótica Móvel*

---

# Robótica Móvel Inteligente: Da Simulação às Aplicações no Mundo Real

**Denis Fernando Wolf**

**Eduardo do Valle Simões**

**Fernando Santos Osório**

**Onofre Trindade Junior**

**Universidade de São Paulo – USP - ICMC**

**Grupo de Sist. Embarcados, Evolutivos e Robóticos**

**LRM – Laboratório de Robótica Móvel**

**INCT – Sistemas Embarcados Críticos**

**INCT**  
*SEC*

**Instituto Nacional de Ciência e Tecnologia  
em Sistemas Embarcados Críticos**

# Robótica Móvel

---

- Simuladores de Robôs Móveis
  - Player/Stage/Gazebo
- Sistemas Robóticos Móveis Inteligentes
  - Localização
  - Mapeamento
  - SLAM
  - Navegação

# Robótica Móvel - Simuladores

---

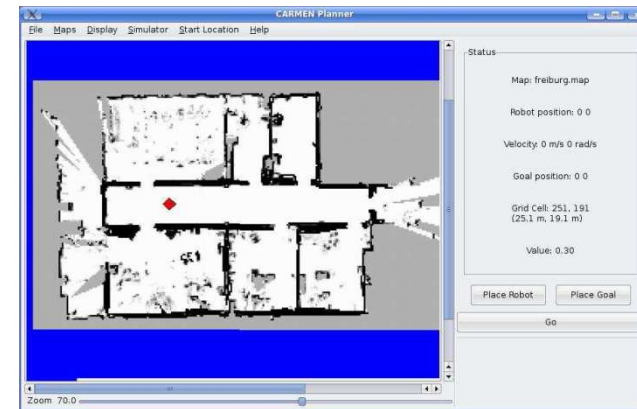
## Por que usar simuladores?

- Economia de recursos financeiros
- Economia de tempo
- Evitar danos aos robôs e sensores
- Evitar acidentes
- Aperfeiçoamento de hardware e software

# Simuladores de Robôs Móveis

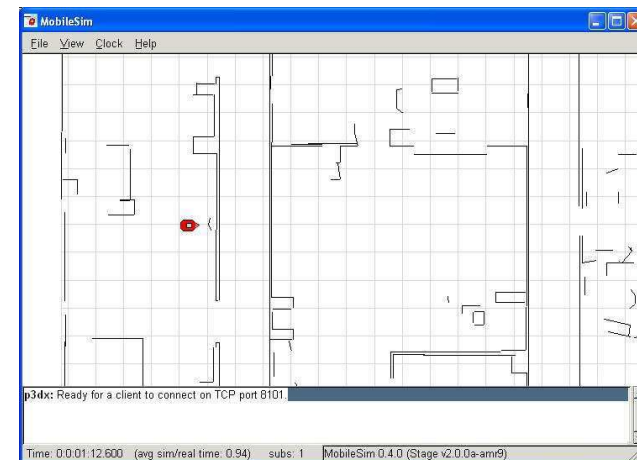
## Carmen

- Desenvolvido na CMU
- Simula e controla robôs móveis
- Gratuito e código aberto
- SO: Linux



## Aria/Saphira

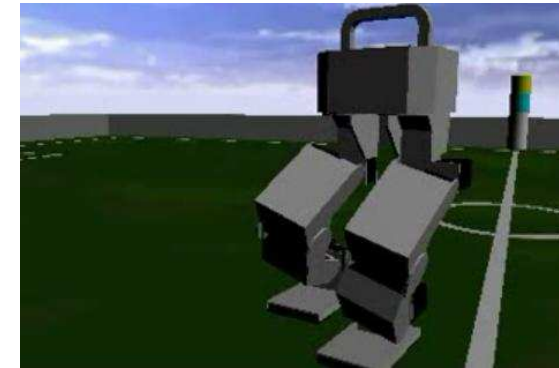
- Desenvolvido pela MobileRobots
- Simula e controla robôs Pioneer
- Exige licença
- SO: Linux e Windows



# Simuladores de Robôs Móveis

## Microsoft Robotics Studio

- Desenvolvido pela Microsoft
- Simula e controla robôs móveis
- Simulação física (3D)
- 3 tipos de licença
- SO: Windows



## Webots

- Desenvolvido pela Cyberbotics
- Simula e controla robôs móveis
- Simulação física (3D)
- Exige licença
- SO: Linux, Windows e Mac.



# Player/Stage/Gazebo

- **Player**

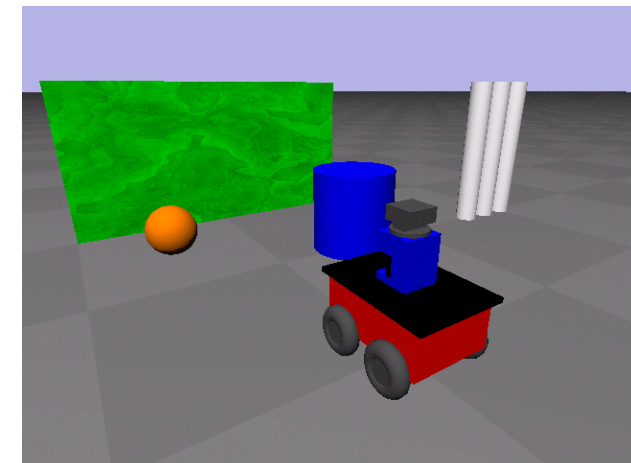
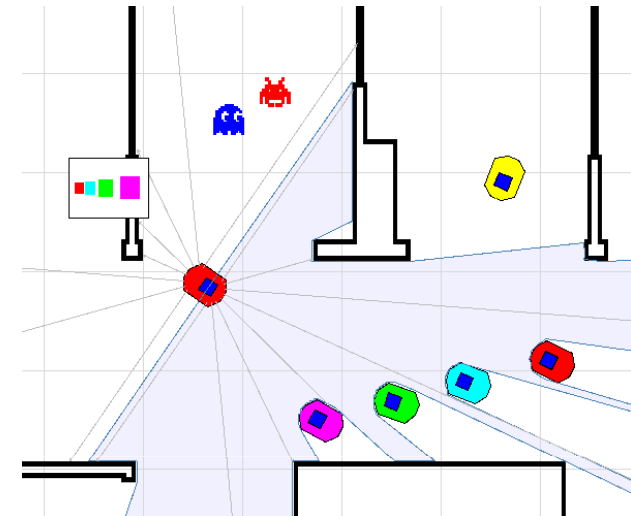
- Sistema para controle de robôs móveis
- Suporta diversos tipos de robôs e sensores

- **Stage**

- Simulador de robôs móveis e sensores
- Ambientes bidimensionais
- Compatível com Player

- **Gazebo**

- Simulador de alta fidelidade
- Ambientes em 3 dimensões
- Compatível com Player



# História do Player

---

- Projeto iniciado em 2000 na University of Southern California.
- Hoje em dia, seu core básico continua sendo desenvolvido pelos autores originais, apesar de diversos pesquisadores de diferentes instituições contribuem para o projeto.
- Atualmente, o Player é utilizado por diversas empresas e universidades em 5 continentes.

**+ 3.000 downloads por mês.**

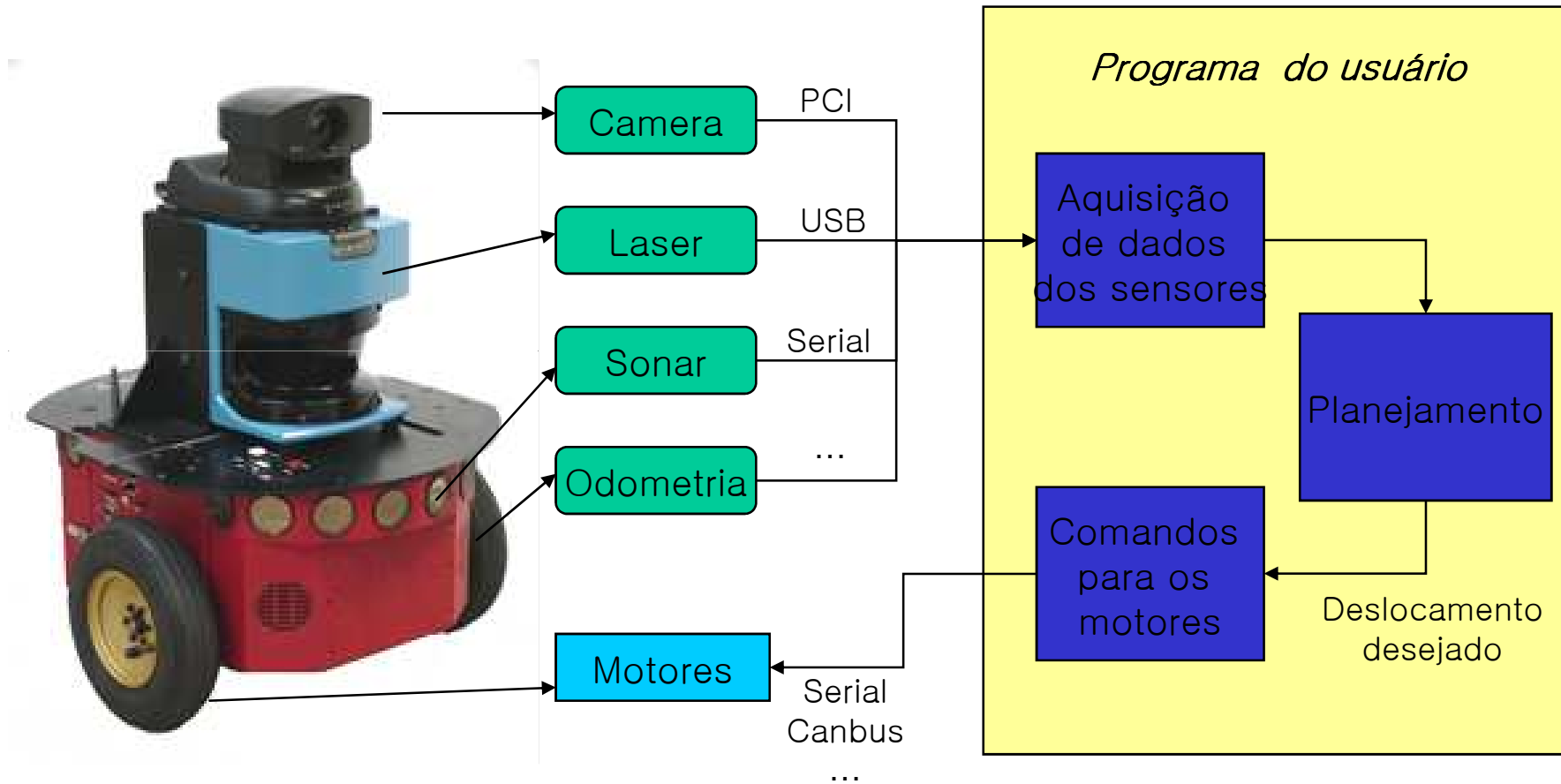


# Características do Player

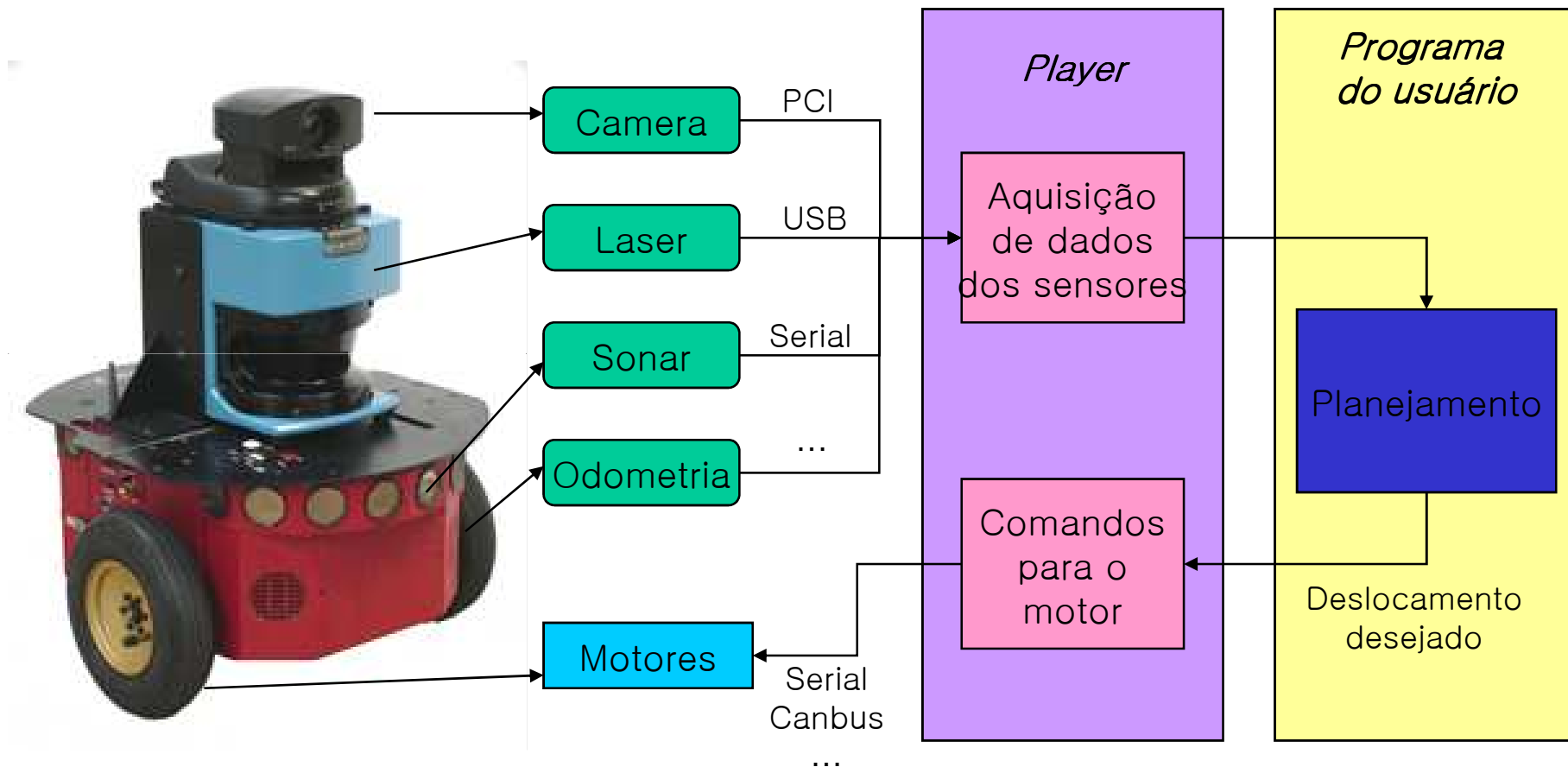
---

- Software livre
- Modelo Cliente/Servidor
- Desenvolvido para sistemas Linux/Unix
- Comunicação baseada em sockets
- Clientes em: C, C++, Java, Python etc
- Interface de alto nível para acesso ao hardware
- Suporta grande quantidade de plataformas robóticas e sensores comerciais

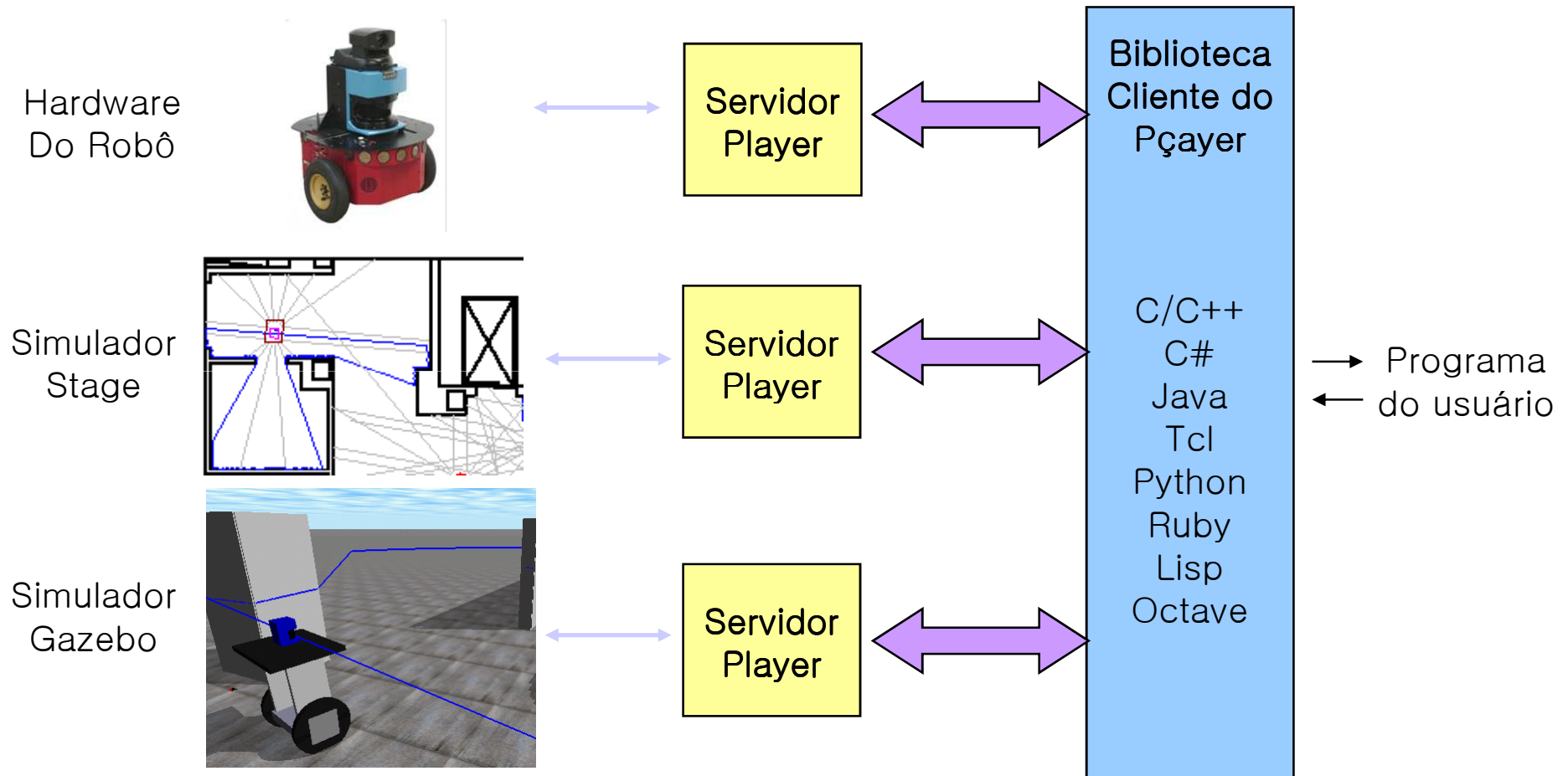
# Programa de controle



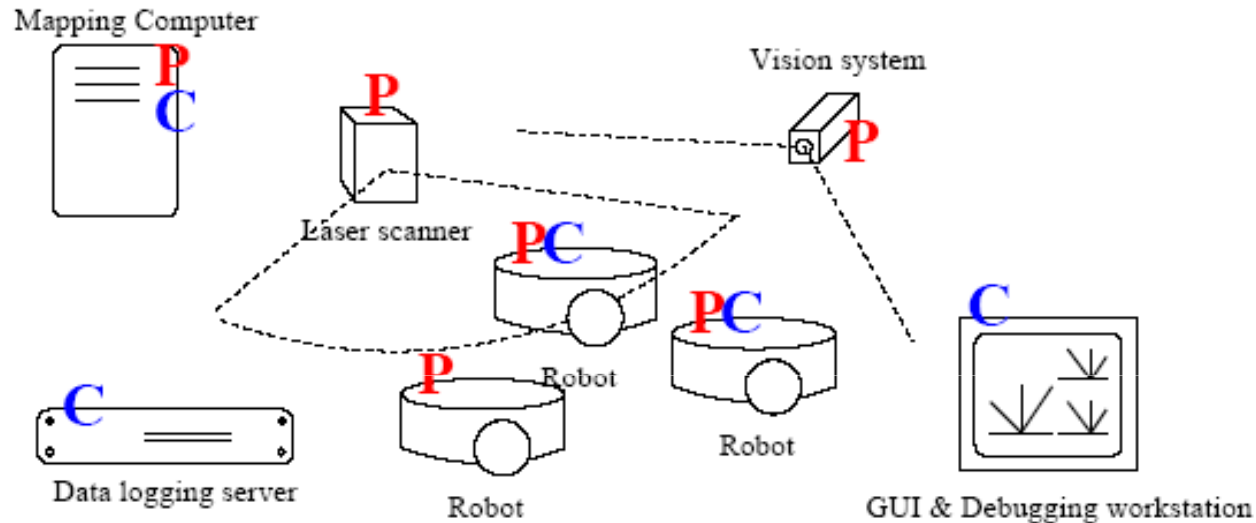
# Player



# Abstração de hardware



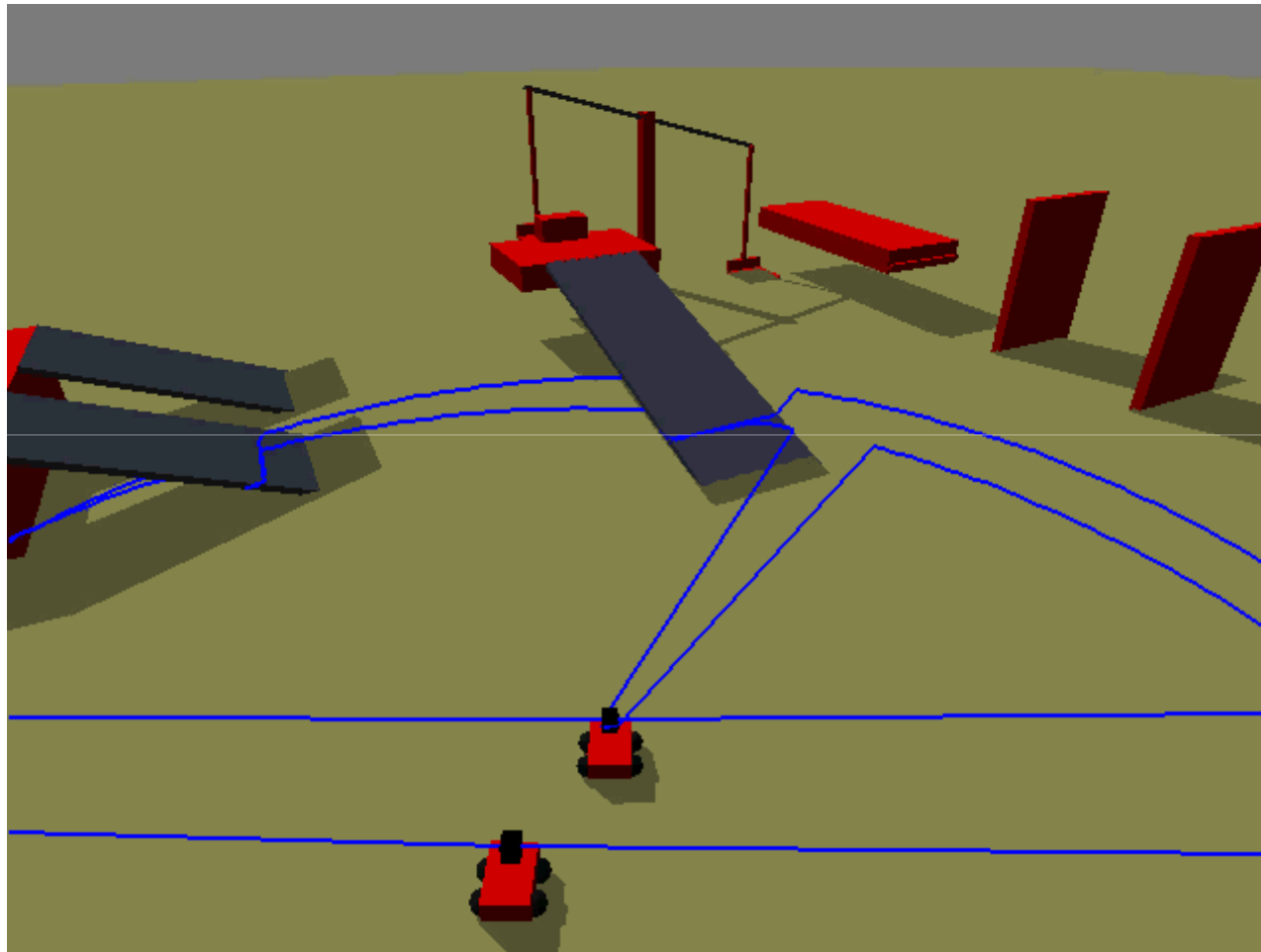
# Modelo Cliente/Servidor



- Clientes podem se conectar a múltiplos servidores
- Servidores aceitam conexão de múltiplos clientes
- Diferentes programas/processos/threads podem processar dados de diferentes sensores do mesmo servidor.
- Operação remota

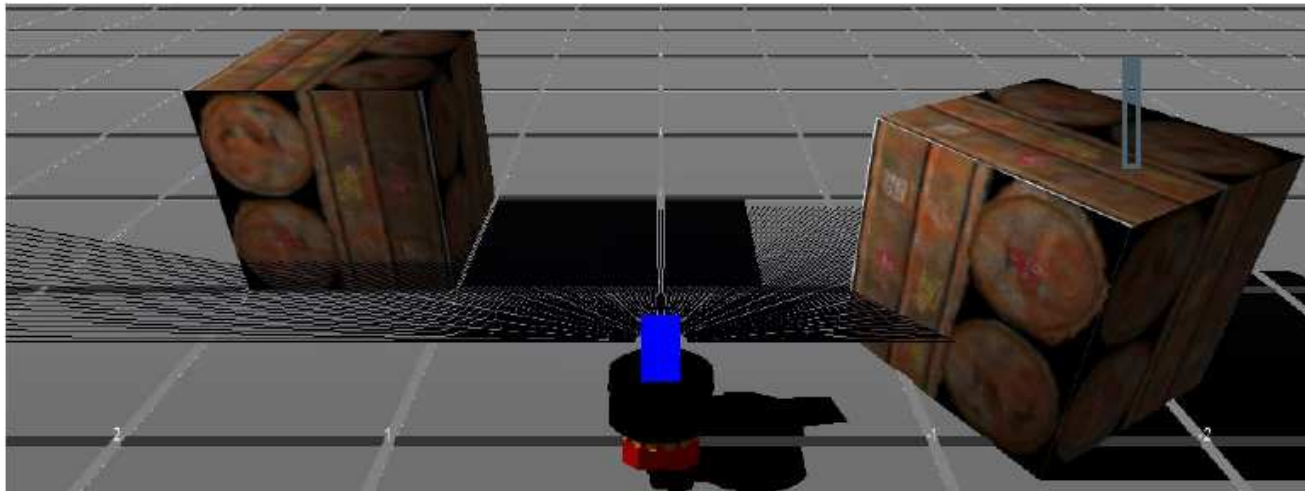
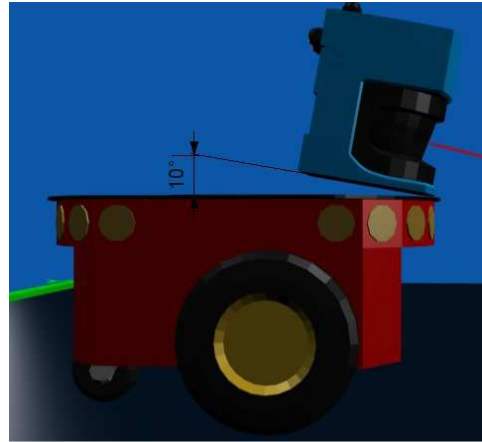
# Gazebo - Simulação 3D

---



# Gazebo v0.8

---



# Player/Stage

---

CSBC 2009 - JAI

# 16 Fundamentos



**LRM**  
*Laboratório de Robótica Móvel*

**INCT** *SEC*

Instituto Nacional de Ciência e Tecnologia  
em **Sistemas Embarcados Críticos**



# Robótica Móvel

---

- Simuladores de Robôs Móveis
  - Player/Stage/Gazebo
- **Sistemas Robóticos Móveis Inteligentes**
  - Localização
  - Mapeamento
  - SLAM
  - Navegação

# Robôs Móveis - Características

---

Um robô móvel é uma máquina capaz de extrair informação do ambiente e usar seu conhecimento sobre o mesmo para se locomover com um propósito definido.

Ronald Arkin

## Principais características:

- Mobilidade
- Capacidade de percepção
- Autonomia
- Inteligência

# Robôs Móveis - Aplicações



Guia de museu



Mapeamento de minas



Verificação da qualidade da água



Navegação autônoma

# Problemas

---

- Sensores são **limitados** e **imprecisos**.
- Atuadores são **limitados** e **imprecisos**.
- O ambiente e o estado interno do robô são **parcialmente observáveis**.
- Ambientes reais são **dinâmicos** e **imprevisíveis**.

# Modelo Básico



# Robótica Móvel

---

- **Simuladores de Robôs Móveis**
  - Player/Stage/Gazebo
- **Sistemas Robóticos Móveis Inteligentes**
  - **Localização**
  - Mapeamento
  - SLAM
  - Navegação

# Localização

Estimar a posição do robô em um **ambiente previamente conhecido**, utilizando informações obtidas por **sensores**.



# Robôs Móveis - Aplicações



Guia de museu



Mapeamento de minas



Verificação da qualidade da água



Navegação autônoma

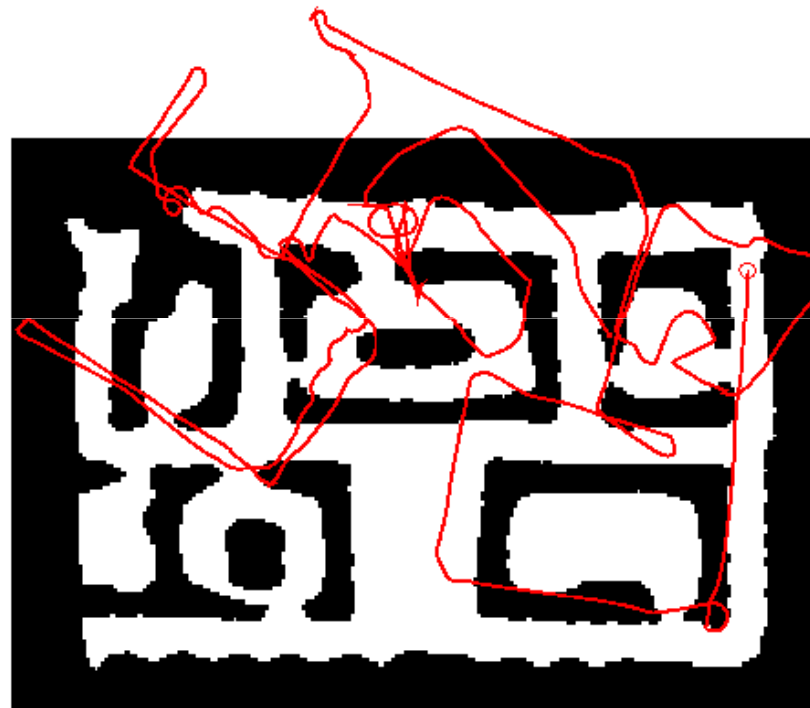


# Localização - Problemas

Efeito da **imprecisão** das informações dos sensores:



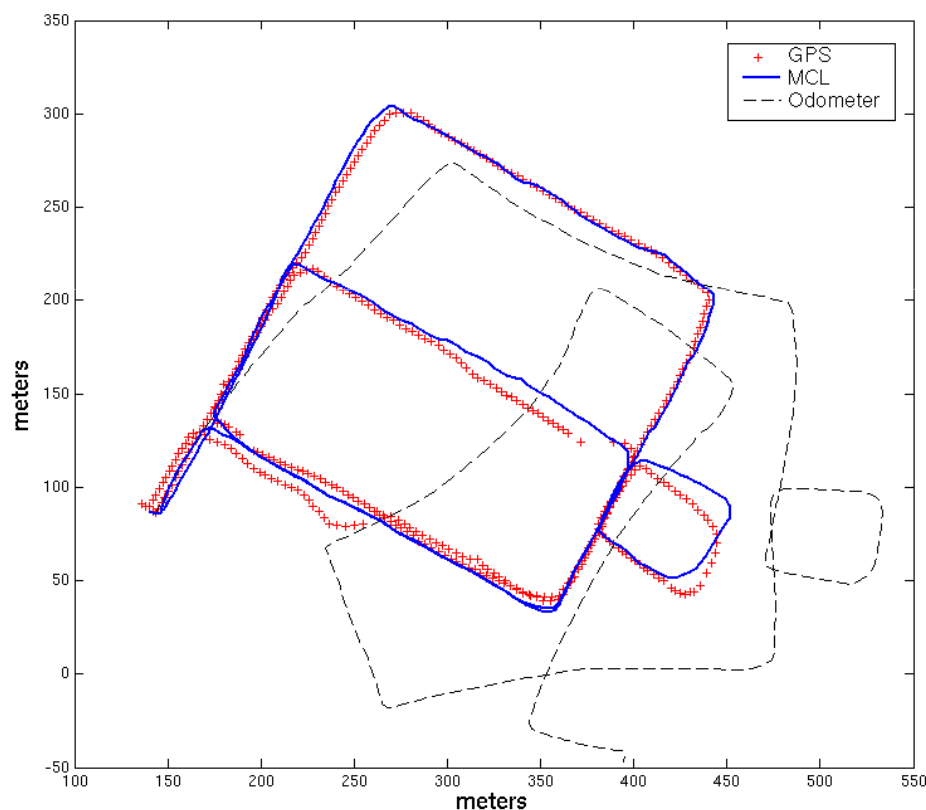
Trajeto real



Odometria

# Localização - Problemas

Efeito da **imprecisão** das informações dos sensores:



+++++ GPS

----- Odometria

# Localização – Tipos de problemas

---

## Tracking

- Posição inicial é conhecida
- Busca local (correção de odometria)

## Localização global

- Posição inicial não é conhecida
- Busca global

# Localização - Soluções

---

**Framework básico:** Filtro de Bayes

**Representação da posição do robô**

- Distribuição Normal (Filtro de Kalman)
- Malha de células (Grid / Markov)
- Partículas/Amostras (Monte Carlo)

# Filtro de Bayes

---

Nova  
estimativa

Observação  
**(diminuição  
de incerteza)**

Deslocamento  
do robô

Estimativa  
anterior

**(aumento de incerteza)**

$$Bel(x_t) = \eta p(o_t|x_t) \int p(x_t|x_{t-1}, a_{t-1}) Bel(x_{t-1}) dx_{t-1}$$

# Filtro de Kalman

---

- Uma das primeiras implementações práticas do filtro de Bayes (1960).
- Hipóteses para utilização do filtro:
  - Erro médio de cada variável igual a zero;
  - Erro independente para cada variável;
  - Modelo linear de evolução do sistema;
  - Relacionamento linear entre variáveis de estado e variáveis medidas.
- Se as hipóteses acima não forem cumpridas, a optimalidade **não** é assegurada.

# Filtro de Kalman

**Algoritmo filtro de Kalman**(  $\mu_{t-1}$ ,  $\Sigma_{t-1}$ ,  $u_t$ ,  $z_t$ ):

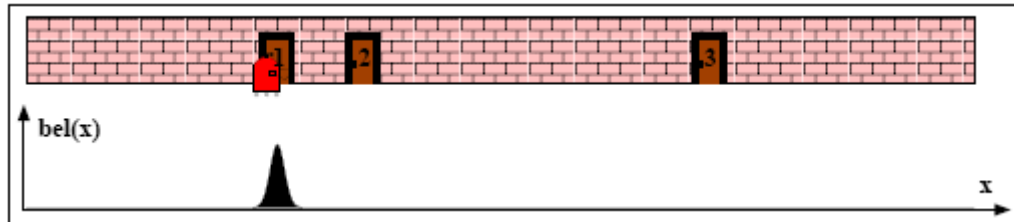
Predição:

1.  $\bar{\mu}_t = A_t \mu_{t-1} + B_t u_t$
2.  $\bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t$

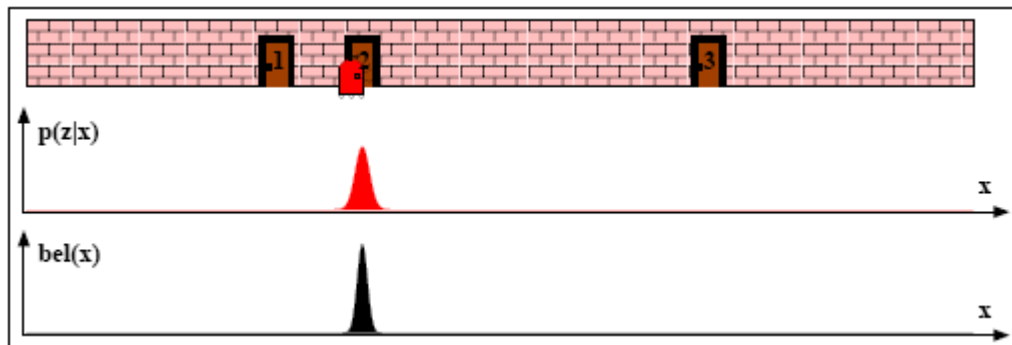
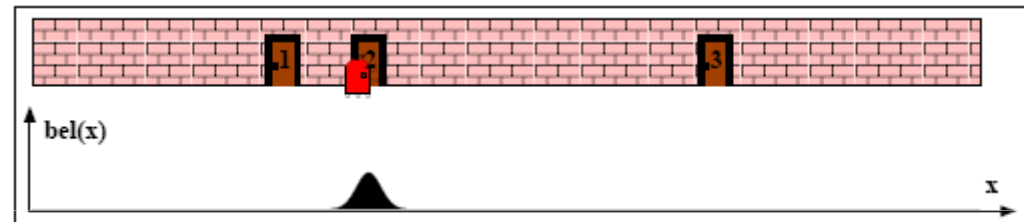
Correção:

1.  $K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1}$
2.  $\mu_t = \bar{\mu}_t + K_t (z_t - C_t \bar{\mu}_t)$
3.  $\Sigma_t = (I - K_t C_t) \bar{\Sigma}_t$
4. **Return**  $\mu_t$ ,  $\Sigma_t$

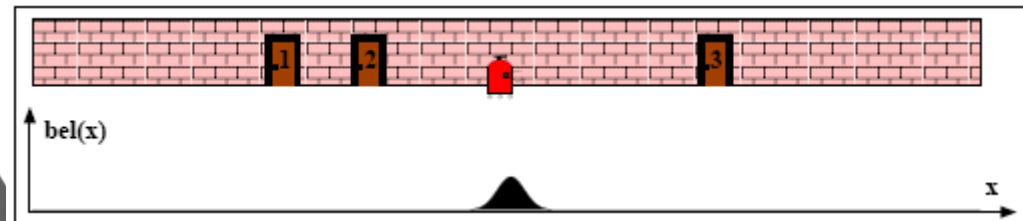
# Localização – Filtro de Kalman (EKF)



Atuação (movimento do robô):  
**aumento de incerteza**

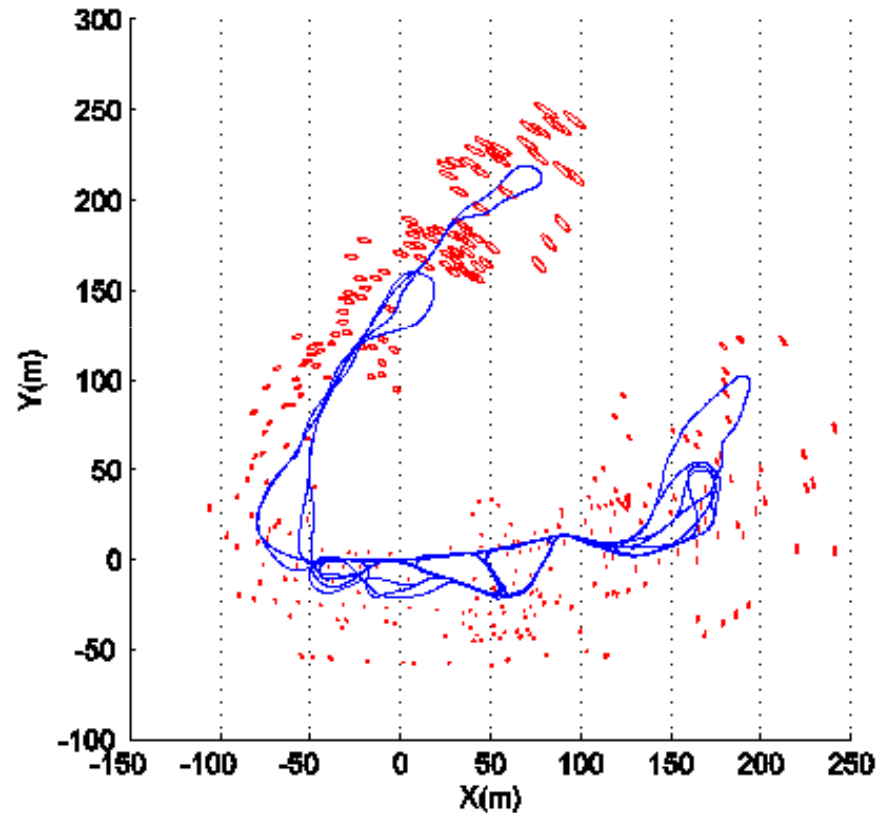


Percepção (sensores):  
**diminuição da incerteza**





# Localização - Filtro de Kalman



# Filtro de Kalman - Avaliação

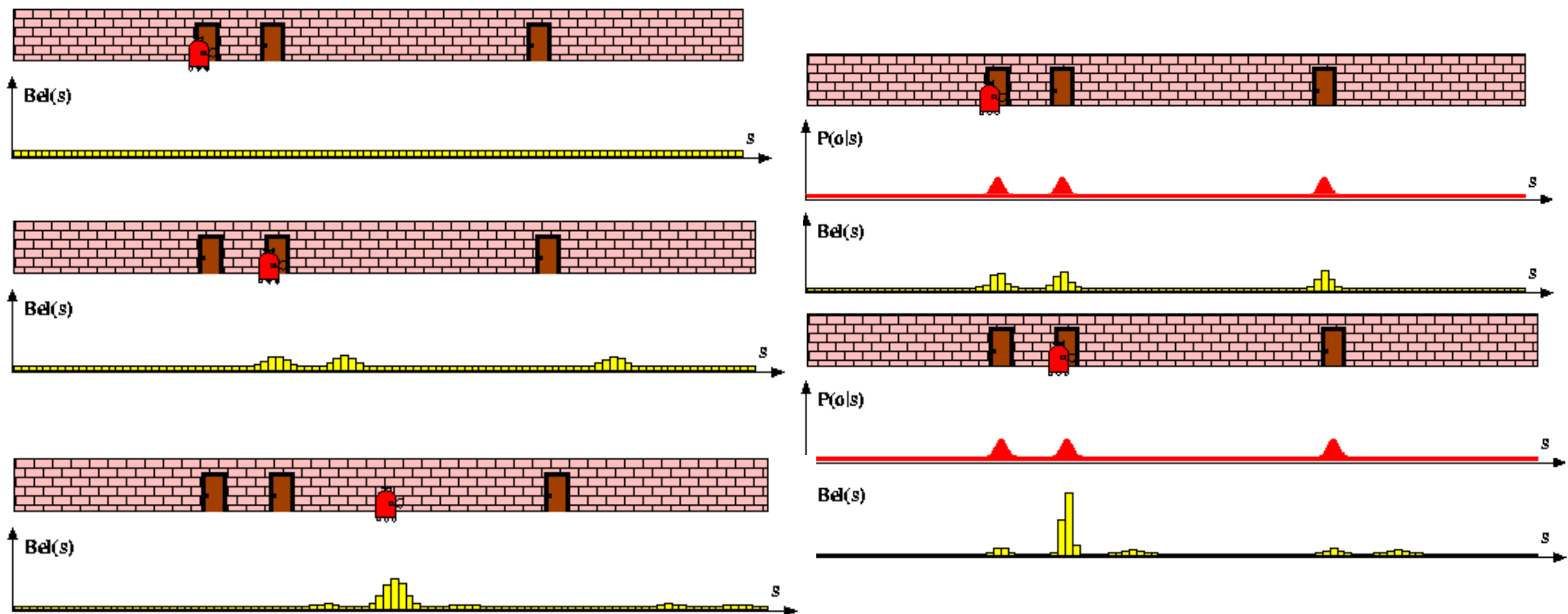
---

- **Não** é ótimo.
- **Pode divergir** de acordo a não-linearidade.
- **Altamente Eficiente**: complexidade polinomial com as dimensões de medição  $k$  e de estado  $n$ :  $O(k^{2.376} + n^2)$
- Funciona **surpreendentemente bem**, mesmo quando as suposições básicas são violadas!
- Adequado para tracking (busca local)

# Localização – Malha de células

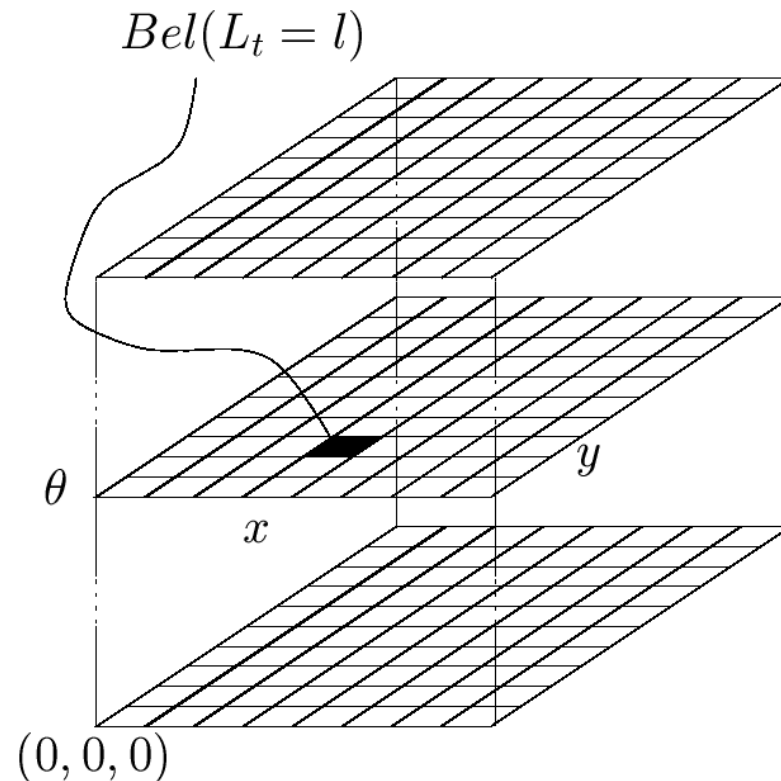
Atuação (movimento do robô): **aumento de incerteza**

Percepção (sensores): **diminuição da incerteza**

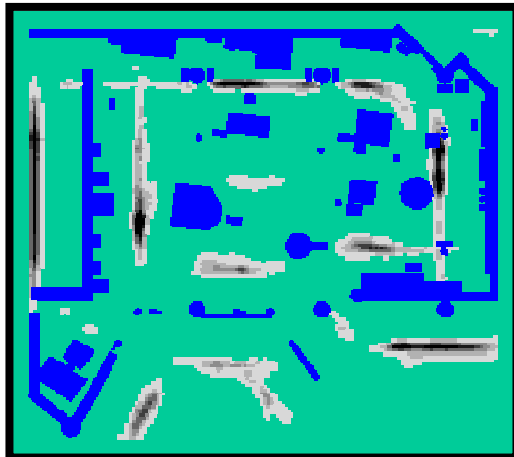
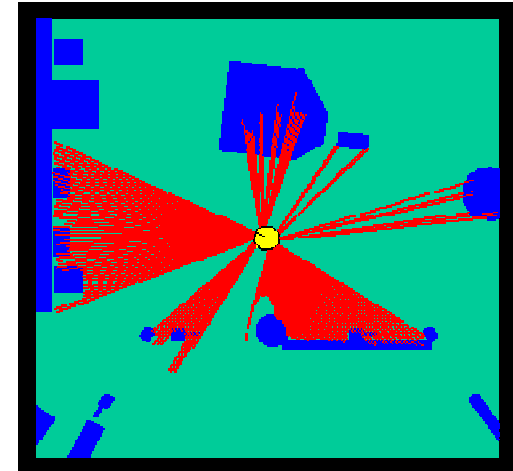
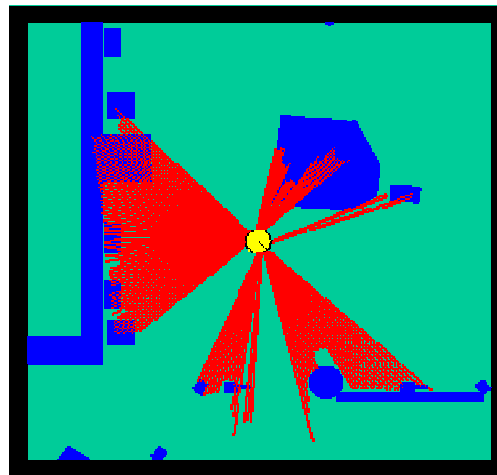
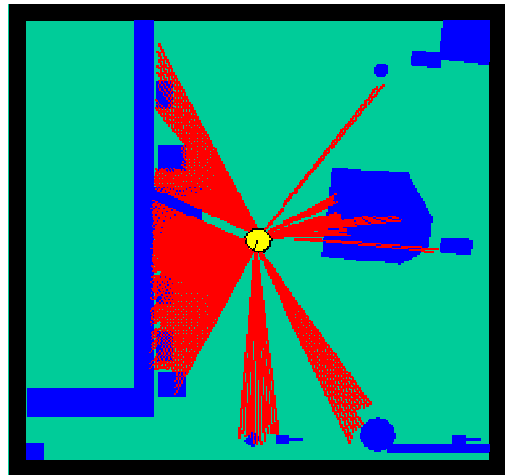


# Localização – Malha de células

Posição do robô:  $(x, y, \theta)$



# Localização - Exemplo



# Malha de Células - Avaliação

---

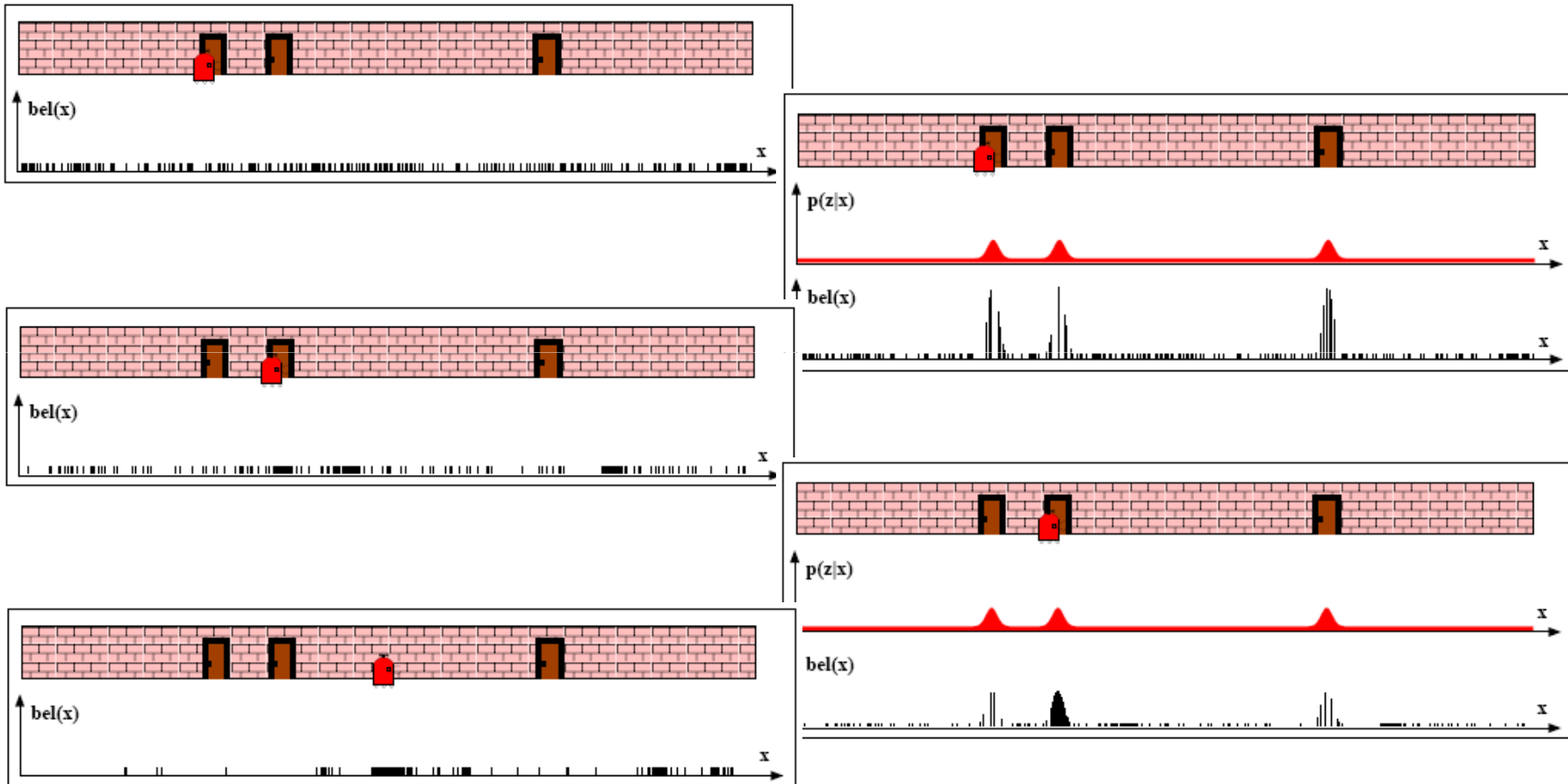
- Possibilita localização **global**.
- Precisão depende das dimensões das células
- **Altíssima** demanda computacional

# Localização – Partículas (Monte Carlo)

---

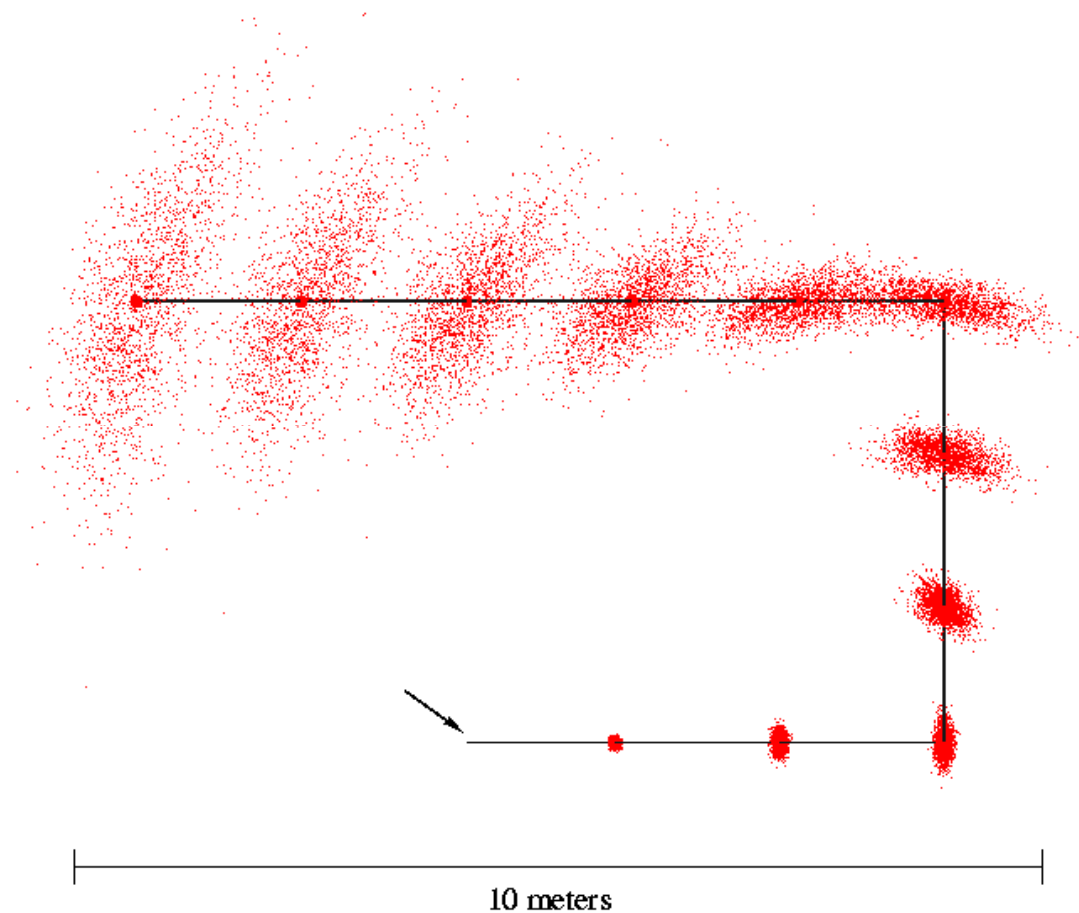
- Cada partícula representa a possibilidade do robô estar naquela posição específica.
- É atribuído um peso a cada partícula proporcional a sua chance de representar a posição do robô.
- Partículas com peso baixo são excluídas.
- Dado número suficiente de partículas, é provado que o método converge.

# Filtro de Partículas



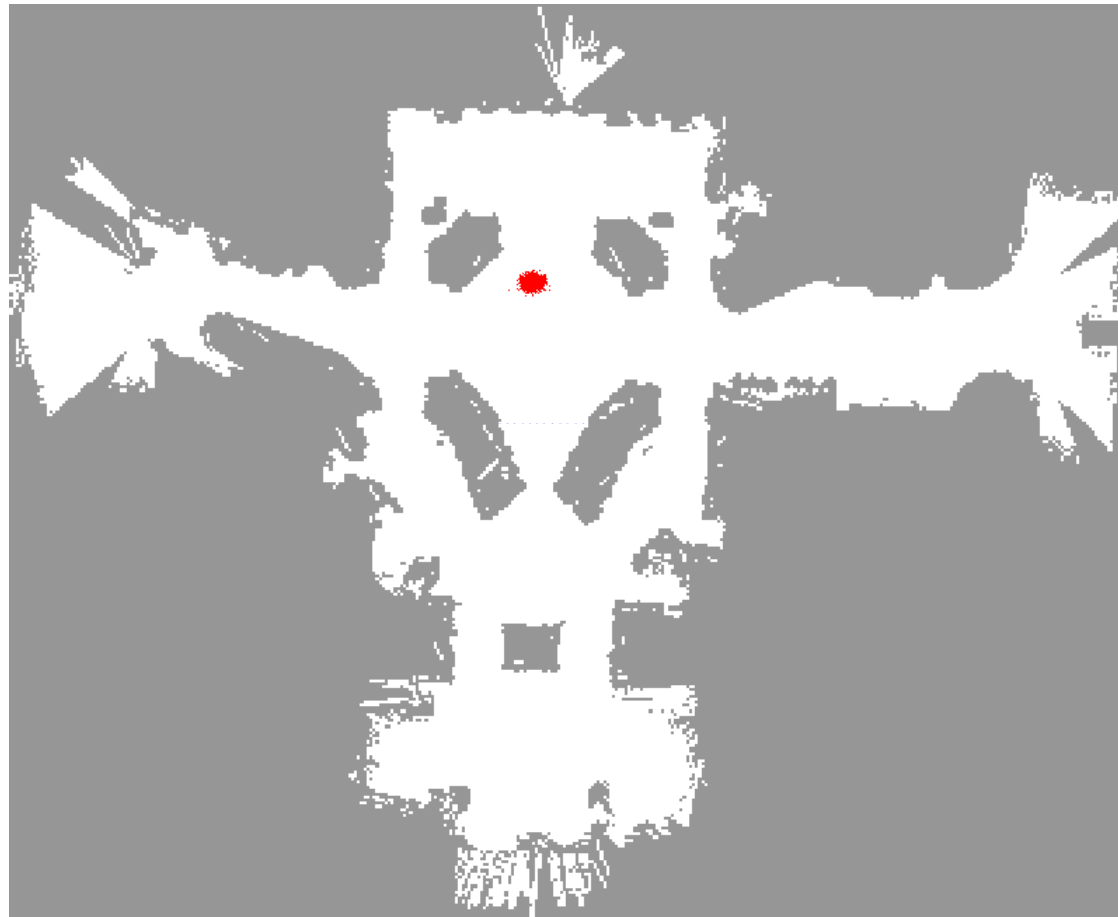


# Propagação das Partículas

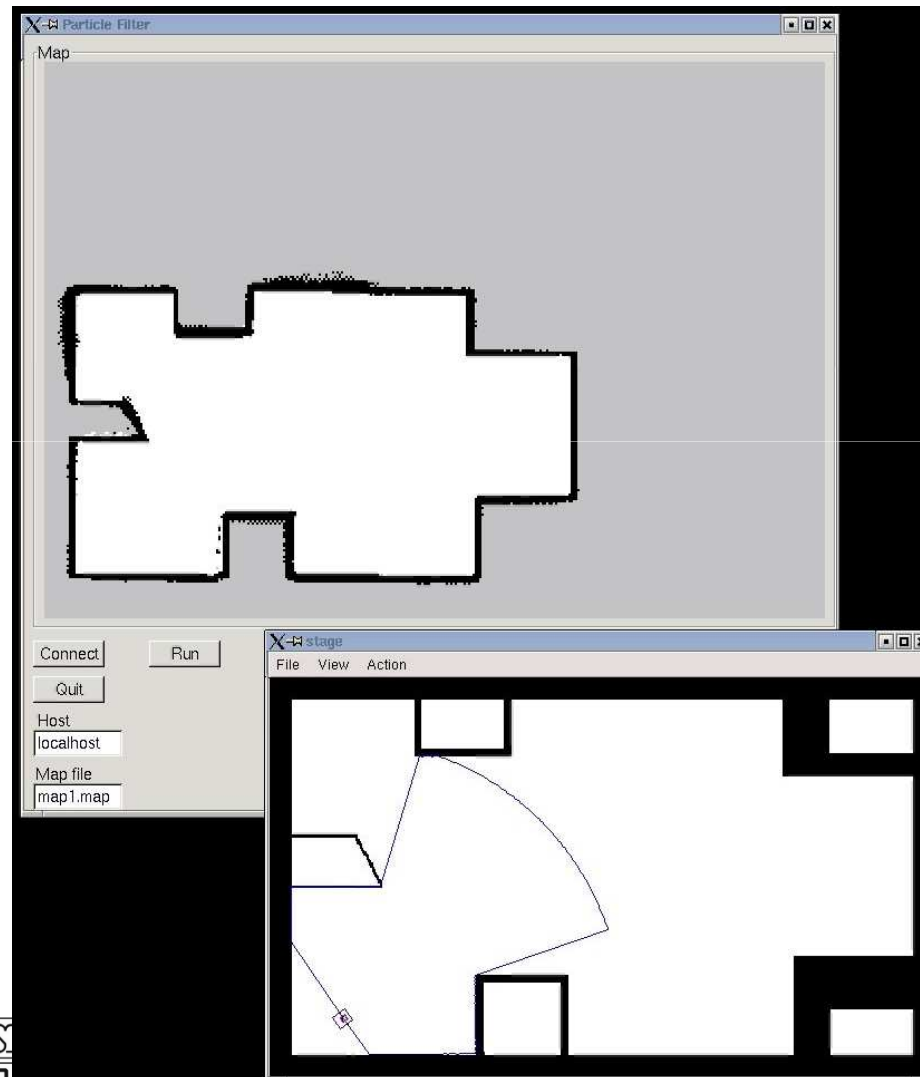


# Filtro de Partículas - Exemplo

---



# Filtro de Partículas - Simulação



# Filtro de Partículas

---

- Algoritmo **bastante eficiente** computacionalmente
- Possibilita **localização global**
- Implementação relativamente **simples**

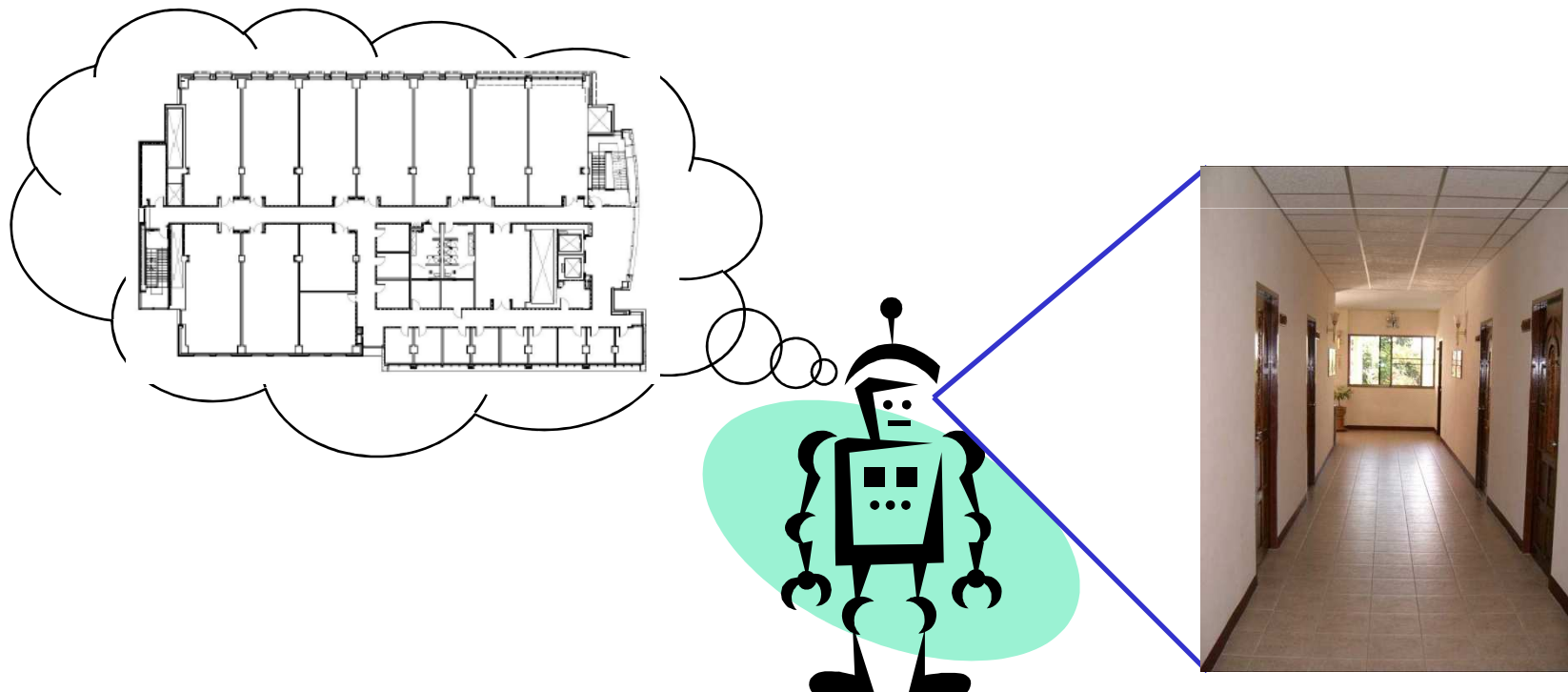
# Robótica Móvel

---

- Simuladores de Robôs Móveis
  - Player/Stage/Gazebo
- **Sistemas Robóticos Móveis Inteligentes**
  - Localização
  - **Mapeamento**
  - SLAM
  - Navegação

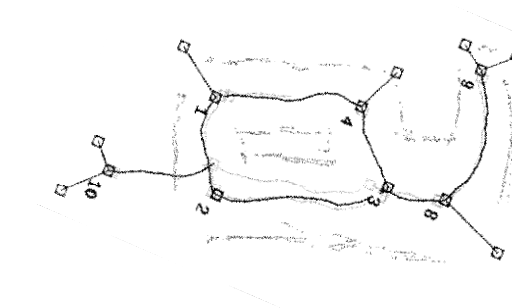
# Mapeamento

Criar um **modelo do ambiente** a partir da **localização** do robô e das **informações obtidas por sensores**.



# Tipos de Mapas

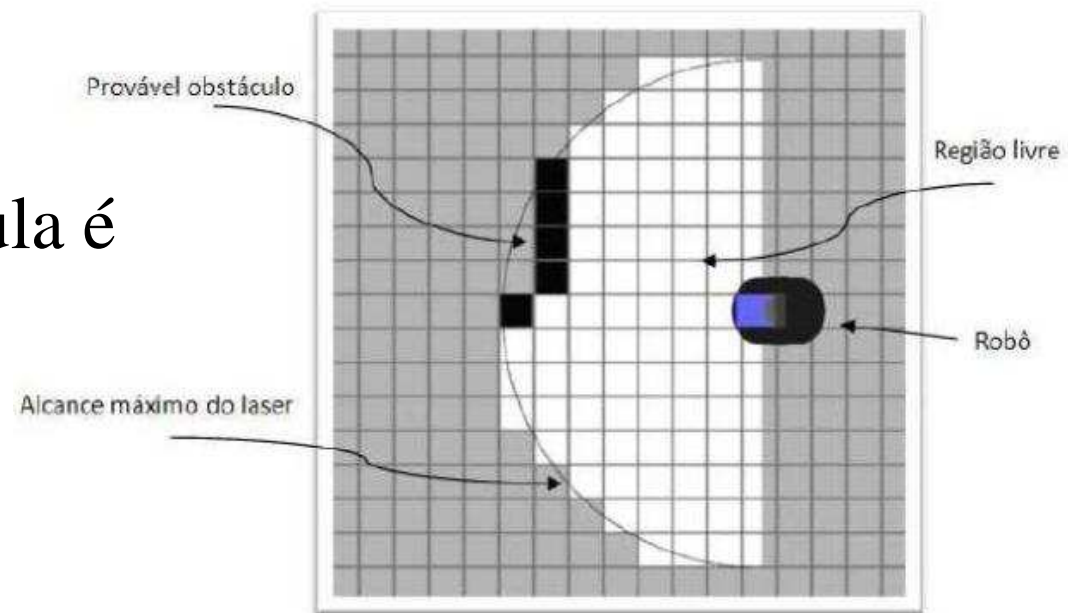
- **Mapas métricos:** representam propriedades geométricas do ambiente de forma quantitativa.
- **Mapas topológicos:** representam a conectividade entre determinados locais do ambiente. Normalmente são utilizados grafos nessa representação.



# Mapeamento Métrico – Grade de ocupação

- Dividir o espaço em células e estimar a probabilidade de ocupação de cada célula individualmente baseado na informação obtida pelos sensores.

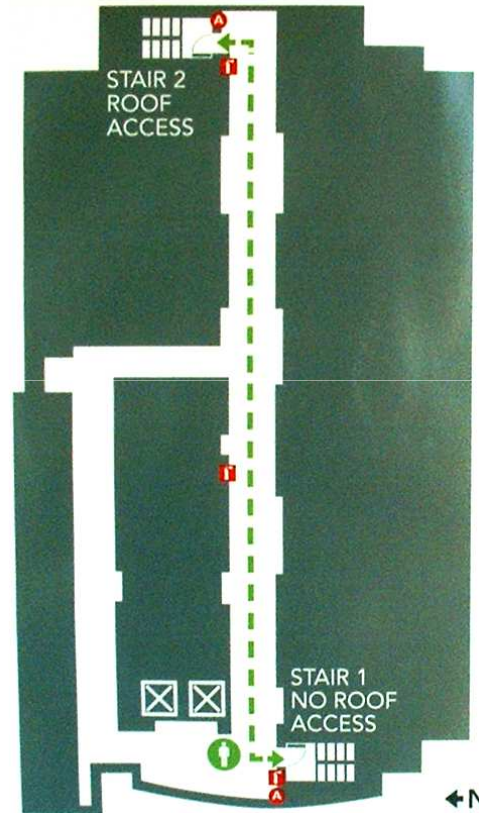
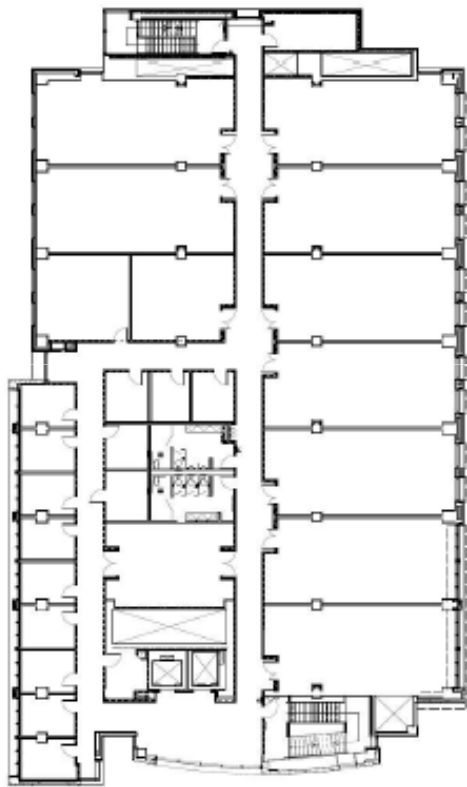
- Ao final, cada célula é classificada como ocupada, livre ou indefinido.





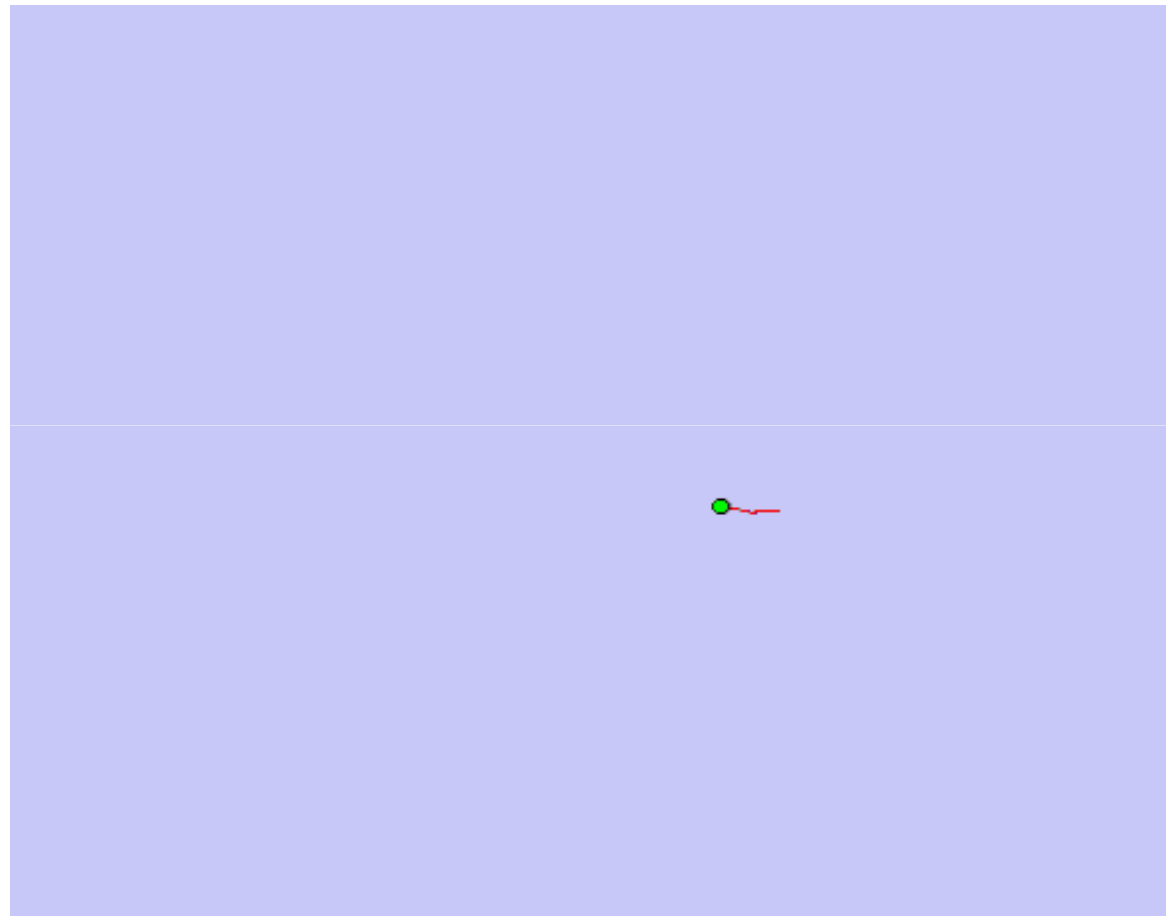


# Grade de Ocupação



# Mapa baseado somente na odometria

---



# Robótica Móvel

---

- Simuladores de Robôs Móveis
  - Player/Stage/Gazebo
- **Sistemas Robóticos Móveis Inteligentes**
  - Localização
  - Mapeamento
  - **SLAM**
  - Navegação

# Localização e Mapeamento Simultâneos

---

**O SLAM é um dos maiores desafios da robótica móvel.**

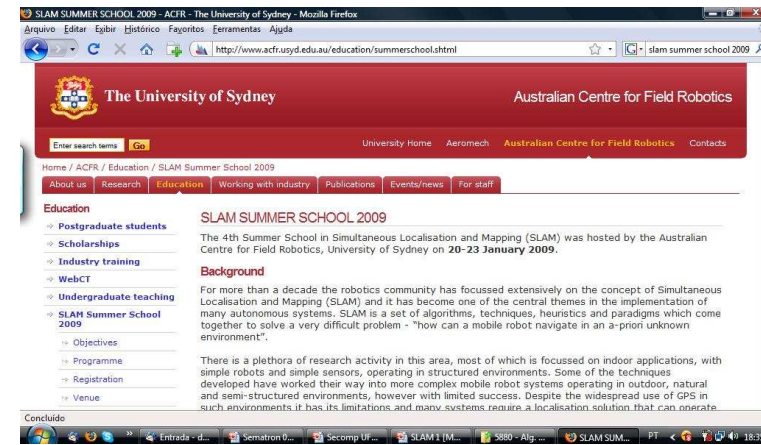
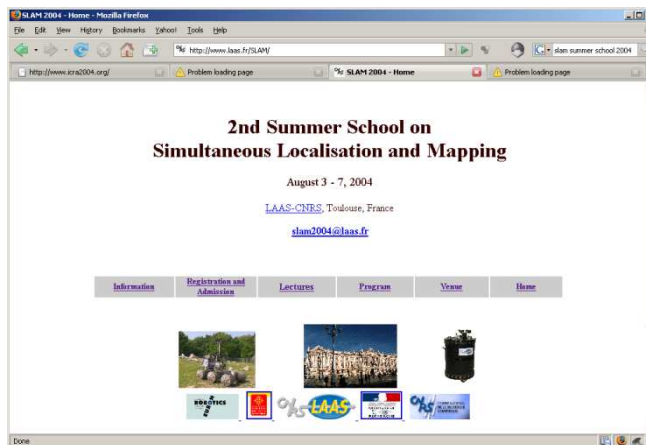
## **Dados:**

- Deslocamento do robô
- Informações obtidas pelos sensores

## **Deve-se estimar:**

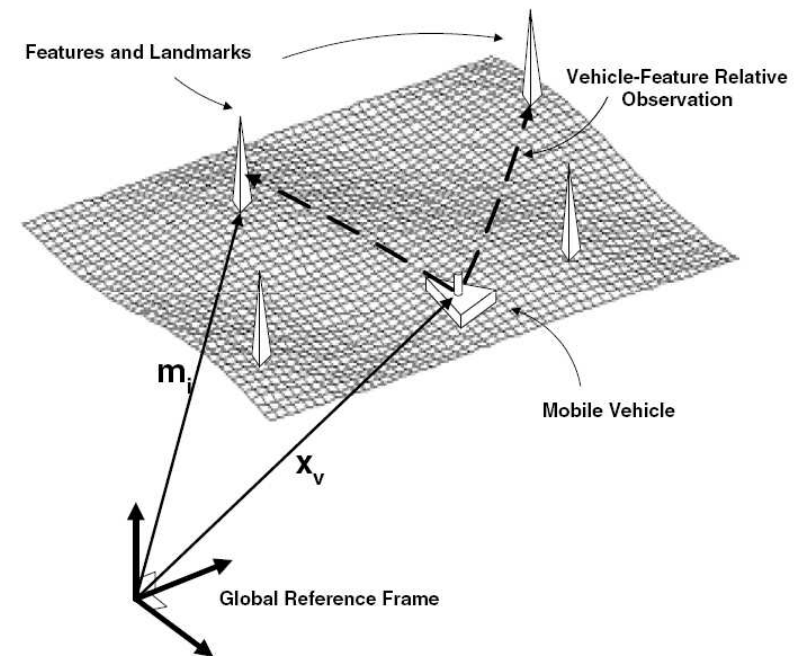
- O mapa do ambiente
- A localização/trajetória do robô

# Slam Summer School

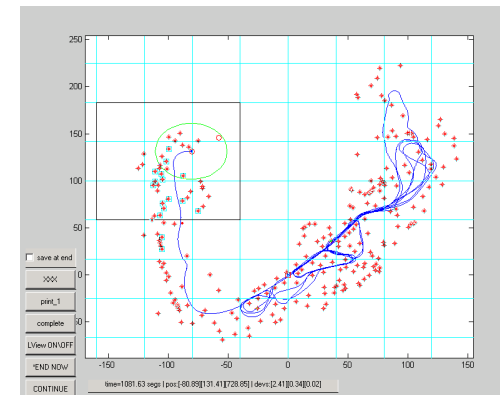


# SLAM – Filtro de Kalman

- Consiste em estimar a posição do robô e dos landmarks com o filtro de Kalman.
- As posição do robô e dos landmarks são correlacionadas através da matriz de covariância



# SLAM – Filtro de Kalman





# SLAM – EKF - Complexidade

$$\text{Be}(x_t, m_t) = \left( \begin{array}{c} x \\ y \\ \theta \\ l_1 \\ l_2 \\ \vdots \\ l_N \end{array} \right), \left( \begin{array}{ccc|ccc} \sigma_x^2 & \sigma_{xy} & \sigma_{x\theta} & \sigma_{xl_1} & \sigma_{xl_2} & \cdots & \sigma_{xl_N} \\ \sigma_{xy} & \sigma_y^2 & \sigma_{y\theta} & \sigma_{yl_1} & \sigma_{yl_2} & \cdots & \sigma_{yl_N} \\ \sigma_{x\theta} & \sigma_{y\theta} & \sigma_\theta^2 & \sigma_{\theta l_1} & \sigma_{\theta l_2} & \cdots & \sigma_{\theta l_N} \\ \hline \sigma_{xl_1} & \sigma_{yl_1} & \sigma_{\theta l_1} & \sigma_{l_1}^2 & \sigma_{l_1 l_2} & \cdots & \sigma_{l_1 l_N} \\ \sigma_{xl_2} & \sigma_{yl_2} & \sigma_{\theta l_2} & \sigma_{l_1 l_2} & \sigma_{l_2}^2 & \cdots & \sigma_{l_2 l_N} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \sigma_{xl_N} & \sigma_{yl_N} & \sigma_{\theta l_N} & \sigma_{l_1 l_N} & \sigma_{l_2 l_N} & \cdots & \sigma_{l_N}^2 \end{array} \right)$$

Um mapa com N landmarks estimará **2N+3 variáveis** e terá uma matriz de covariância de **2N+3 x 2N+3 elementos**.

# FastSLAM

---

**Idéia básica:** utilizar um **filtro de partículas** para estimar a **posição do robô** e o **mapa do ambiente**.

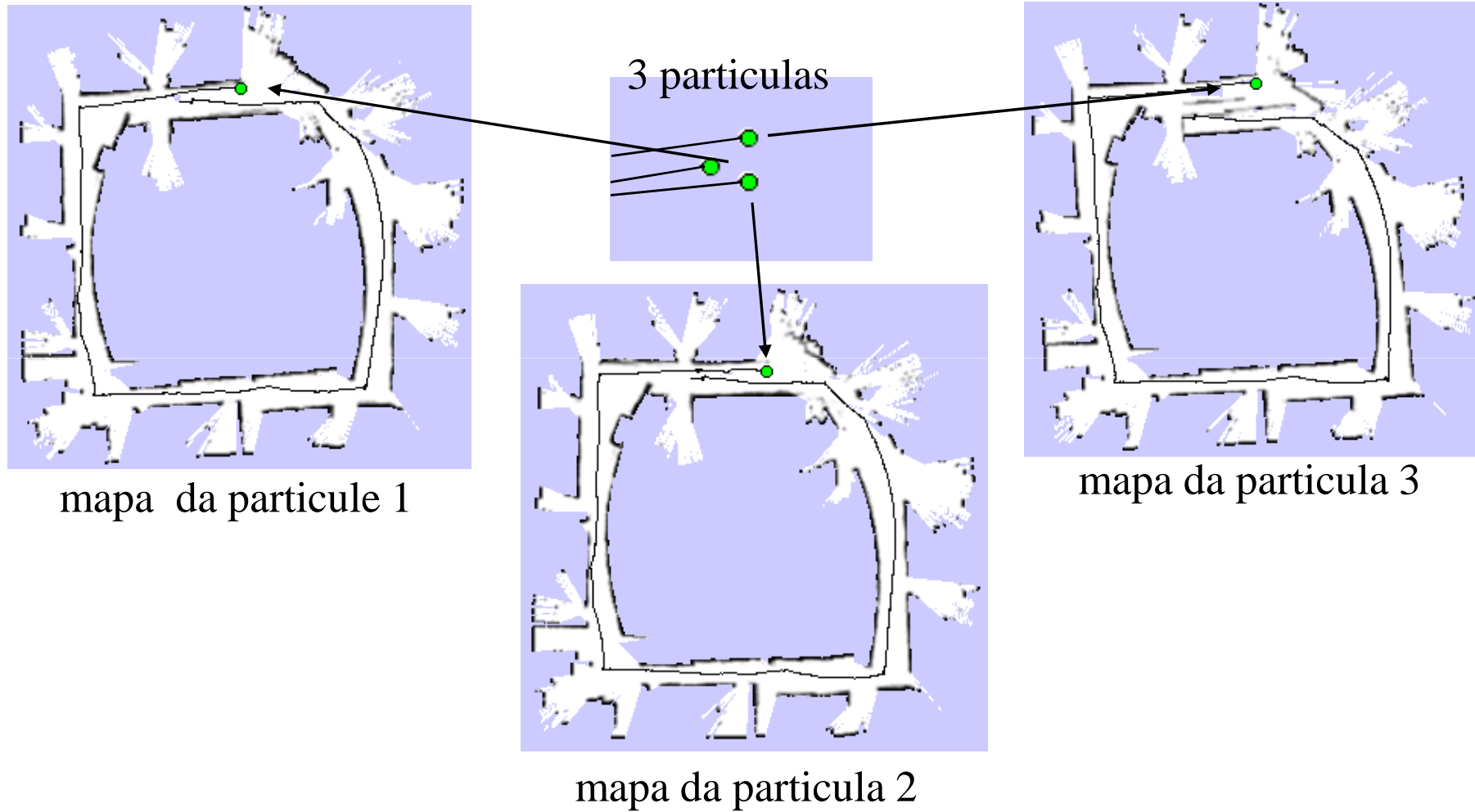
## Vantagens:

- Modelos de percepção e controle **não-lineares** 😊
- Permite **diferentes associações de dados** em paralelo 😊
- Estima **toda a trajetória** do robô **on-line** 😊

## Desvantagem:

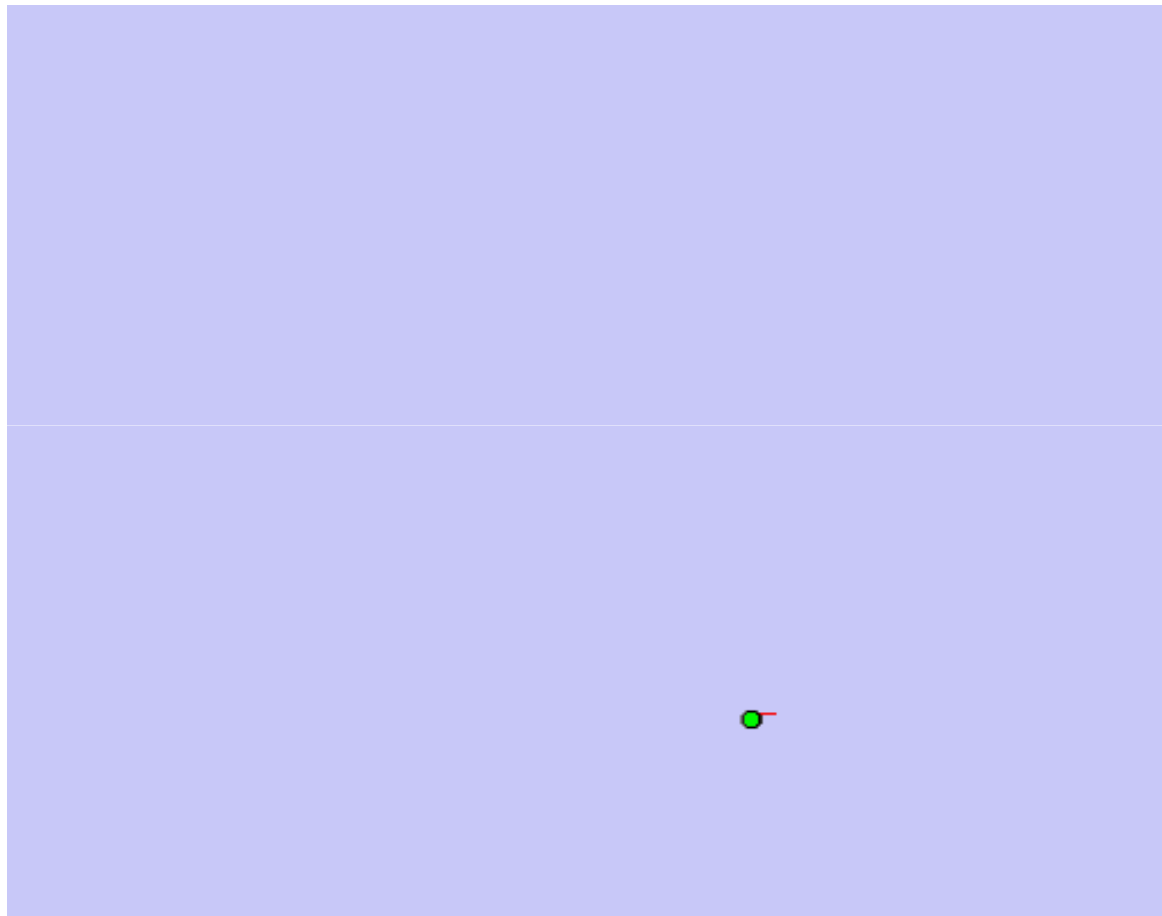
- Número de partículas necessário cresce **exponencialmente** com o número de variáveis. 😞 😞 😞

# FastSLAM



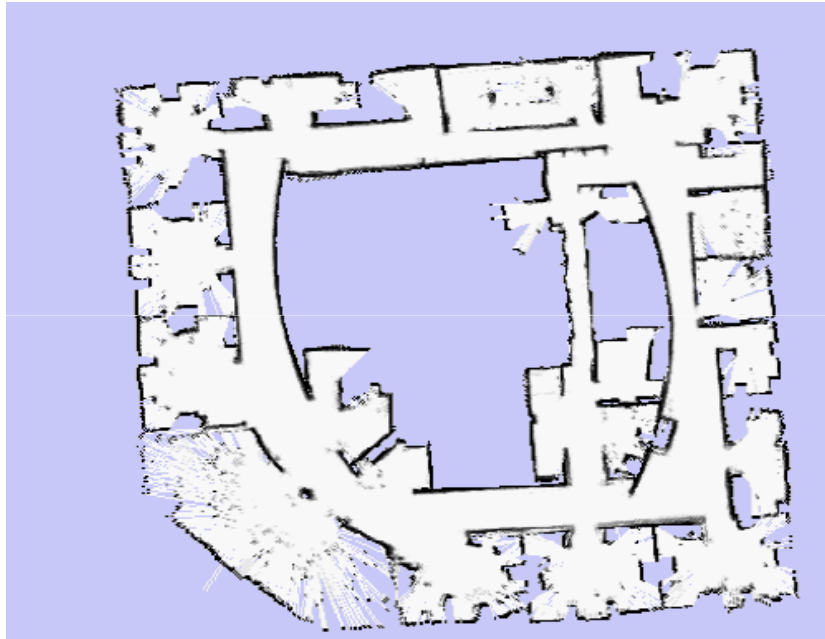
# FastSLAM

---



# FastSLAM - Resultados

---



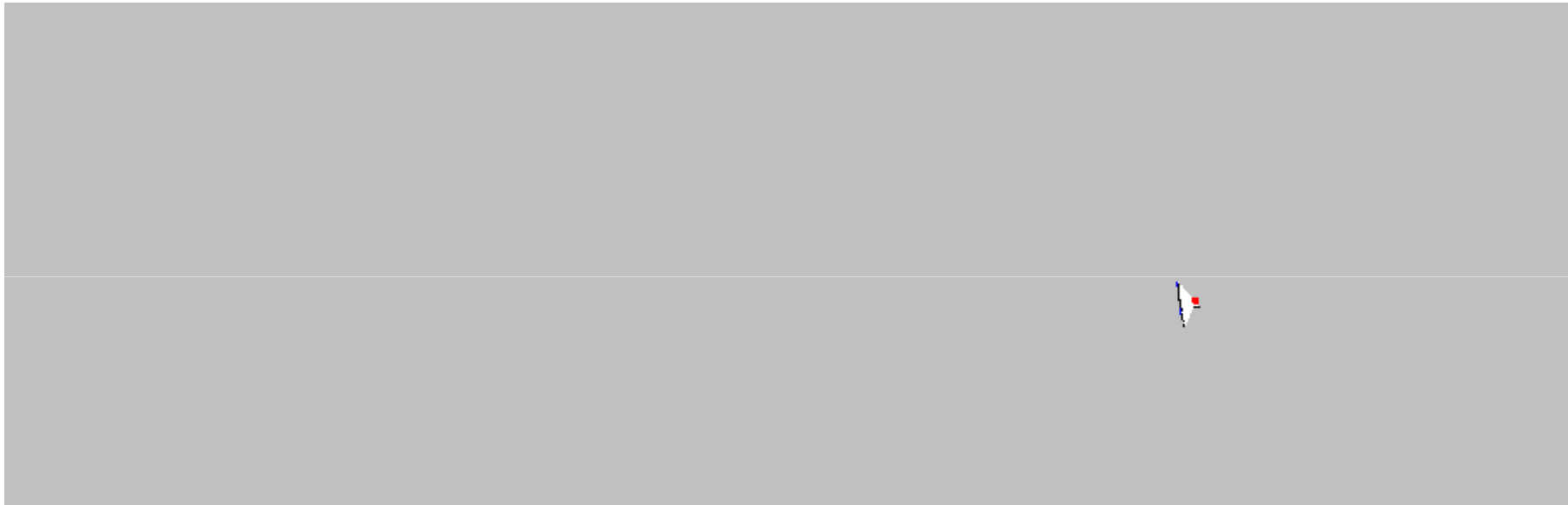
FastSLAM



Odometria

# SLAM - Multirrobo

---



Posição inicial **desconhecida.**

# Localização e Mapeamento – Robô Guia de Museu

---



# Localização e Mapeamento em Ambientes Urbanos

---

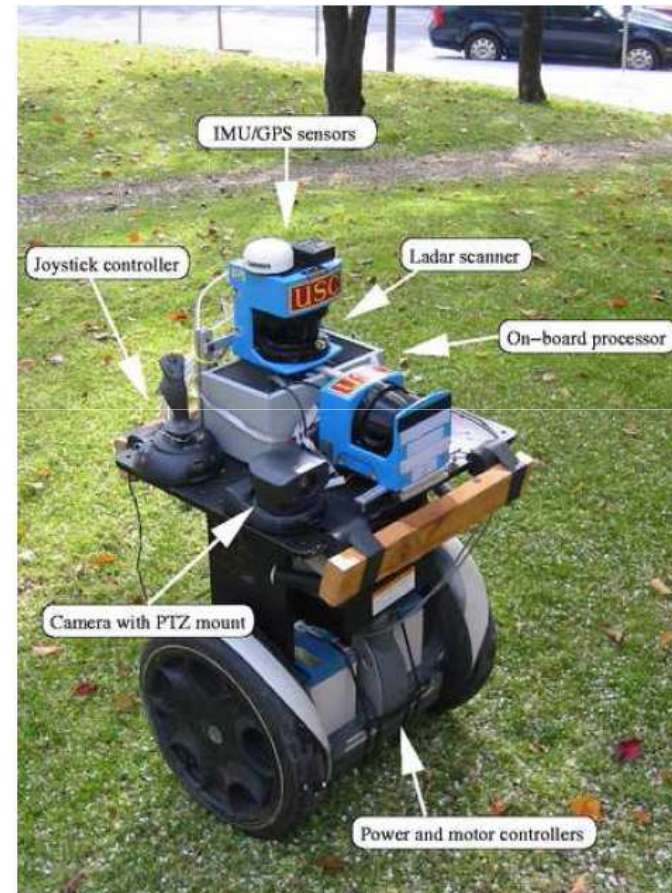
## Problemas:

- Complexidade
- Escala
- Irregularidade do terreno
- Difícil representação





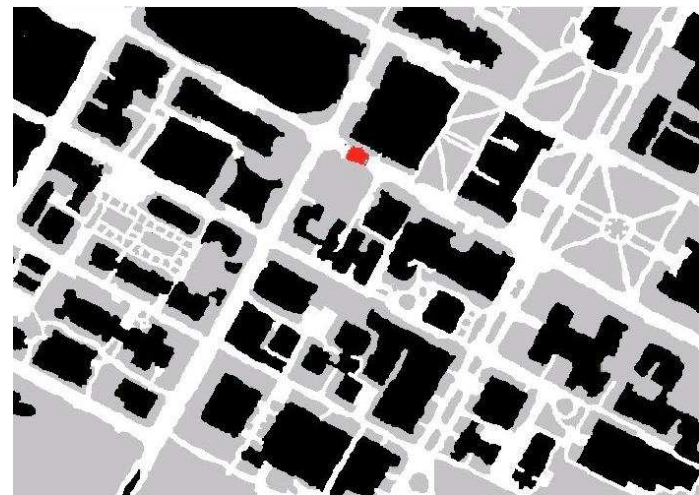
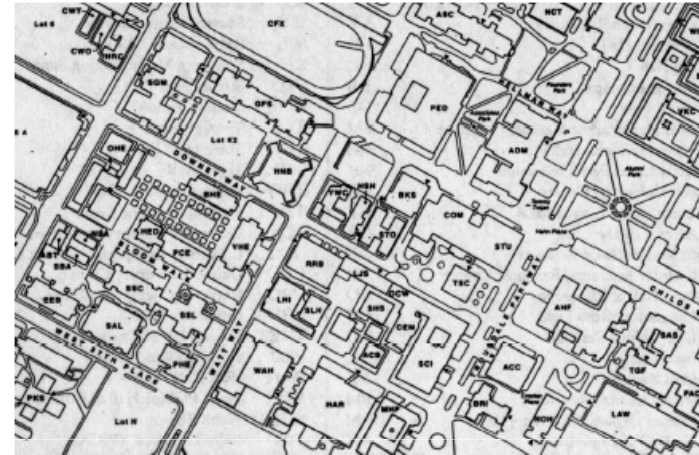
# Plataforma Experimental



# Localização – Monte Carlo

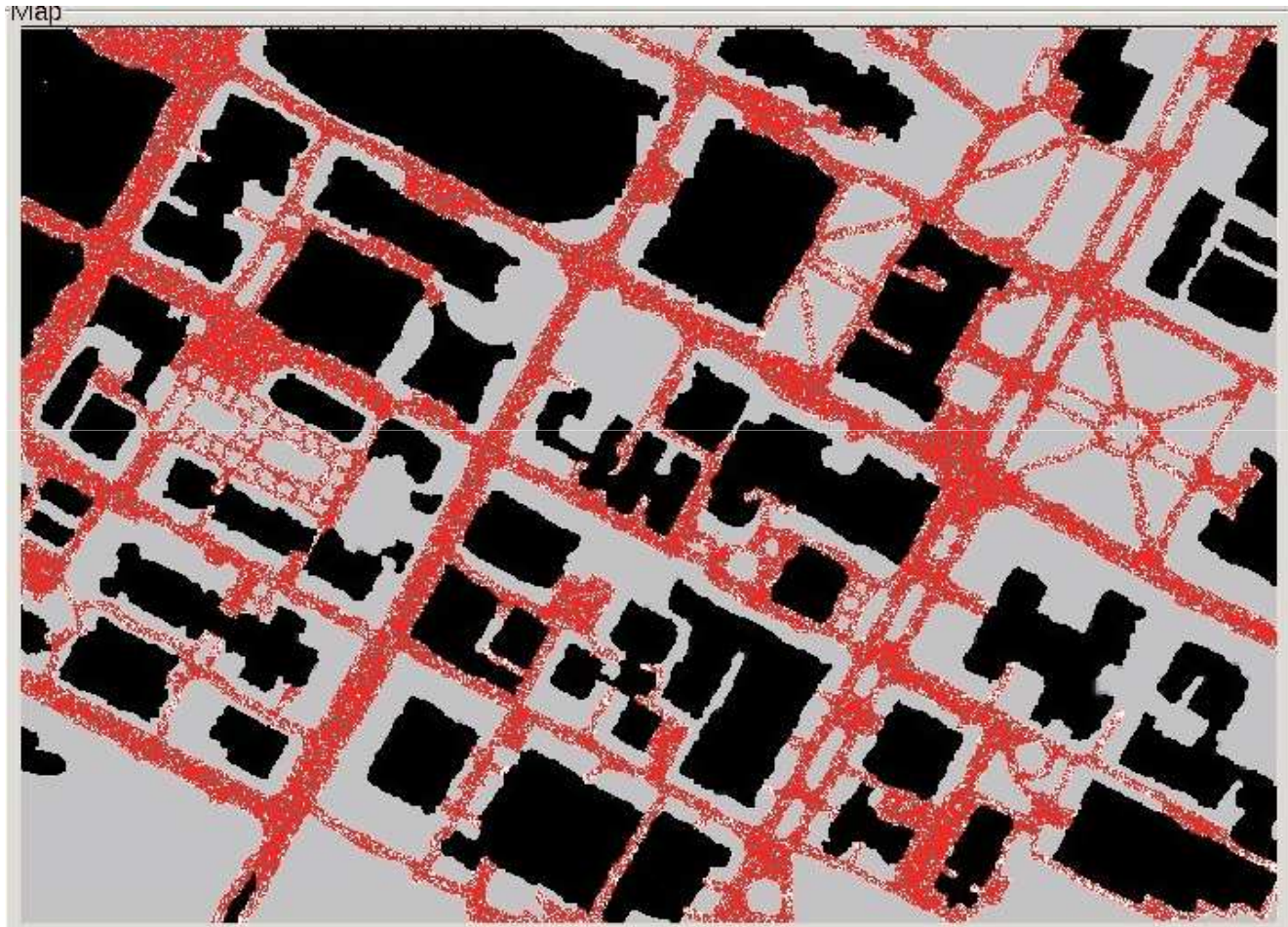
## Solução:

- Grande número de partículas
- Criação de áreas semi-ocupadas.
- Obtenção de pitch e roll por uma unidade de medida inercial
- Depois de localizar o robô, estima-se a trajetória utilizando o filtro de partículas no sentido contrário.

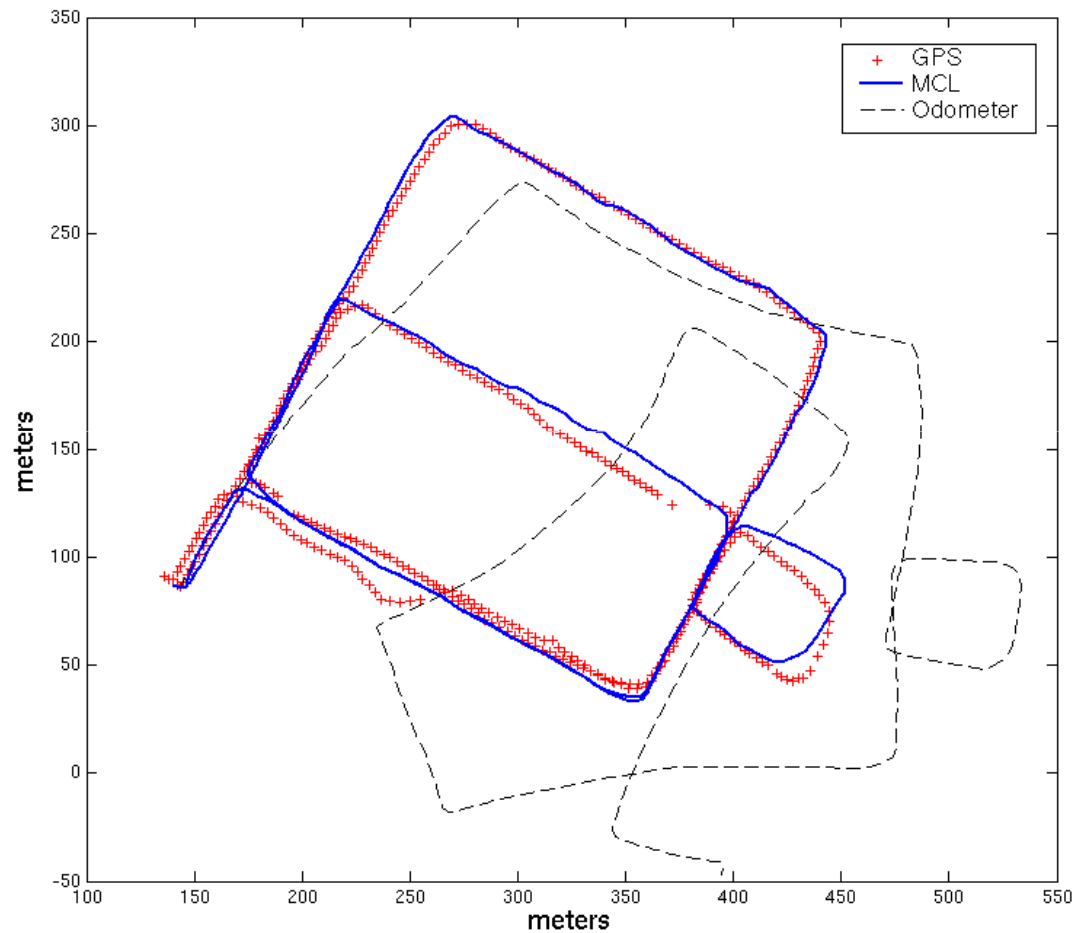


# Localização – Monte Carlo

---

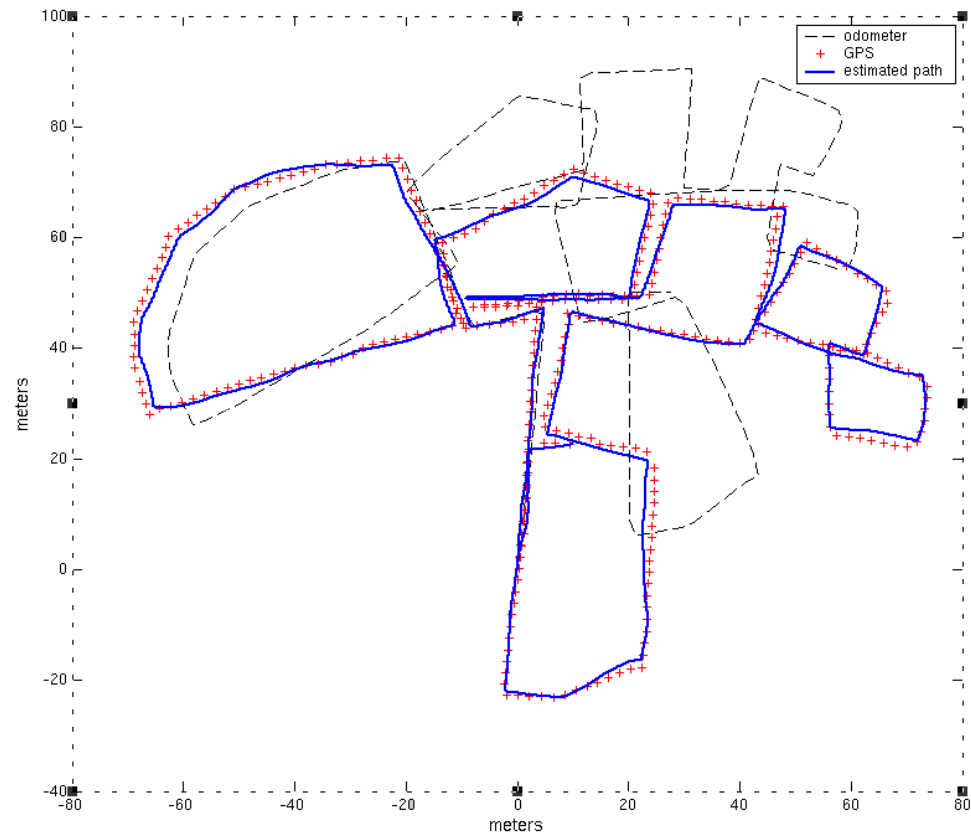
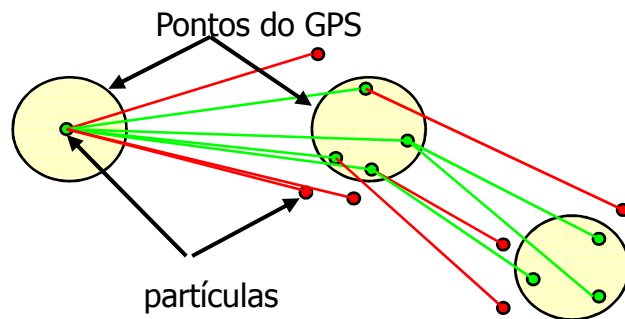


# Localização - Resultados



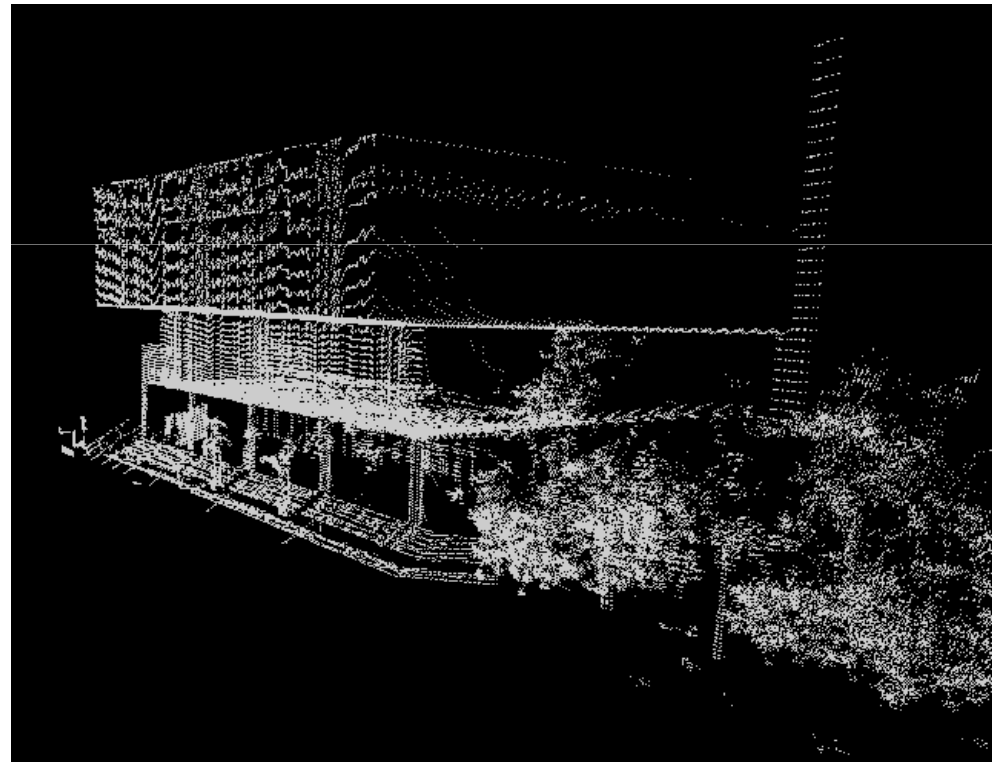
# Localização – Filtro de Partículas e GPS

- Cada partícula representa uma possível trajetória completa do robô
- É atribuído um peso a cada partícula de acordo com sua proximidade do GPS.
- Partículas que divergem do GPS recebem peso baixo e são eliminadas.



# Mapeamento - Resultados

---



CSBC 2009 - JAI

# 70 Fundamentos



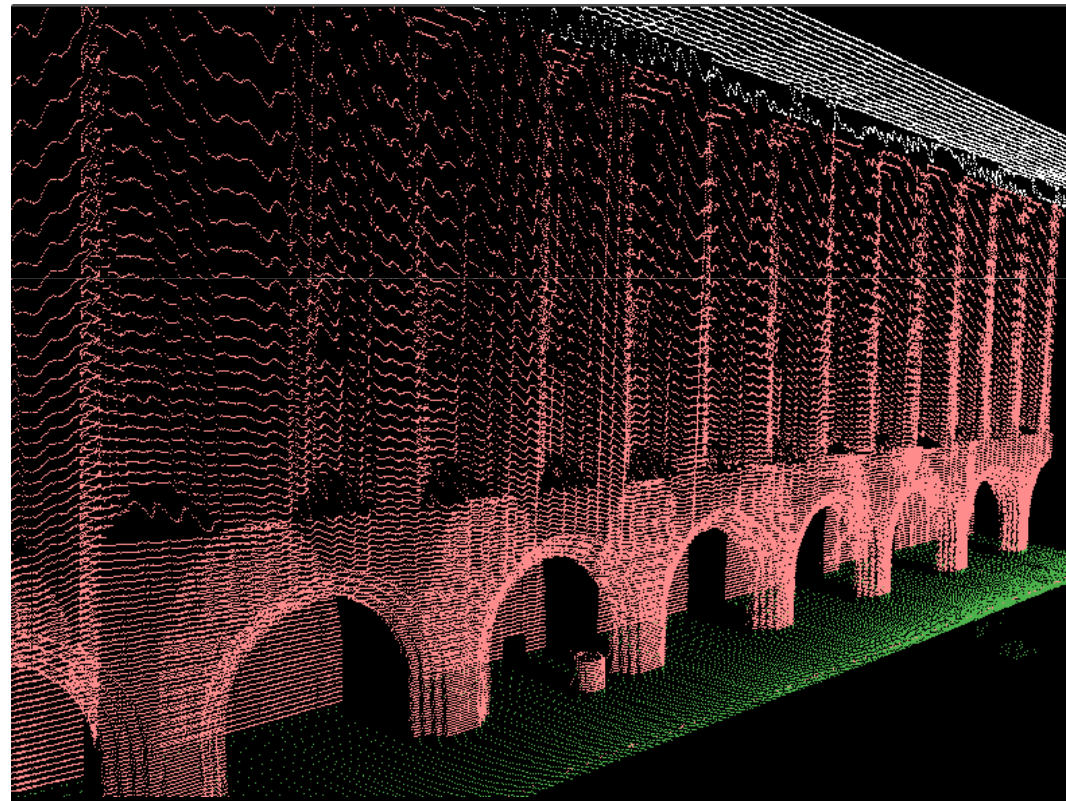
**LRM**  
*Laboratório de Robótica Móvel*

**INCT** *SEC*

Instituto Nacional de Ciência e Tecnologia  
em **Sistemas Embarcados Críticos**

# Mapeamento - Resultados

---



CSBC 2009 - JAI

# 71 Fundamentos



**LRM**  
Laboratório de Robótica Móvel

**INCT** SEC

Instituto Nacional de Ciência e Tecnologia  
em **Sistemas Embarcados Críticos**

# Mapeamento - Resultados

---



CSBC 2009 - JAI

# 72 Fundamentos



**LRM**  
*Laboratório de Robótica Móvel*

**INCT** *SEC*

Instituto Nacional de Ciência e Tecnologia  
em **Sistemas Embarcados Críticos**



# Mapeamento – Aquisição de dados

---



CSBC 2009 - JAI

# 73 Fundamentos



**LRM**  
Laboratório de Robótica Móvel

**INCT** *SEC*

Instituto Nacional de Ciência e Tecnologia  
em **Sistemas Embarcados Críticos**

# Robotic Embedded Systems Lab.

## 3D Map of McKenna Range, MOUT Site

Denis Wolf & Gaurav Sukhatme

CSBC 2009 - JAI

# 74 Fundamentos



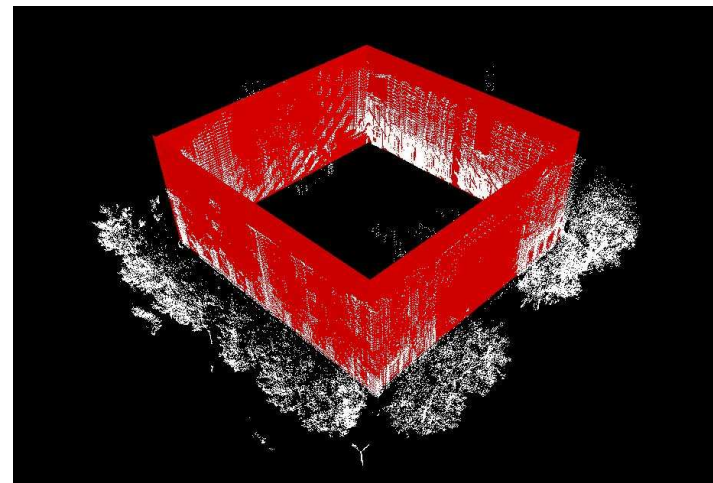
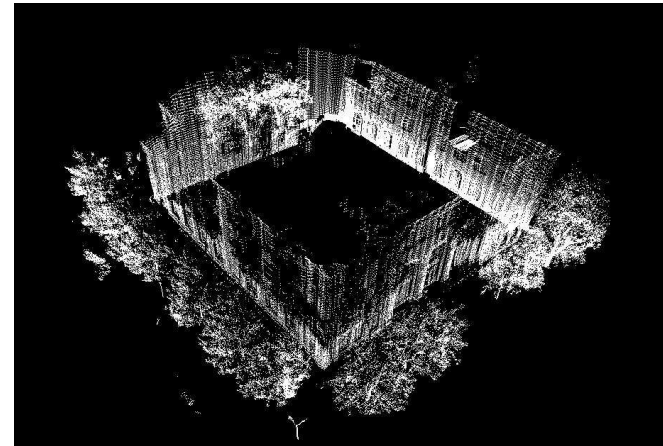
**LRM**  
*Laboratório de Robótica Móvel*

**INCT** *SEC*

Instituto Nacional de Ciência e Tecnologia  
em **Sistemas Embarcados Críticos**

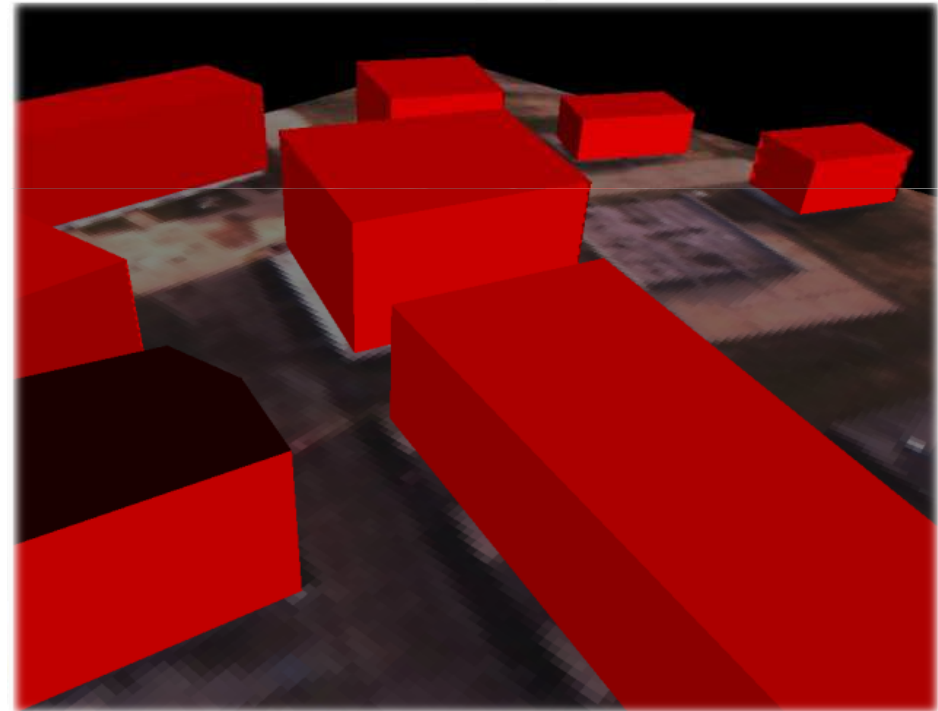
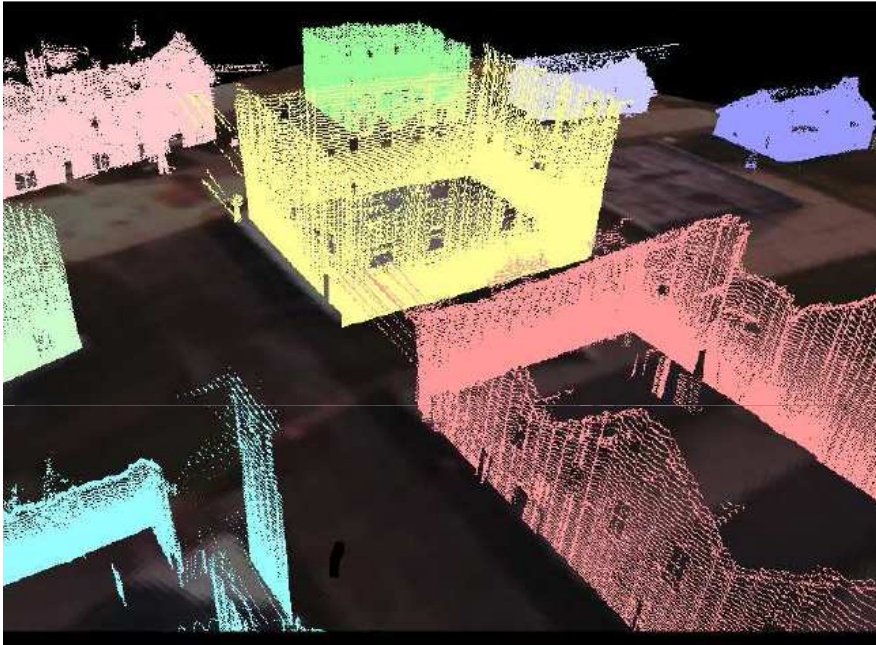
# Outras Representações para Mapas 3D

---



# Mapas 3D

---



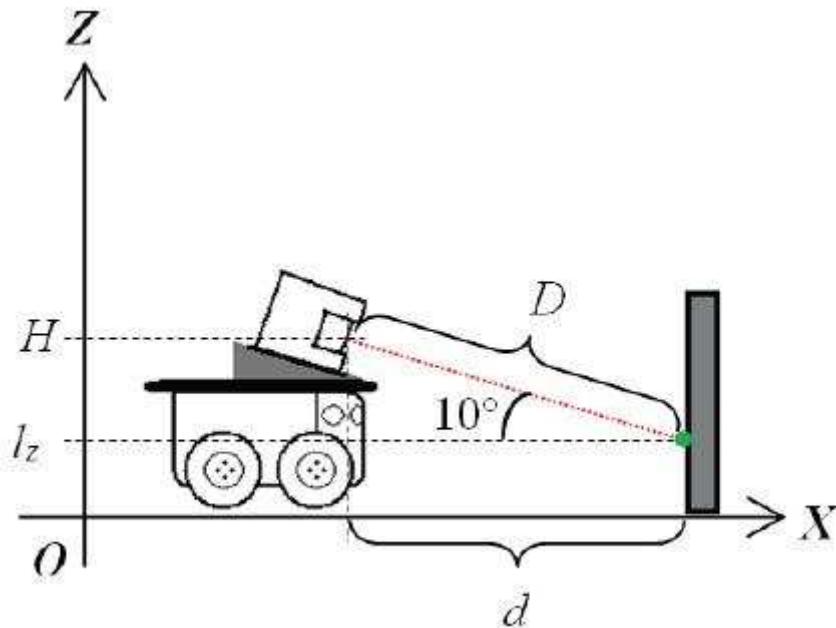
# Mapeamento 3D – ICMC/USP



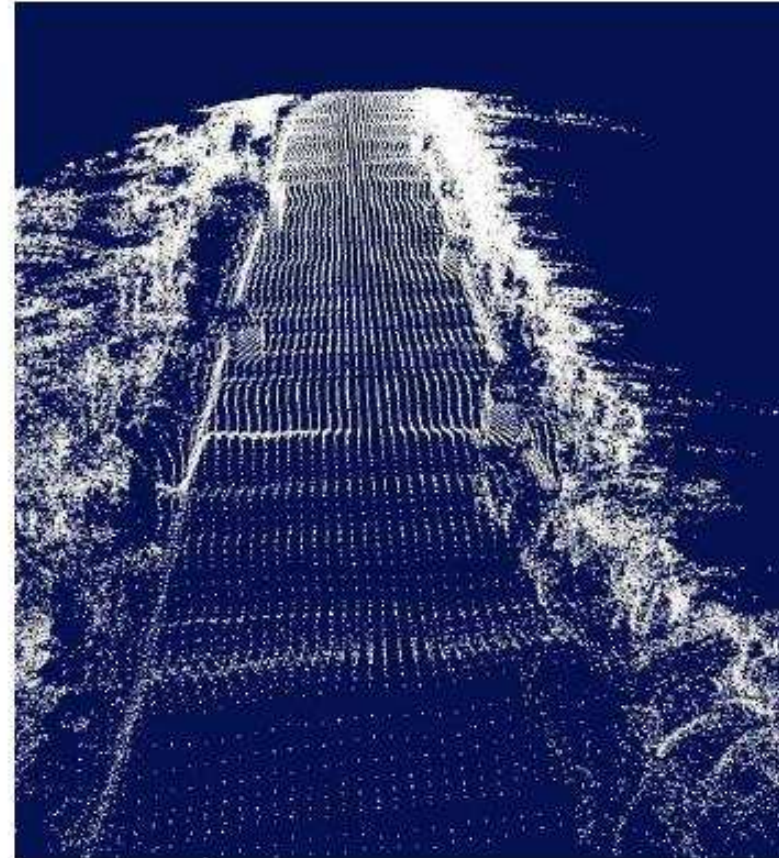
ICMC – Bloco 1



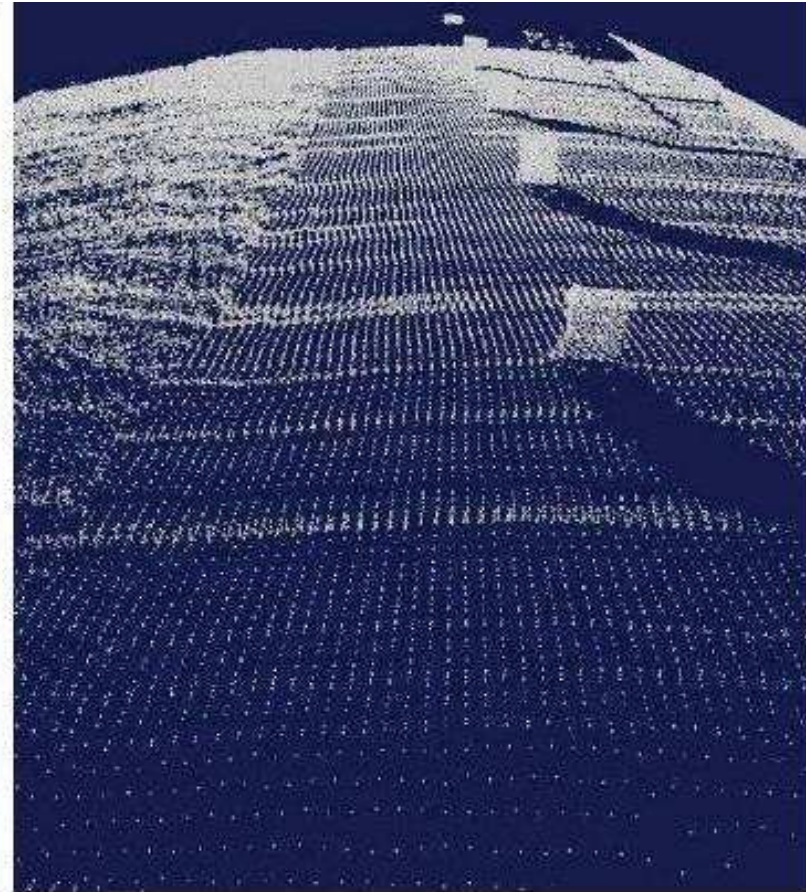
# Mapeamento de Terreno



# Mapeamento de Terreno



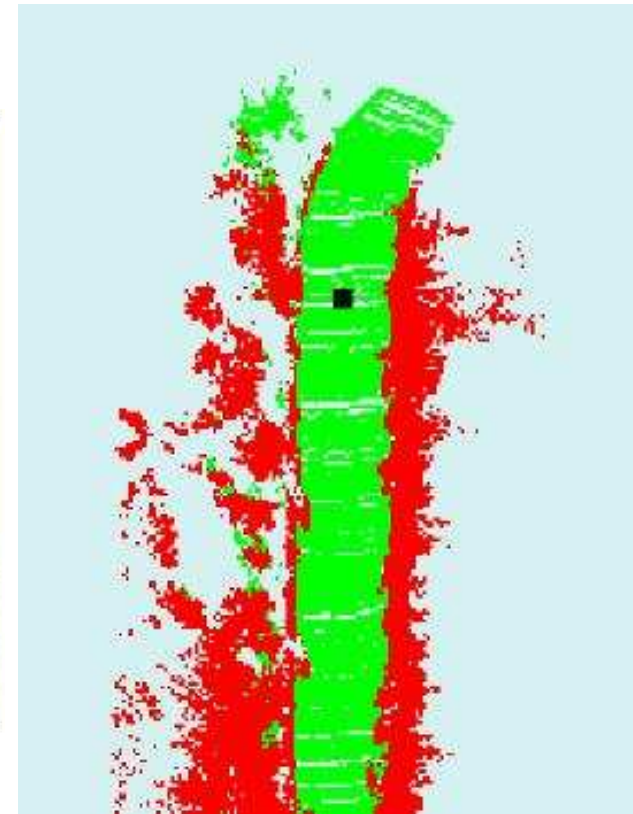
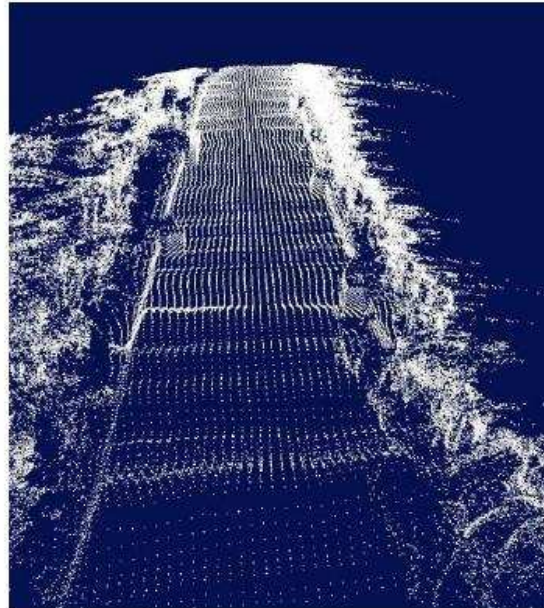
# Mapeamento de Terreno



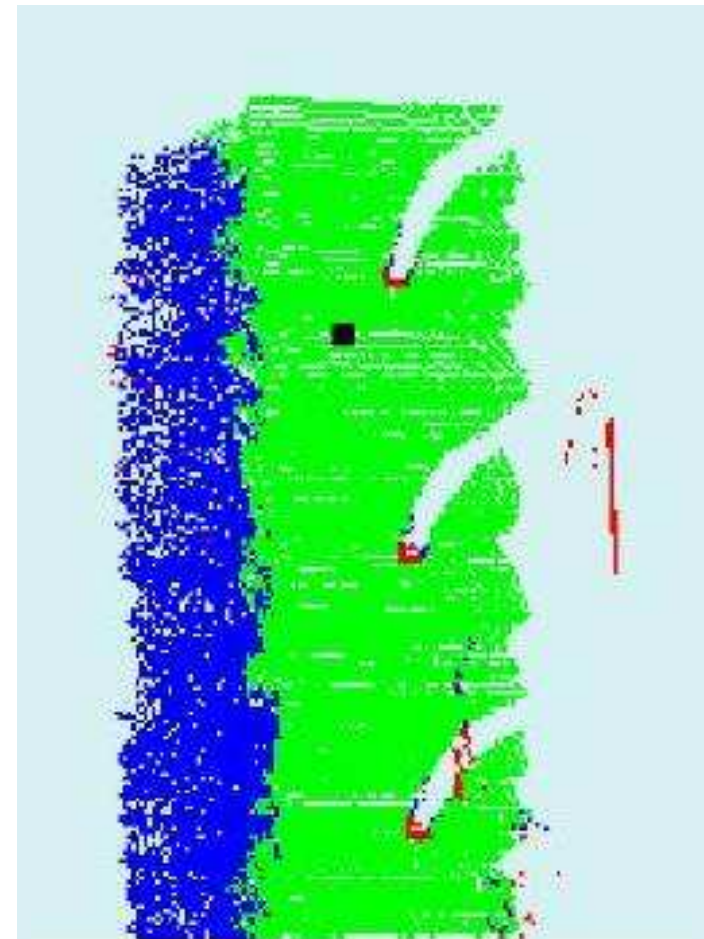


# Classificação de Terreno – Redes Neurais

---



# Classificação de Terreno – Redes Neurais



# Projeto – Veículo Autônomo



CSBC 2009 - JAI

# 83 Fundamentos



**LRM**  
Laboratório de Robótica Móvel

**INCT** SEC

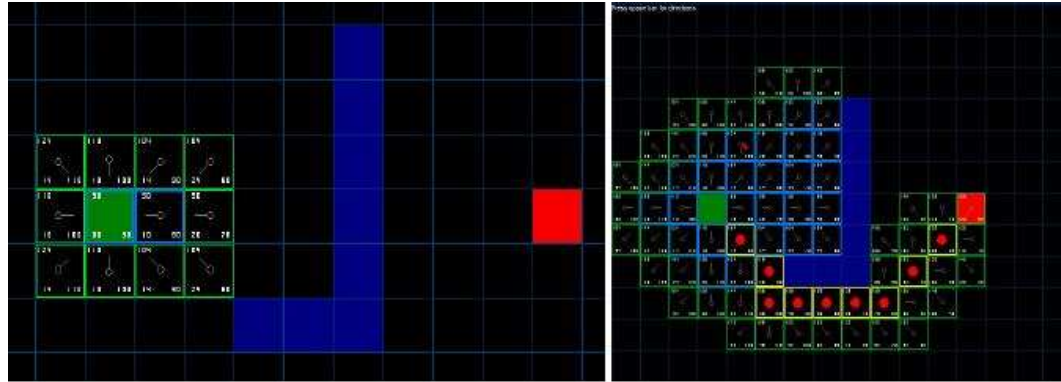
Instituto Nacional de Ciência e Tecnologia  
em **Sistemas Embarcados Críticos**

# Robótica Móvel

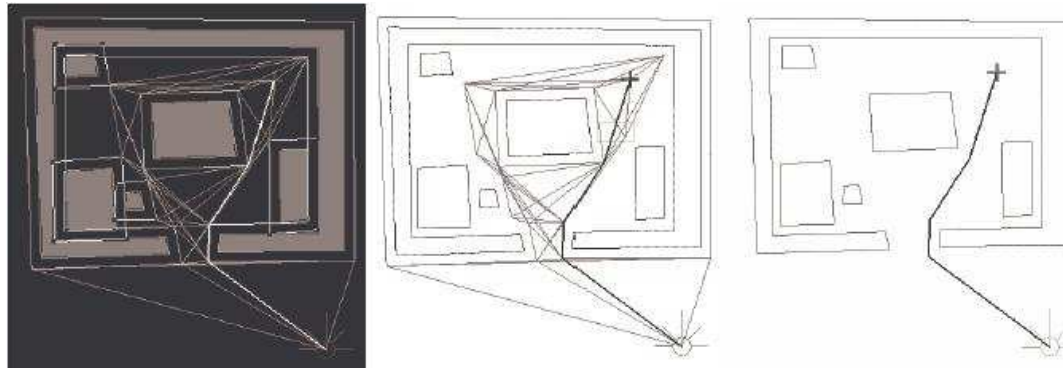
---

- Simuladores de Robôs Móveis
  - Player/Stage/Gazebo
- **Sistemas Robóticos Móveis Inteligentes**
  - Localização
  - Mapeamento
  - SLAM
  - **Navegação**

# Planejamento de Trajetória

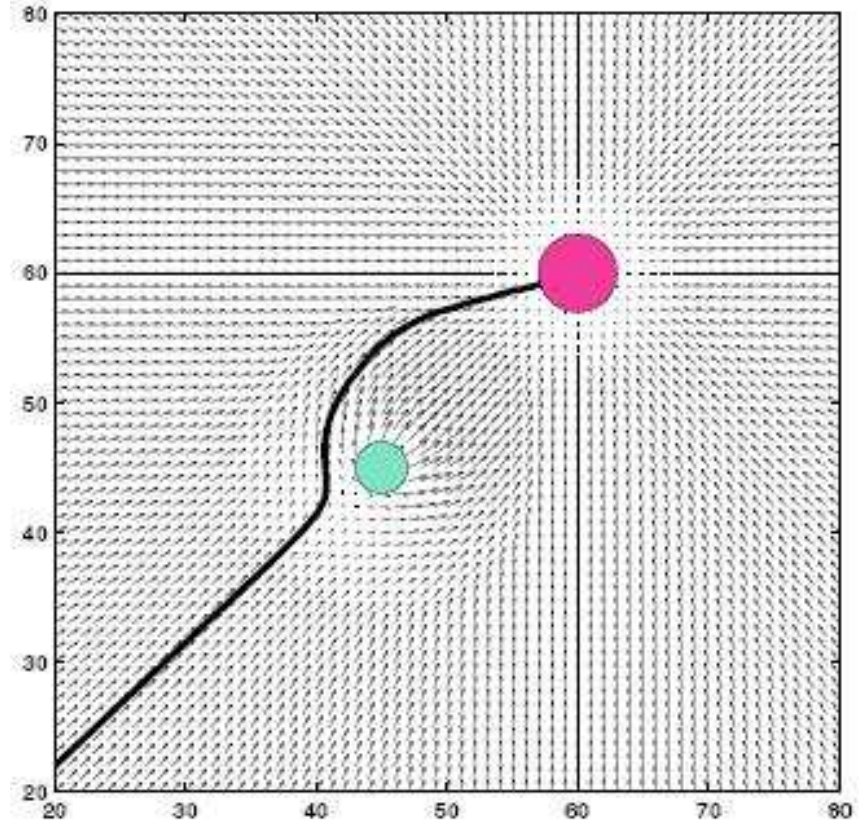
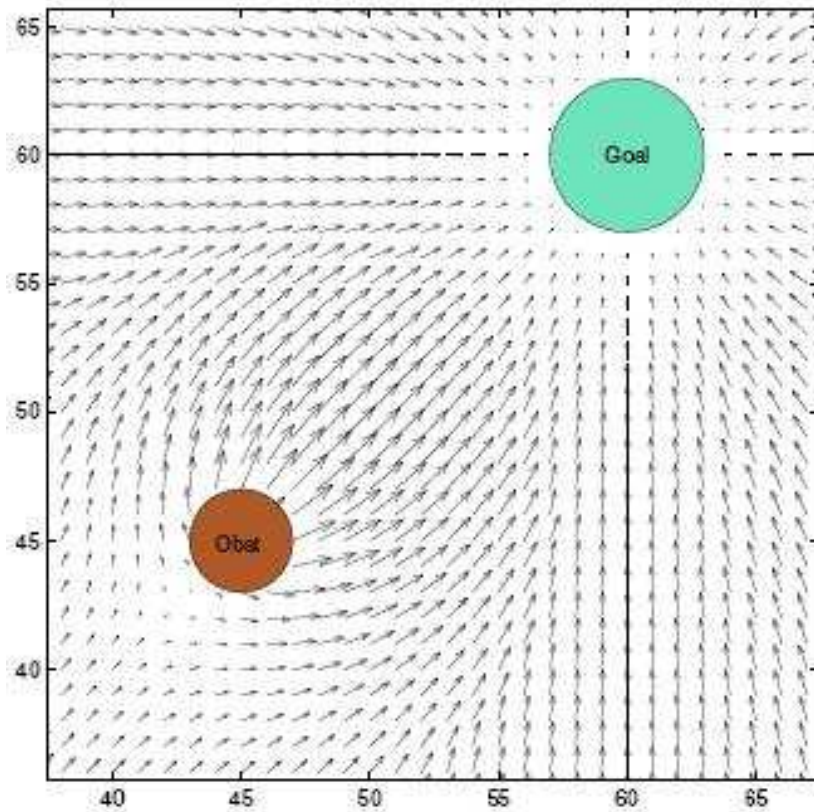


**A\*:** mapas métricos (grid)

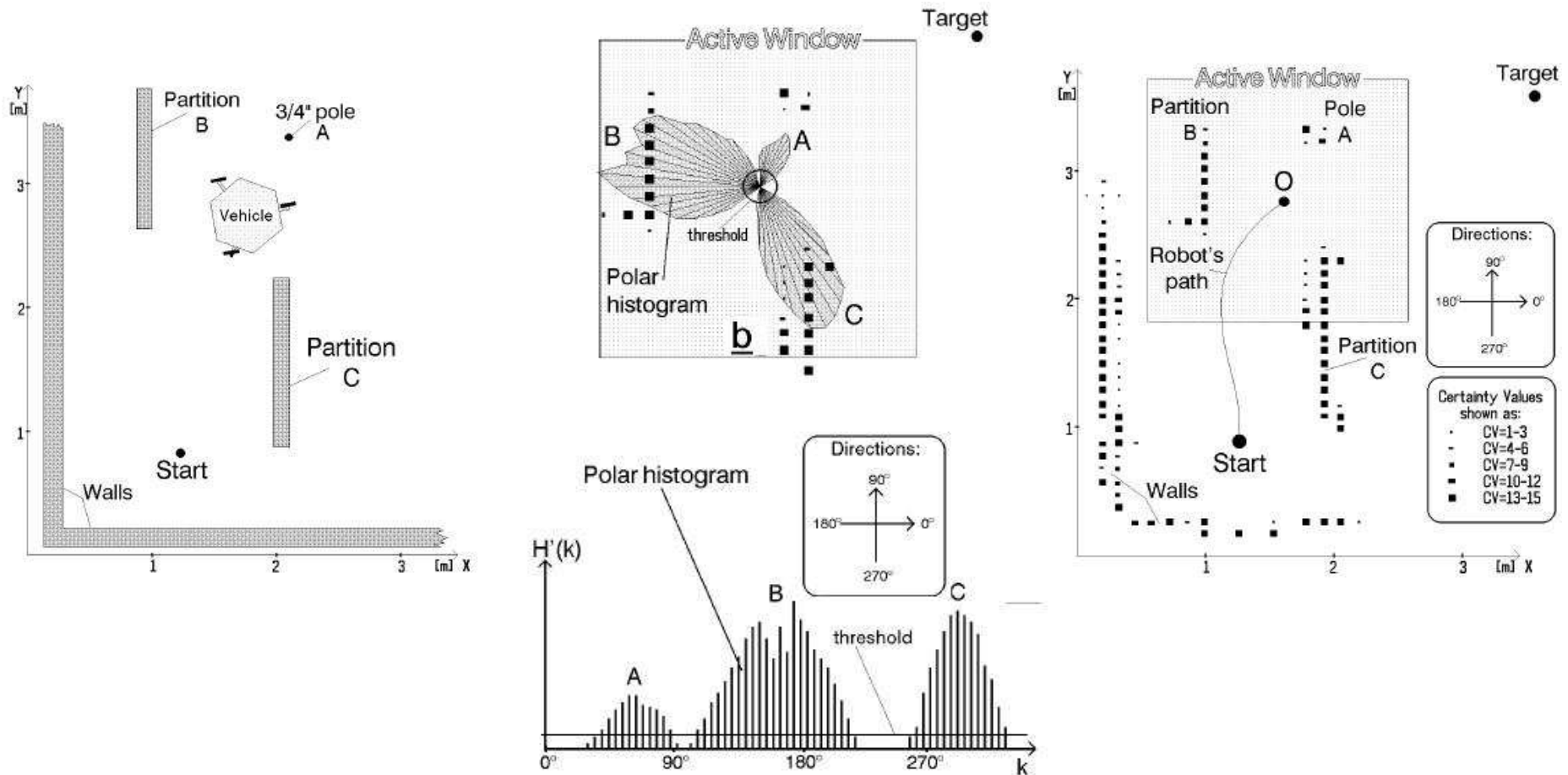


**Dijkstra:** mapas métricos/topológicos (grafos)

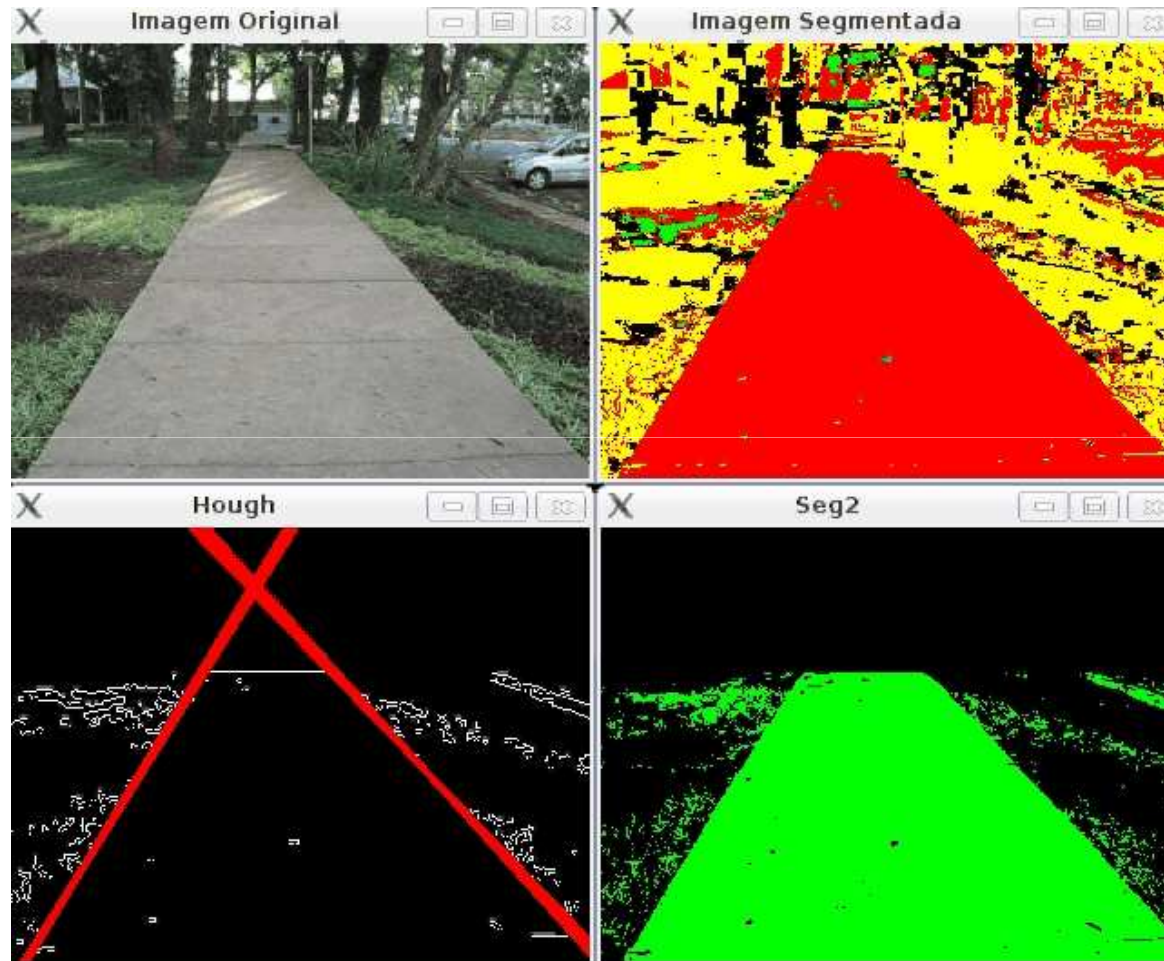
# Campos Potencias



# Vector Field Histogram - VFH



# Navegação Visual





# OBRIGADO!

---



# LRM

*Laboratório de Robótica Móvel*

[Http://www.icmc.usp.br/~lrm](http://www.icmc.usp.br/~lrm)

**Denis Fernando Wolf** – [denis@icmc.usp.br](mailto:denis@icmc.usp.br)  
**Eduardo do Valle Simões** – [simoes@icmc.usp.br](mailto:simoes@icmc.usp.br)  
**Fernando Santos Osório** - [fosorio@icmc.usp.br](mailto:fosorio@icmc.usp.br)  
**Onofre Trindade Junior** - [otjunior@icmc.usp.br](mailto:otjunior@icmc.usp.br)

CSBC 2009 - JAI

# 89 FINAL



**LRM**  
*Laboratório de Robótica Móvel*

**INCT** *SEC*

Instituto Nacional de Ciência e Tecnologia  
em **Sistemas Embarcados Críticos**

