UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

RENATA DAS CHAGAS NEULAND

Uma Hibridização do Método de Monte Carlo com Técnicas Intervalares para o Problema de Localização Global

Dissertação apresentada como requisito parcial
para a obtenção do grau de
Mestre em Ciência da Computação

Profa. Dra. Mariana Luderitz Kolberg
Orientador

Prof. Dr. Edson Prestes e Silva Junior
Co-orientador

Porto Alegre, junho de 2014

CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Neuland, Renata das Chagas

Uma Hibridização do Método de Monte Carlo com Técnicas Intervalares para o Problema de Localização Global / Renata das Chagas Neuland. – Porto Alegre: PPGC da UFRGS, 2014.

100 f.: il.

Dissertação (mestrado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2014. Orientador: Mariana Luderitz Kolberg; Co-orientador: Edson Prestes e Silva Junior.

1. Auto localização global. 2. Hibridização. 3. Análise de intervalos. 4. Filtro de partículas. I. Kolberg, Mariana Luderitz. II. Junior, Edson Prestes e Silva. III. Título.

AGRADECIMENTOS

Considerando este trabalho como resultado de uma caminhada que foi iniciada antes do meu ingresso na UFRGS, agradecer não é uma tarefa fácil. Para não ser injusta, agradeço de antemão a todos que de alguma forma contribuíram para a construção não apenas deste trabalho, mas também de quem eu sou hoje.

Agradeço, especialmente, a algumas pessoas pela contribuição:

Aos meus pais, que pacientemente me guiaram até aqui, garantindo que eu tivesse todo o carinho e suporte necessários para vencer os desafios em meu caminho. Agradeço pela motivação e consolo nas horas mais difíceis e pelas alegrias compartilhadas mesmo que a distância.

Aos meus orientadores Mariana Kolberg e Edson Prestes que me auxiliaram nessa caminhada. Pela motivação e dedicação constantes.

A todos os mestres que me ajudaram a chegar até aqui, seus ensinamentos científicos e pessoais serão sempre uma fonte de inspiração.

Aos meus amigos e colegas de laboratório, sem vocês certamente essa teria sido uma etapa muito difícil de ser vencida. Um agradecimento especial ao Renan Maffei que alem de um amigo foi também um excelente orientador não oficial. A amiga Cristina Otto que me ajudou a superar as fases difíceis e sempre esteve disposta a me ouvir (e a dividir um pote de sorvete).

A Deus por colocar as oportunidades no meu caminho e me dar força para enfrentar os obstáculos.

E por fim, a CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior) pela concessão da bolsa durante todo o período de realização deste mestrado.

Obrigada a todos!

“Go placidly amid the noise and the haste, and remember what peace there may in the silence.”

— MAX EHRMANN

A Hybridization of the Monte Carlo Method with Interval Techniques applied to Global Localization Problem

ABSTRACT

Probabilistic approaches are extensively used to solve high-dimensionality problems in many different fields. The particle filter is a prominent approach in the field of Robotics, due to its adaptability to non-linear models with multi-modal distributions. Nonetheless, its result is strongly dependent on the quality and the number of samples required to cover the space of possible solutions.

In contrast, interval analysis deals with high-dimensionality problems by reducing the space enclosing the actual solution. These reductions are made through interval techniques that guarantee mathematically that the searched solution is contained in the result of the method, once the problem modeling has been done correctly. Interval methods do not discard any feasible solutions, with the result that may be may be too conservative. Notwithstanding, it cannot precise where in the resulting subspace the actual solution is.

We devised a strategy that combines the best of both worlds. The main idea of the proposed method is to use interval techniques to improve the performance of the particle filter, limiting the scattering of particles and accelerating the convergence of the method. We expect that the proposed method can make the spread and control of particles more efficiently, possibly resulting in a more accurate method. Our approach is illustrated by solving the global localization problem for underwater robots.

Keywords: global localization, hybridization, interval analysis, particle filter.

RESUMO

Abordagens probabilísticas são extensivamente utilizadas para resolver problemas de alta dimensionalidade em diferentes campos. O filtro de partículas é uma abordagem proeminente no campo da Robótica, devido a sua adaptabilidade a modelos não lineares com distribuições multimodais. Contudo, seus resultados são fortemente dependentes da qualidade e do número de amostras requeridas para cobrir o espaço de busca.

Em contrapartida, análise de intervalos lida com problemas de alta dimensionalidade através da redução do espaço de busca. Essas reduções são feitas através de técnicas intervalares que garantem matematicamente que a solução procurada está contida no resultado do método, uma vez que a modelagem do problema tenha sido feita corretamente. Métodos intervalares não descartam quaisquer soluções factíveis, com isso o resultado pode ser pouco representativo. Não obstante, não é possível definir precisamente onde a solução está no intervalo definido como solução.

A estratégia proposta combina o melhor das duas abordagens. A ideia principal do método proposto é usar técnicas intervalares para melhorar os resultados do filtro de partículas, limitando o espalhamento das partículas e acelerando a convergência do método. Nós esperamos que o método proposto consiga fazer a distribuição e controle de partículas de forma mais eficiente, resultando possivelmente em um método mais preciso. A abordagem proposta é ilustrada através do tratamento do problema de localização global de robôs subaquáticos.

Palavras-chave: Auto localização global, hibridização, análise de intervalos, filtro de partículas.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ALGORITMOS

LISTA DE ABREVIATURAS E SIGLAS

BPF	Box Particle Filter
CSP	Constraint Satisfaction Problem
HC	Hull Consistency
MCL	Monte Carlo Localization
SIVIA	Set Inversion Via Interval Analysis
SLAM	Simultaneous Localization and Mapping

SUMÁRIO

1 INTRODUÇÃO

A criação de robôs autônomos é um dos maiores desafios da robótica. Para que um robô seja realmente autônomo, ele deve ser capaz de se movimentar e cumprir suas tarefas sem interferência humana. Para tanto, é muito importante que o robô consiga definir sua localização no ambiente. Uma localização imprecisa pode não apenas dificultar a realização das tarefas, por mais simples que estas sejam, como ainda pode trazer sérios riscos à operação do robô. Em tarefas de detecção de minas, por exemplo, conhecer a localização do robô é essencial para definir a localização das minas detectadas e não colidir com uma acidentalmente.

Problemas de localização tratam a correspondência entre as coordenadas de um mapa do ambiente no qual o robô está inserido, e os dados retornados por seus sensores. Se o problema fosse apenas encontrar a correspondência entre a leitura obtida pelo robô e o mapa do ambiente, a tarefa não seria tão complexa como é na realidade. O fator considerado como principal causador do aumento da complexidade do problema, é a necessidade de tratar as incertezas sobre as informações. Essas incertezas são provenientes, em sua maioria, das medições imprecisas retornadas pelos sensores (GUIBAS; MOTWANI; RAGHAVAN, 1995) (THRUN et al., 2005).

Outra fonte a ser considerada é a possível existência de ambiguidades no próprio ambiente. O fato de um ambiente ter lugares diferentes que retornam leituras muito semelhantes pode confundir o robô no processo de localização. Exemplos desses lugares são os cantos de salas. Em uma sala quadrada e sem objetos, para cada leitura obtida de um determinado local, existem pelo menos mais três lugares que retornariam a mesma leitura. O ponto chave para contornar tal problema está na análise de sucessivas observações feitas pelo robô. Ao impormos restrições entre observações consecutivas acabamos por reduzir as ambiguidades do ambiente, permitindo o aprimoramento da localização do robô.

1.1 Motivação

A incerteza está presente em diversos problemas da robótica, e a resolução desses problemas depende diretamente do tratamento adequado das incertezas envolvidas. Duas principais abordagens usadas para tratar incertezas são a probabilística (THRUN et al., 2005) e a intervalar (JAULIN et al., 2001).

Abordagens probabilísticas, como as baseadas em filtragem bayesiana, são extensiva-

mente usadas para tratar problemas de alta dimensionalidade em diversos campos. Dentre elas, o filtro de partículas (DELLAERT et al., 1999) é uma abordagem bastante utilizada na robótica, devido principalmente a suas características que permitem tratar modelos não lineares e crenças multimodais. Contudo, por estimar distribuições de probabilidades usando uma aproximação de Monte Carlo, a qualidade da solução encontrada com o filtro de partículas é dependente do número de partículas utilizadas. Se a incerteza do problema for muito grande, o filtro de partículas pode exigir muitas partículas para cobrir o espaço de soluções, tornando o seu uso proibitivo.

Em contrapartida, abordagens intervalares lidam com problemas de alta dimensionalidade através da redução de domínios, essas reduções são feitas através das restrições do problema. A partir da correta modelagem do problema, o resultado de um método intervalar é matematicamente garantido. Contudo, dada a sua característica conservativa, onde nenhuma solução factível é eliminada, a solução final pode ser pouco significativa.

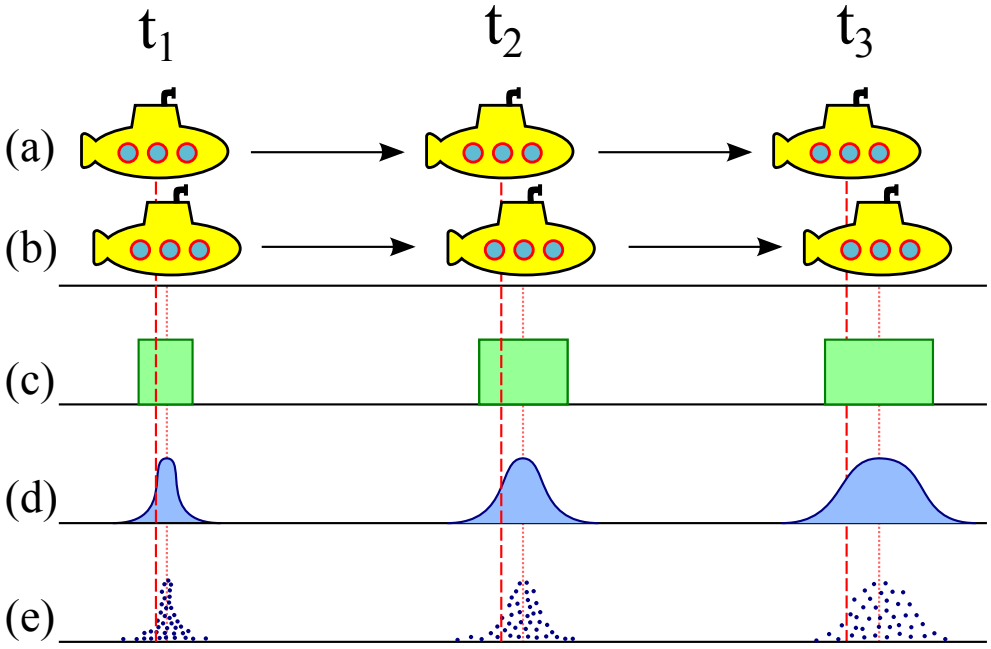


Figura 1.1: Modelagem da incerteza

A Figura 1.1 apresenta uma comparação das abordagens intervalar e filtro de partículas no que se refere a modelagem da incerteza de localização. A figura mostra o comportamento dos métodos no decorrer do tempo (t1, t2, t3). Em (a) são mostradas as posições reais do robô no tempo. Dado que a informação sobre a localização do robô é extraída de sensores, ela está sujeita a erros. Esses erros são representados em (b) , onde é apresentada a localização do robô de acordo com os dados retornados pelos sensores.

Ainda na Figura 1.1 é mostrada a modelagem da incerteza de acordo com as abordagens intervalar em (c) e probabilística em (d). A abordagem intervalar usa um intervalo para definir a localização do robô considerando a incerteza associada aos sensores. Com o passar do tempo e o aumento da incerteza, o intervalo também aumenta, de forma que a real posição do robô estará contida nesse intervalo. Entretanto não existe informações sobre onde está a real posição dentro desse intervalo. Em (d) a incerteza é modelada através

de uma gaussiana, comumente utilizadas em abordagens probabilísticas. O pico da gaussiana representa o local com maior probabilidade de o robô estar localizado. Entretanto, considerando que o problema abordado não tem uma distribuição alvo conhecida e pode ser multimodal, a representação da incerteza através de gaussianas pode ser inadequada. Uma possibilidade para contornar tal problema é estimar a distribuição de probabilidades referente à localização do robô usando técnicas de amostragem. Em (e), é apresentado um exemplo de modelagem usando filtro de partículas, no qual as partículas (hipóteses sobre a localização do robô) são espalhadas para cobrir a região de incerteza. Caso a região de incerteza seja muito grande o número de partículas necessárias para alcançar um bom resultado é muito alto.

A partir dos conceitos e resultados apresentados por essas diferentes abordagens, Abdallah, Gning e Bonnifait (2008) criaram o *Box Particle Filter* (BPF), o qual utiliza técnicas intervalares juntamente com o filtro de partículas. Os resultados apresentados pelos autores mostraram ganho em tempo de execução, entretanto comparações sobre precisão não mostraram ganho.

1.2 Objetivo e Contribuições

Neste trabalho é proposto um método híbrido para tratar o problema de auto localização global a partir de técnicas intervalares e filtro de partículas. O principal objetivo do método é construir uma distribuição inteligente das partículas, reduzindo a área de incerteza sobre a localização do robô e aumentando a cobertura da área de incerteza, consequentemente melhorando a precisão do resultado. Outra contribuição do método está no fato de que os importantes ganhos na precisão do resultado não implicam em perdas expressivas no tempo gasto pelo método. Além disso, o método é capaz de detectar casos de má convergência e reinicializar o processo de localização.

1.3 Metodologia

Este trabalho teve início com um estudo sobre o estado da arte do problema abordado e algumas das principais soluções propostas. O foco da pesquisa foi o estudo de trabalhos de grandes pesquisadores como Sebastian Thrun (THRUN et al., 2005) (THRUN, 2002) e Luc Jaulin (JAULIN et al., 2001) (JAULIN, 2009). Apesar das diferentes abordagens utilizadas por esses autores, ambos mostram sucesso no tratamento do problema de localização.

O trabalho de Abdallah, Gning e Bonnifait (2008) foi um dos principais motivadores para a proposta de uma solução híbrida. Assim como no trabalho deles, a solução proposta nesta pesquisa usa o método de Filtro de Partículas em conjunto com técnicas intervalares. Diferentemente dos autores, que obtiveram melhorias em relação ao tempo de execução do método, nosso trabalho visa melhorar principalmente a precisão da solução.

O passo seguinte foi a implementação do método de filtro de partículas. Nesta primeira versão, para fins de aprendizagem, o filtro de partículas tratava o problema de localização global em ambientes 2D estruturados utilizando laser. Dado que o ambiente alvo dessa pesquisa é subaquático e desestruturado, estudou-se uma opção para extração

de informações que auxiliem na localização do robô independente de obstáculos. A opção escolhida foi o uso de marcadores distinguíveis. O método de filtro de partículas implementado foi adaptado para este novo cenário sem obstáculos.

Visando o aumento na precisão dos resultados do filtro de partículas, usou-se análise de intervalos para reduzir o espalhamento das partículas, concentrando-as em áreas de real interesse. Uma primeira tentativa foi utilizar o método SIVIA (JAULIN; WALTER, 1993) junto com o filtro de partículas. Após testes preliminares foi possível perceber que, apesar do aumento na precisão, o custo associado ao SIVIA é muito alto. Neste contexto, optou-se por testar outra técnica, usando contratores. Contratores (JAULIN et al., 2001) podem não prover resultados tão precisos quanto o SIVIA, mas seu custo computacional é menor. Tendo os métodos implementados mostrado bons resultados o problema foi expandido para um ambiente 3D.

Por fim, os métodos foram testados e comparados em experimentos simulados, utilizando diversas trajetórias e configurações de ambientes.

1.4 Organização

Este trabalho está organizado da seguinte maneira:

- Capítulo 2: apresenta uma introdução à análise de intervalos, focando em conceitos e técnicas fortemente aplicadas para o tratamento de problemas da robótica.
- Capítulo 3: mostra o problema abordado nesta pesquisa, definições e classificações do problema de auto localização. Ainda neste capítulo são apresentados algoritmos baseados em análise de intervalos e algoritmos baseados em filtro de partículas para o tratamento do problema de auto localização, assim como trabalhos relacionados com essas abordagens para problemas da robótica.
- Capítulo 4: detalha os métodos propostos. Neste capítulo algoritmos são apresentados para demonstrar o funcionamento dos métodos.
- Capítulo 5: mostra experimentos e discussões dos resultados obtidos.
- Capítulo 6: descreve as conclusões obtidas a partir deste trabalho, assim como a descrição de trabalhos futuros relacionados à pesquisa.

2 INTRODUÇÃO À ANÁLISE DE INTERVALOS

A análise de intervalos baseia-se na representação de números em um formato intervalar. Moore (1966), Nickel (1966) e Hansen (1965) foram pesquisadores que na década de 60 deram início ao estudo de técnicas intervalares. Desde então, a pesquisa na área de análise de intervalos evoluiu e vem mostrando bons resultados para problemas de diversas áreas, como por exemplo, a robótica (JAULIN et al., 2001).

Algoritmos convencionais, ou pontuais, fornecem uma estimativa de resposta, alguns até uma previsão de erro, mas não se pode afirmar com exatidão que a resposta encontrada é realmente a correta. Com o uso de técnicas intervalares as respostas passam de um formato pontual para um formato intervalar, e garantem que a resposta correta estará contida em determinado intervalo, gerando uma garantia de qualidade (DIMURO, 1991).

Além disso, a representação de um número de ponto flutuante usada pelos computadores é limitada, sendo assim, existe uma vasta gama de valores de ponto flutuante que não podem ser representados por um computador. Essa limitação pode causar erros durante as computações, podem acontecer, por exemplo, casos de *overflow* ou divisão por zero. Tais problemas podem ser evitados com a aplicação de uma abordagem intervalar (HICKEY; JU; VAN EMDEN, 2001).

Sendo assim, os erros encontrados em algoritmos pontuais derivam de três fontes (DIMURO, 1991):

- *Propagação de erro nos dados iniciais*. Alguns dados de entrada como temperatura, distância e tempo são adquiridos normalmente através de sensores. Esses dados tendem a ser imprecisos, e devido a isso necessitam um tratamento adequado.

- *Erros de arredondamento*. Esse tipo de erro ocorre pelo fato de que os computadores precisam arredondar os números de acordo com suas limitações de representação de ponto flutuante.

- *Erros de truncamento*. Esses erros são causados ao truncar sequências infinitas de operações após um número finito de etapas.

Sistemas destinados a situações de risco, como controle de aviões ou mísseis, exigem um rigoroso controle de erros. Por menores e mais insignificantes que possam parecer esses erros, uma computação imprecisa pode causar sérios danos, como de fato já ocorreu:

- Em fevereiro de 1991, durante a Guerra do Golfo, ocorreu um erro em uma bateria de mísseis usados para interceptação de misseis inimigos. Morreram 28 soldados que estavam em um quartel do exército americano. A falha foi atribuída a um erro de arredondamento no programa (ARNOLD, 2013).

- Em junho de 1996, um foguete não tripulado explodiu pouco depois do lançamento, custando uma década de desenvolvimento e US$ 7 bilhões. A falha foi atribuída a um erro na conversão de um número em ponto flutuante (ARNOLD, 2013).

A utilização do formato intervalar beneficia principalmente casos em que a representação correta de um valor é necessária. Além disso, a aritmética intervalar garante que as computações feitas com intervalos gerarão um resultado intervalar contendo o resultado real esperado (ROKNE, 2001).

Métodos intervalares são frequentemente utilizados para otimização global. Rokne (2001) cita algumas vantagens do uso de análise de intervalos:

- Os métodos são robustos e confiáveis.

- São independentes de um ponto de partida.

- A existência e unicidade das soluções podem ser provadas.

Algumas desvantagens também apontadas por Rokne (2001) são:

- Operações de subtração e divisão não são operações inversas à adição e multiplicação.

- Operações usando aritmética intervalar consomem mais tempo que a operação real correspondente.

- Computações com intervalos não descartam quaisquer soluções válidas, sendo assim, no final das computações o resultado gerado pode ser um intervalo de tamanho grande, fornecendo pouca informação sobre o resultado real.

A análise de intervalos define uma série de regras para cálculos intervalares (SEIGNEZ; LAMBERT, 2008). Dado que a utilização de cálculos intervalares vem se tornando mais frequente, foram criadas algumas bibliotecas para facilitar a programação de algoritmos intervalares, tais como IBEX (CHABERT, 2010), PROFIL/BIAS (KNUPPEL, 1994), ALIAS (MERLET, 2000), INTLAB (RUMP, 1999) e C-XSC (KLATTE; CORLISS, 1993). Nas seções seguintes serão descritos alguns conceitos e operações intervalares.

2.1 Intervalos: Definições e operações básicas

Jaulin et al. (2001) apresenta diversas definições intervalares, algumas das quais são mostradas nesta seção.

Um intervalo real $[x]$ é considerado um subconjunto conectado de $\mathbb{R}$, onde $[x]$ é composto por um limite inferior $\underline{x}$ e um limite superior $\overline{x}$. Por exemplo, $[x] = [1; 8]$, onde $\underline{x} = 1$ e $\overline{x} = 8$. É possível definir formalmente um intervalo real como:

$$[x] = [\underline{x}; \overline{x}] = \{x \in \mathbb{R} \mid \underline{x} \leq x \leq \overline{x}\}$$

Um intervalo possui um tamanho $w([x])$ e um ponto médio $mid([x])$, os quais são definidos respectivamente por:

$$w([x]) = \overline{x} - \underline{x}$$
$$mid([x]) = \frac{\underline{x} + \overline{x}}{2}$$

Operações aritméticas também são definidas para cálculos intervalares. Considerando dois intervalos reais $[a] = [\underline{a}; \overline{a}]$ e $[b] = [\underline{b}; \overline{b}]$, as operações básicas podem ser definidas como:

Soma: $[a] + [b] = [\underline{a} + \underline{b}; \overline{a} + \overline{b}]$

Subtração: $[a] - [b] = [\underline{a} - \overline{b}; \overline{a} - \underline{b}]$

Multiplicação: $[a].[b] = [\min\{\underline{a}.\underline{b}, \underline{a}.\overline{b}, \overline{a}.\underline{b}, \overline{a}.\overline{b}\}; \max\{\underline{a}.\underline{b}, \underline{a}.\overline{b}, \overline{a}.\underline{b}, \overline{a}.\overline{b}\}]$

Divisão: $\frac{[a]}{[b]} = [\min\{\frac{\underline{a}}{\underline{b}}, \frac{\underline{a}}{\overline{b}}, \frac{\overline{a}}{\underline{b}}, \frac{\overline{a}}{\overline{b}}\}; \max\{\frac{\underline{a}}{\underline{b}}, \frac{\underline{a}}{\overline{b}}, \frac{\overline{a}}{\underline{b}}, \frac{\overline{a}}{\overline{b}}\}]$

Operações sobre teoria de conjuntos também são definidas para intervalos. Considerando dois intervalos reais $[a] = [\underline{a}; \overline{a}]$ e $[b] = [\underline{b}; \overline{b}]$, operações básicas como união ($\cup$) e intersecção ($\cap$) podem ser definidas como:

Intersecção: $[a] \cap [b] = [\max\{\underline{a}, \underline{b}\}; \min\{\overline{a}, \overline{b}\}]$

União: $[a] \cup [b] = [\min\{\underline{a}, \underline{b}\}; \max\{\overline{a}, \overline{b}\}]$

Alguns exemplos sobre as definições e operações citadas anteriormente nessa seção são mostrados abaixo. Considere os intervalos $[a] = [6; 9]$ e $[b] = [3; 9]$.

$w([a]) = 3$

$mid([b]) = 6$

$[a] + [b] = [9; 18]$

$[a] - [b] = [-3; 6]$

$[a] * [b] = [18; 81]$

$[a]/[b] = [0,66; 3]$

$[a] \cap [b] = [6; 9]$

$[a] \cup [b] = [3; 9]$

Além dessas operações básicas, intervalos também podem estar presentes em operações mais complexas, como funções. As definições associadas a funções intervalares podem ser vistas na Seção 2.3

2.2 Vetor Intervalar

Um vetor intervalar real, também chamado de caixa, é um subconjunto de $\mathbb{R}^n$ que pode ser definido como o produto cartesiano de intervalos. Jaulin et al. (2001) mostra algumas definições referentes a caixas, por exemplo, uma caixa n-dimensional pode ser definida como:

$$[\mathbf{x}] = [x_1] \times [x_2] \times \ldots \times [x_n]$$

Onde $[x_i] = [\underline{x}_i; \overline{x}_i]$ para $i = 1, \ldots, n$ e n representa o número de dimensões da caixa

Cada um dos componentes intervalares $[x_i]$ do vetor são projeções de $[\mathbf{x}]$ no i-ésimo eixo cartesiano. A Figura 2.1 representa uma caixa com $n = 2$.

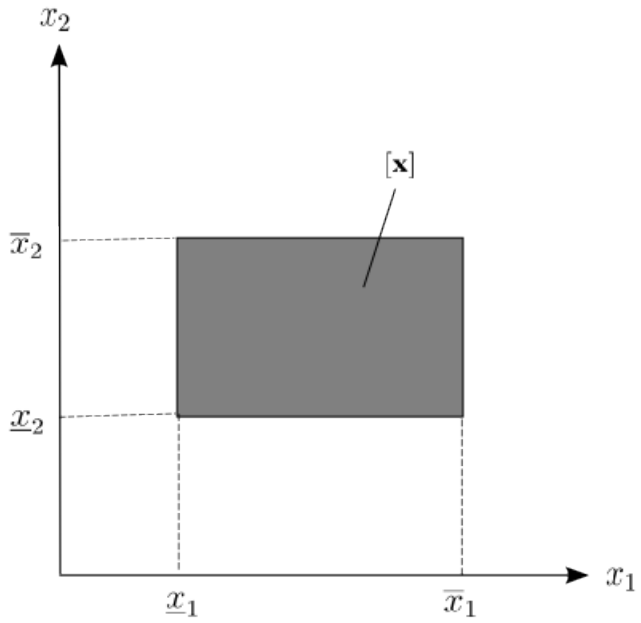


Figura 2.1: Exemplo de caixa, $[\mathbf{x}]$ de $\mathbb{IR}^2$. Imagem extraída de (JAULIN et al., 2001)

O tamanho de uma caixa $w([\mathbf{x}])$ é definido pelo tamanho do maior intervalo que compõe o vetor, ou seja:

$$w([\mathbf{x}]) = \max_{1 \leq i \leq n} w([x_i])$$

Assim como um intervalo, uma caixa possui ponto médio $mid([\mathbf{x}])$, esse valor é representado por um vetor definido como:

$$mid([\mathbf{x}]) = (mid([x_1]), ..., mid([x_n]))$$

Operações básicas também podem ser definidas para caixas, por exemplo, considerando as caixas $[\mathbf{a}]$ e $[\mathbf{b}]$:

$$[\mathbf{a}] * [\mathbf{b}] = [a_1] * [b_1] \times ... \times [a_n] * [b_n]$$
$$[\mathbf{a}] + [\mathbf{b}] = [a_1] + [b_1] \times ... \times [a_n] + [b_n]$$

Nas operações entre caixas, cada dimensão da caixa resultante é obtida a partir de operações feitas nos intervalos referentes àquela dimensão das caixas iniciais.

Uma vez que o tamanho de uma caixa seja conhecido, pode-se definir seu plano principal. O plano principal de uma caixa corresponde ao plano perpendicular ao maior lado da caixa. Alguns algoritmos de bissecção utilizam o plano principal como referência para a bissecção de caixas. A Figura 2.2 mostra todos os planos de uma caixa com três dimensões, onde o plano principal está destacado na cor vermelha.

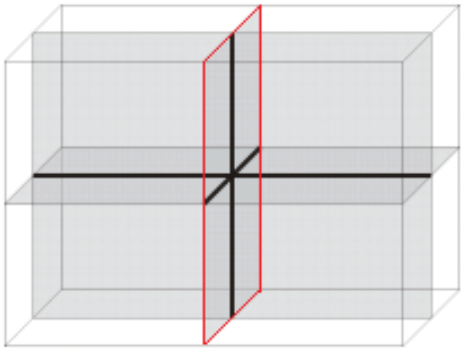


Figura 2.2: Planos de uma caixa 3D. Imagem baseada em (JAULIN et al., 2001)

Podem existir casos em que multiplas caixas são usadas durante uma computação, um operador bastante útil nesses casos é o *interval hull*. Dado um conjunto $\mathbb{A} \in \mathbb{R}^n$ o *interval hull* desse conjunto $hull(\mathbb{A})$ é dado pela menor caixa que contenha $\mathbb{A}$ como ilustra a Figura 2.3.

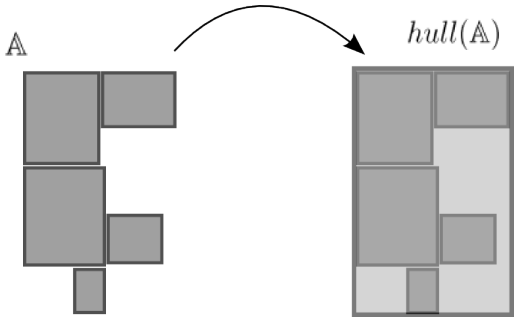


Figura 2.3: Exemplo de $hull(\mathbb{A})$.

2.3 Funções Intervalares

Além das definições apresentadas até o momento, a análise de intervalos também permite cálculos mais complexos, tais como o cálculo de funções. A imagem intervalar $\mathcal{I}$ de uma função real $f([x])$ pode ser definida como:

$$\mathcal{I}(f,[x]) = [\min\{f([x]) \mid x \in [x]\}, \max\{f([x]) \mid x \in [x]\}]$$

Qualquer função $f : \mathbb{R} \to \mathbb{R}$ tradicional, composta por operadores aritméticos e funções elementares, pode ser definida como uma função de inclusão. Uma função intervalar

$[f] : \mathbb{IR} \rightarrow \mathbb{IR}$ pode ser considerada uma função de inclusão quando respeita a propriedade (KUEVIAKOE; LAMBERT; TARROUX, 2012), (JAULIN et al., 2001):

$$f([\mathbf{x}]) \subset [f]([\mathbf{x}])$$

Como exemplo, considerando a função $f(x) = x^2 + 2x + 4$, pode-se definir como função de inclusão $[f]([x]) = [x]^2 + 2[x] + 4$. Para $[x] = [-3; 4]$, $[f]([x])$ pode ser resolvida seguindo os passos apresentados na Equação 2.1.

$$\begin{aligned}
[f]([x]) &= [x]^2 + 2[x] + 4 \\
&= [-3; 4]^2 + 2[-3; 4] + 4 \\
&= [0; 16] + 2[-3; 4] + 4 \\
&= [0; 16] + [-6; 8] + 4 \\
&= [-6; 24] + 4 \\
&= [-2; 28]
\end{aligned} \tag{2.1}$$

Para resolver $f([x])$ é necessário resolver a equação para todos os elementos do intervalo. Assim, a Figura 2.4 apresenta os resultados.

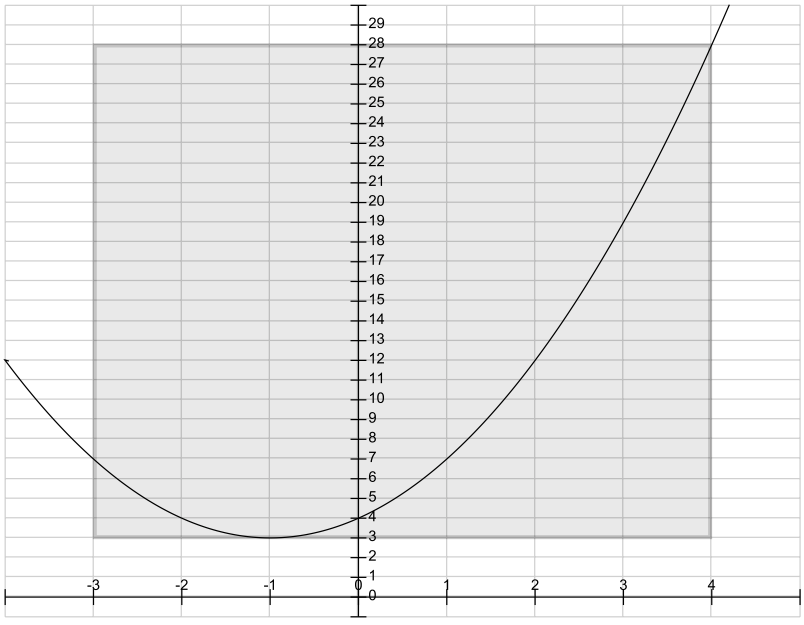


Figura 2.4: Representação gráfica da equação $f([-3; 4]) = x^2 + 2x + 4$

$f([x]) = [3; 28]$ é subconjunto de $[f]([x]) = [-2; 28]$.

A imagem de uma função f não tem um formato predeterminado, assim o resultado de uma função de inclusão $[f]$ precisa ser suficiente para abranger essa imagem. A Figura 2.5 mostra a imagem de uma função sendo representada através de caixas resultantes de funções de inclusão. Quando o resultado de uma função de inclusão é a menor caixa possível que abrange a imagem de f, essa função é chamada minimal. Na Figura 2.5 a

função minimal é representada por $[f]^*([\mathbf{x}])$. Nem sempre uma função de inclusão será minimal, como por exemplo, a função $[f]([\mathbf{x}])$ representada na Figura 2.5 (JAULIN et al., 2001).

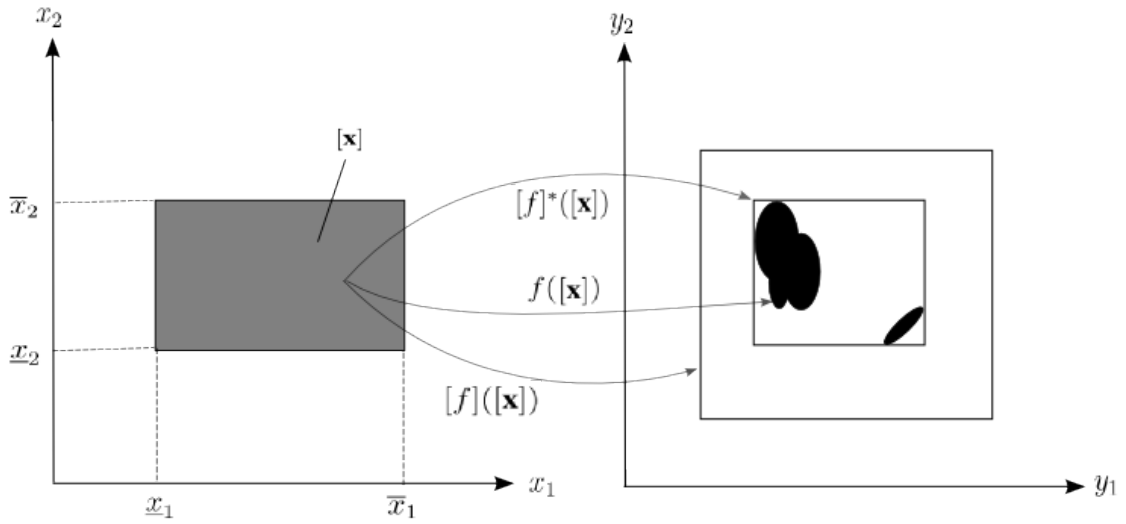


Figura 2.5: Funções de Inclusão. Imagem baseada em (JAULIN et al., 2001)

2.4 Funções de Teste de Inclusão

Funções de teste de inclusão servem, por exemplo, para provar que todos os pontos representados por um dado intervalo ou caixa satisfazem ou não uma certa propriedade (JAULIN et al., 2001). Uma função $[t]$, onde $[t] : \mathbb{IR}^n \rightarrow \mathbb{IB}$, é considerada um teste de inclusão se:

$$[t]([\mathbf{x}]) = 1 \Rightarrow (\forall \mathbf{x} \in [\mathbf{x}], t(\mathbf{x}) = 1)$$
$$[t]([\mathbf{x}]) = 0 \Rightarrow (\forall \mathbf{x} \in [\mathbf{x}], t(\mathbf{x}) = 0)$$

Considere, por exemplo, a função de teste apresentada na Equação 2.2.

$$t(\mathbf{x}) = \begin{cases} 1, & \text{se} \quad x_1 + x_2 \leq 5, \\ 0, & \text{se} \quad x_1 + x_2 > 5. \end{cases} \tag{2.2}$$

O teste de inclusão para tal função (Equação 2.2) é apresentado na Equação 2.3. Onde x_1 e x_2 são agora considerados intervalos reais $[x_1]$ e $[x_2]$.

$$[t]([\mathbf{x}]) = \begin{cases} [1;1], & \text{se} \quad \overline{x}_1 + \overline{x}_2 \leq [5;5], \\ [0;0], & \text{se} \quad \underline{x}_1 + \underline{x}_2 > [5;5], \\ [0,1], & \text{caso contrário.} \end{cases} \tag{2.3}$$

Uma função de teste de inclusão além de assumir valores de verdadeiro ou falso, pode assumir um estado indefinido. Este terceiro estado é representado pelo intervalo $[0;1]$ e acontece quando o intervalo testado possui intersecção, porém não está inteiramente contido no intervalo ao qual está sendo comparado.

2.5 Subpavings

Nas seções anteriores foram descritas formas de representar informações através de intervalos e caixas. Entretanto, esta forma de representação nem sempre é suficiente. Um exemplo disso é a representação da união de subconjuntos desconexos (JAULIN et al., 2001). Dadas as representações de informação citadas até o momento, os subconjuntos seriam representados através de uma única caixa. Caso os conjuntos estejam distantes um do outro, a caixa que representa esses subconjuntos poderá ter um tamanho grande e pouco significativo, além de conter informações que certamente não pertencem ao resultado buscado. Isso pode ser visto na Figura 2.6, onde os subconjuntos (em preto) são representados por uma caixa que engloba toda a área em cinza, ou seja, a informação relevante é apenas uma parte desta caixa.

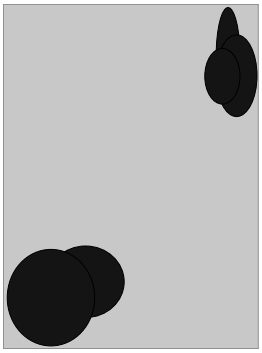


Figura 2.6: Representação de subconjuntos desconexos através de uma única caixa.

Quando o resultado esperado exige maior precisão, a utilização de uma caixa para representá-lo pode não prover informação suficiente. Neste contexto, é possível usar outra forma de representação intervalar, chamada *subpaving*. Uma *subpaving* (em tradução livre: calçamento, ladrilhos) é formada por um conjunto de caixas não sobrepostas, que juntas representam um determinado conjunto solução. Quando cada uma das caixas de uma *subpaving* é obtida através de um número finito de bissecções e seleções, a *subpaving* é chamada regular. *Subpavings* regulares também são chamadas *n-trees* (JAULIN et al., 2001).

Apesar da utilização de várias caixas, *subpavings* proporcionam maior precisão sobre a localização da informação relevante no espaço de busca quando comparadas com a caixa resultante de funções de inclusão. A Figura 2.7 mostra uma solução através de *subpaving*, destacando o conjunto buscado, em cinza escuro, e as caixas que compõem a *subpaving*, em cinza claro (KIEFFER; JAULIN; WALTER, 2002). A partir do espaço de busca $[0;8] \times [0;8]$ a lista $\mathcal{L}$ de caixas que compõem a *subpaving* pode ser definida como:

$$\mathcal{L} = \{[0;2] \times [0;4]; [4;6] \times [2;4]; [4;6] \times [4;8]\}$$

Subpavings podem ser usadas para representar conjuntos, a Figura 2.8 mostra a representação do conjunto $\mathbb{X}$ através de duas *subpavings* $\underline{\mathbb{X}}$ e $\overline{\mathbb{X}}$, onde $\mathbb{X}$ é representado por $\underline{\mathbb{X}} \cup \overline{\mathbb{X}}$.

Considerando por exemplo, o conjunto dado por $\mathbb{X} = \{(x,y)|(x-1)^2 + (y-1)^2 \in [2;4]\}$ e um espaço de busca inicial $[\mathbf{x}] = [-3;3] \times [-3;3]$, $\mathbb{X}$ é representado na Figura

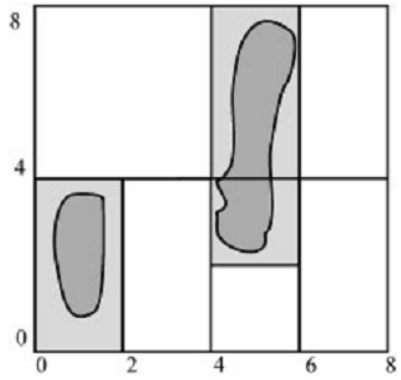


Figura 2.7: Representação de subconjuntos desconexos através de *subpavings*. Extraída de (KIEFFER; JAULIN; WALTER, 2002)

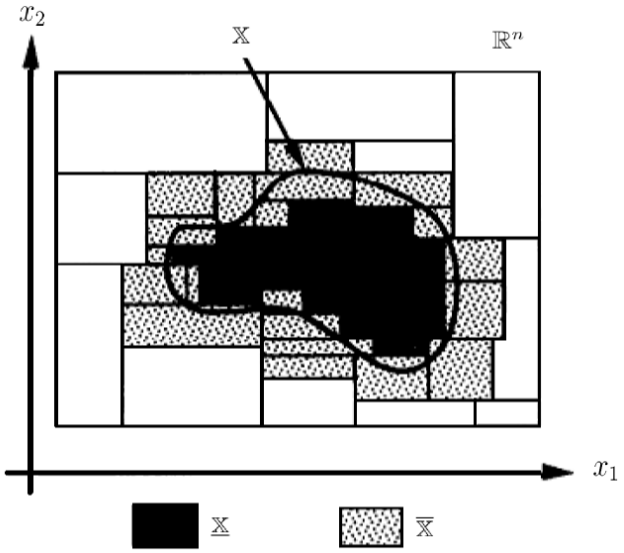


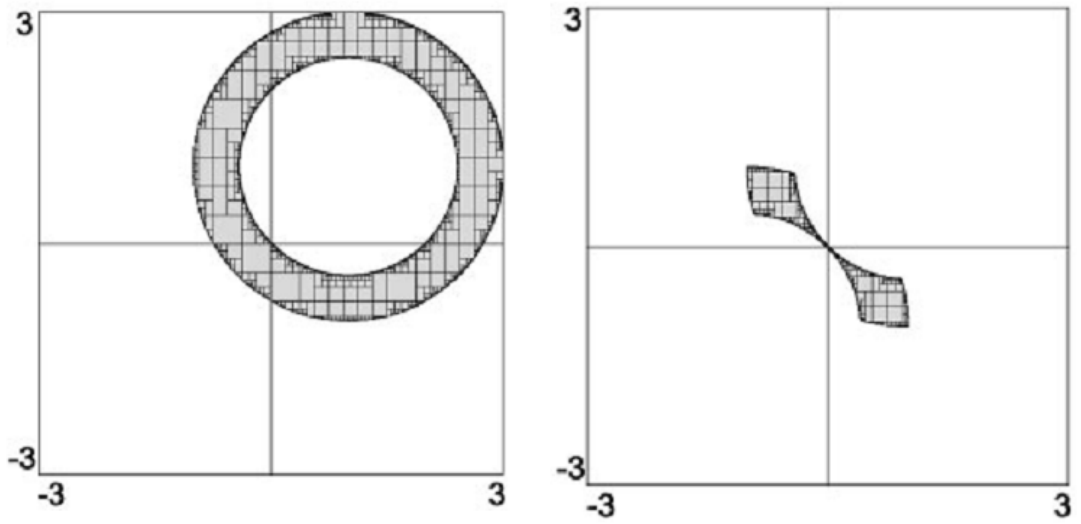
Figura 2.8: Representação de conjuntos através de *subpavings*. Extraída de (JAULIN; WALTER, 1993)

2.9(a), e pode ser definido como:

$$\underline{\mathbb{X}} \subset \mathbb{X} \subset \overline{\mathbb{X}}$$

Um segundo exemplo pode ser visto na Figura 2.9(b), considerando o conjunto $\mathbb{X} = \{(x,y)|(((x-1)^2+(y-1)^2) \cap ((x+1)^2+(y+1)^2)) \in [2;4]\}$ e o espaço de busca inicial $[\mathbf{x}] = [-3;3] \times [-3;3]$. Onde $\underline{\mathbb{X}}$ é representado em cinza claro e $\overline{\mathbb{X}}$ é representado em cinza escuro (caixas pequenas localizadas nas bordas da imagem em cinza claro).

Para encontrar as *subpavings* que definem um conjunto $\mathbb{X}$ são utilizados algoritmos de bissecção e classificação de caixas. Um desses algoritmos é o *Set Inversion Via Interval Analysis* (SIVIA) (JAULIN; WALTER, 1993), mostrado com mais detalhes na Seção 2.5.1



(a) Resultados do primeiro exemplo. (b) Resultados do segundo exemplo.

Figura 2.9: *Subpavings* obtidas com o algoritmo SIVIA. Extraída de (KIEFFER; JAULIN; WALTER, 2002)

2.5.1 SIVIA

O algoritmo SIVIA (*Set Inversion Via Interval Analysis*) apresentado por Jaulin e Walter (1993) é utilizado para resolver o problema de inversão de conjuntos. O qual pode ser definido como:

$$\mathbb{X} = \{\mathbf{x} \in \mathbb{R}^n | f(\mathbf{x}) \in \mathbb{Y}\} = f^{-1}(\mathbb{Y})$$

Ou seja, deseja-se definir $\mathbb{X}$ dado uma função f e sua imagem $\mathbb{Y}$. Para o funcionamento do algoritmo SIVIA, uma caixa $[\mathbf{x}]$ é requerida como espaço de busca. O algoritmo, apresentado em pseudo-código no Algoritmo 1, pode ser resumido em quatro passos:

1º. Caso $f([\mathbf{x}])$ não tenha interseção com $\mathbb{Y}$, $[\mathbf{x}]$ é descartada. Algoritmo 1 linhas 1 - 3.

2º. Caso $f([\mathbf{x}])$ esteja contida em $\mathbb{Y}$, $[\mathbf{x}]$ será considerada parte da solução. Algoritmo 1 linhas 4 - 8.

3º. Caso a caixa $[\mathbf{x}]$ seja pequena, ou seja $w([\mathbf{x}])$ é menor que o limite predefinido pelo usuário, representado pelo parâmetro ε, e ainda considerada indeterminada, a caixa é aceita como parte da solução. Algoritmo 1 linhas 9 - 12.

4º. Caso $f([\mathbf{x}])$ tenha interseção, mas não está contida em $\mathbb{Y}$, $[\mathbf{x}]$ é considerada indeterminada. Se $w([\mathbf{x}])$ for maior que um limite predefinido a caixa é bisseccionada e as novas caixas serão testadas. Algoritmo 1 linhas 13 - 14.

A principal desvantagem apontada para o método SIVIA é o custo computacional, que cresce exponencialmente de acordo com o número de dimensões (JAULIN; WALTER, 1993).

Algoritmo 1: SIVIA

Data: $f, \mathbb{Y}, [\mathbf{x}], \varepsilon$
Result: $\underline{\mathbb{X}}, \overline{\mathbb{X}}$
if $[f]([\mathbf{x}]) \cap \mathbb{Y} = \emptyset$ **then**
  return
end
if $[f]([\mathbf{x}]) \subset \mathbb{Y}$ **then**
  $\underline{\mathbb{X}} = \underline{\mathbb{X}} \cup [\mathbf{x}]$;
  $\overline{\mathbb{X}} = \overline{\mathbb{X}} \cup [\mathbf{x}]$;
  return;
end
if $w([\mathbf{x}]) < \varepsilon$ **then**
  $\overline{\mathbb{X}} = \overline{\mathbb{X}} \cup [\mathbf{x}]$;
  return;
end
$SIVIA(f, \mathbb{Y}, L[\mathbf{x}], \varepsilon, \underline{\mathbb{X}}, \overline{\mathbb{X}})$;
$SIVIA(f, \mathbb{Y}, R[\mathbf{x}], \varepsilon, \underline{\mathbb{X}}, \overline{\mathbb{X}})$;

Através da bissecção e seleção de caixas o algoritmo cria um conjunto de caixas não sobrepostas que representam a solução para o problema. O resultado apresentado na Figura 2.9 foi obtido com o algoritmo SIVIA.

Nesta seção vimos que algoritmos baseados em bissecção, como o SIVIA, representam o resultado através de um conjunto de caixas (*subpavings*). Essa representação é mais precisa que o uso de uma única caixa, entretanto o custo computacional é maior. Uma opção é manter o uso de apenas uma caixa e tentar reduzi-la tanto quanto possível. Para tanto, pode-se aplicar outra técnica intervalar chamada contratores, a qual discutiremos a seguir.

2.6 Contratores

Contratores são operadores utilizados para contrair domínios de acordo com restrições impostas, com a garantia de que nenhum valor factível será descartado do domínio. Essa técnica não executa bissecções de domínios, conseguindo assim manter uma complexidade polinomial de tempo e espaço (JAULIN et al., 2001). Um operador $\mathcal{C}$ é um contrator se, dada uma restrição c e um domínio $[\mathbf{x}]$ ele obedece as seguintes propriedades (JAULIN, 2011):

- Completude (*completeness*): $(c \cap [\mathbf{x}]) \subset \mathcal{C}([\mathbf{x}])$ Todo elemento do intervalo $[\mathbf{x}]$ que atende a restrição c está contido no resultado do operador $\mathcal{C}([\mathbf{x}])$, ou seja, não ocorre perda de soluções factíveis.

- Contração/Encolhimento (*contractance*): $\mathcal{C}([\mathbf{x}]) \subset [\mathbf{x}]$ A caixa resultante da operação de contração $\mathcal{C}([\mathbf{x}])$ está contida no domínio inicial $[\mathbf{x}]$.

Contratores podem ser usados a fim de aumentar a precisão do resultado de funções de inclusão, dado que uma função de inclusão nem sempre é minimal. Isso pode ser útil para lidar com espaços altamente dimensionais, especialmente por ter complexidade polinomial (JAULIN et al., 2001).

Contratores são utilizados para reduzir domínios a partir de um conjunto de restrições. Neste cenário, um problema que se enquadra nessas características é o problema de satisfação de restrições (CSP). É possível representar diversos problemas como um problema de satisfação de restrições, dentre eles, problemas das áreas de controle e robótica (JAULIN, 2011). Um CSP é composto por:

- Um conjunto de variáveis $\mathcal{V} = x_1, ...x_n$
- Um conjunto de equações (restrições) $C = c_1, ...c_n$
- Um conjunto de domínios aos quais as variáveis estão associadas. A partir de uma análise intervalar, consideramos domínios intervalares $[x_1], ..., [x_n]$.

Um CSP é resolvido quando um estado que satisfaça as restrições do sistema de equações é encontrado. Para tanto, é usado propagação de restrições. A propagação de restrições consiste em fazer reduções nos domínios utilizando as restrições definidas no problema. As reduções são feitas até que nenhum domínio possa ser contraído (KUEVIAKOE; LAMBERT; TARROUX, 2012).

Existem diferentes tipos de contratores, um deles é o contrator *forward-backward* $\mathcal{C}_{\downarrow\uparrow}$, o qual se baseia em propagação de restrições. Os contratores *forward-backward* são compostos por dois passos (WANG et al., 2013):

1. *Forward*: Contrai y usando $[y] \cap [f]([x])$
2. *Backward*: Contrai x usando $[x] \cap [f^{-1}]([y])$

Considerando que uma restrição pode ser escrita $y = f(x)$ ou em sua forma inversa $x = f^{-1}(y)$. Esses passos se repetem até que não sejam feitas mais contrações significantes. Para exemplificar um processo de contração, considere a equação $x_3 = x_1 + x_2$, onde $[x_1] = [-\infty; 5]$, $[x_2] = [-\infty; 4]$ e $[x_3] = [6; \infty]$.

$$x_3 = x_1 + x_2 \Rightarrow x_3 \in [6; \infty] \cap ([-\infty; 5] + [-\infty; 4]) = [6; \infty] \cap [-\infty; 9] = [6; 9]$$

$$x_1 = x_3 - x_2 \Rightarrow x_1 \in [-\infty; 5] \cap ([6; \infty] - [-\infty; 4]) = [-\infty; 5] \cap [2; \infty] = [2; 5]$$

$$x_2 = x_3 - x_1 \Rightarrow x_2 \in [-\infty; 4] \cap ([6; \infty] - [-\infty; 5]) = [-\infty; 4] \cap [1; \infty] = [1; 4]$$

Apenas reescrevendo a equação e utilizando o operador de intersecção ($\cap$) pode-se perceber uma contração significativa, os novos valores de domínio são:

$$[x_1] = [2; 5]$$

$$[x_2] = [1; 4]$$

$$[x_3] = [6; 9].$$

Dentre os contratores que utilizam a estratégia *forward-backward* pode-se citar o HC4. O método usa uma árvore binária para representar as restrições, onde nas folhas estão as constantes ou variáveis e nos nós são representados os símbolos das operações elementares (BENHAMOU et al., 1999). O HC4 possui dois passos de execução (KUEVIAKOE; LAMBERT; TARROUX, 2012):

1. *Forward*: a árvore é percorrida das folhas até a raiz resolvendo as expressões de cada nó.
2. *Backward*: a árvore é percorrida da raiz até as folhas aplicando operadores de contração para eliminar valores inconsistentes.

Dada a restrição $2x = z - y^2$ e os domínios $x = [0; 20]$, $y = [-10; 10]$ e $z = [0; 16]$ a árvore usada pelo contrator HC4 é apresentada nas Figuras 2.10 e 2.11. A Figura 2.10 apresenta os cálculos executados a partir das folhas da árvore, por exemplo, a operação $(*(2, x))$ resulta no intervalo $[0; 40]$. A Figura 2.11 representa a segunda etapa do processo, o qual é executado a partir da raiz. O primeiro cálculo feito nesta segunda etapa é referente à $[0; 40] \cap [-100; 16]$, essa intersecção corresponde à igualdade entre os intervalos originados respectivamente de $(*(2, x))$ e $(-(z, \hat{}(y, 2)))$.

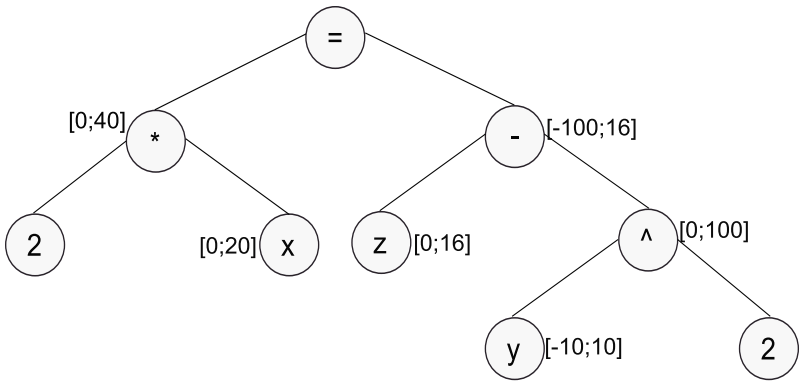


Figura 2.10: HC4 - *Forward*. Baseada em (BENHAMOU et al., 1999)

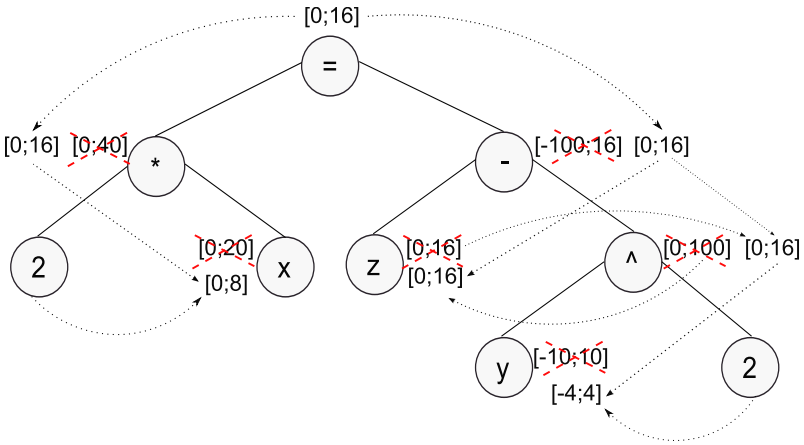


Figura 2.11: HC4 - *Backward*. Baseada em (BENHAMOU et al., 1999)

Kueviakoe, Lambert e Tarroux (2012) fazem uma comparação entre diferentes contratores para resolução de CSP. A partir dos resultados encontrados os autores indicam

o contrator HC4 para localização de veículos, sendo que, dentre os algoritmos testados, o HC4 apresentou o menor tempo de execução. A principal limitação do método é a sensibilidade para casos de múltiplas ocorrências de uma variável. Um exemplo dessa limitação é: considere a restrição $x \times x + y^2 = 2$, o HC4 não consegue reduzir a caixa inicial. Entretanto se a restrição for reescrita para a forma $x^2 + y^2 = 2$ o contrator não terá problemas para reduzir a caixa inicial.

Quando comparado com o SIVIA, contratores apresentam uma desvantagem em relação a precisão dos resultados. O uso de contratores garante a inclusão do resultado correto na solução final, entretanto a solução pode ser dada como um intervalo maior que o esperado ou aceitável. O uso de uma caixa para a representação de conjuntos desconexos distantes no espaço de busca é um exemplo de casos onde o uso de contratores pode retornar resultados pouco significativos.

3 PROBLEMA DE AUTO LOCALIZAÇÃO

O problema de localização de robôs móveis consiste em definir, dado um mapa do ambiente, qual a posição do robô. Esse é um problema clássico da robótica que foi e ainda é frequentemente foco de pesquisas, visto que a maioria das tarefas dependem do conhecimento sobre a posição do robô e dos objetos manipulados no ambiente. A localização do robô se dá através da correspondência entre as leituras dos sensores do robô e o mapa do ambiente. Apesar da simples descrição do problema, a dificuldade inerente à ele está em inferir posições no ambiente somente a partir das medições de sensores. Para estimar a localização do robô não é suficiente uma medição, sendo necessário relacionar as informações extraídas ao longo do tempo (THRUN et al., 2005).

A Figura 3.1 apresenta o modelo gráfico do problema de localização para as abordagens probabilísticas e intervalares apresentadas neste trabalho. As leituras z dos sensores e as ações u tomadas pelo robô são ações do sistema, enquanto que a posição x do robô é o estado a ser estimado. A modelagem do problema de localização é baseada na suposição de Markov ou suposição de estado completo. Ela diz que é possível estimar o estado atual x_t, apenas conhecendo a última ação u_t e as últimas observações z_t, desde que se conheça o estado anterior x_{t-1}. Todas as informações passadas não são mais necessárias, uma vez que já foram consideradas no processo de estimativa dos estados anteriores.

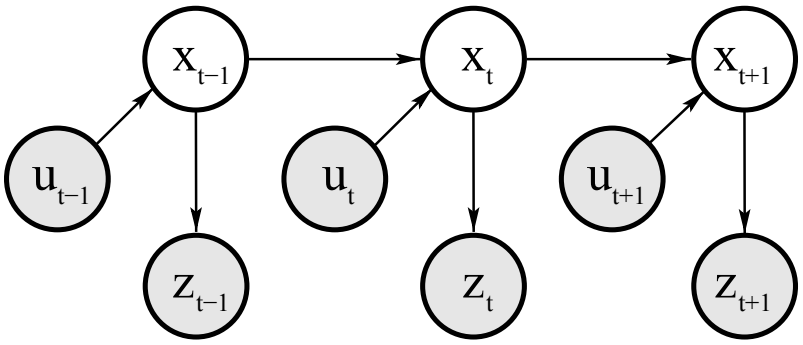


Figura 3.1: Modelo Gráfico de Localização. Extraída de (THRUN et al., 2005)

Um exemplo do processo de localização de um robô capaz de detectar marcadores no ambiente é mostrado na Figura 3.2. A cada instante t de tempo, é feita uma predição da posição do robô com base na sua posição anterior x_{t-1} e na sua última ação u_t. Essa predição é feita através de um modelo de movimentação baseado em odômetros ou sensores de velocidade, etc, que em geral são bastante imprecisos. Após este passo, a estimativa sobre a posição do robô é corrigida usando-se as medições dos sensores z_t. Nesta etapa

busca-se ajustar a estimativa de posição do robô para uma posição onde ele seja capaz de observar aquilo que foi medido.

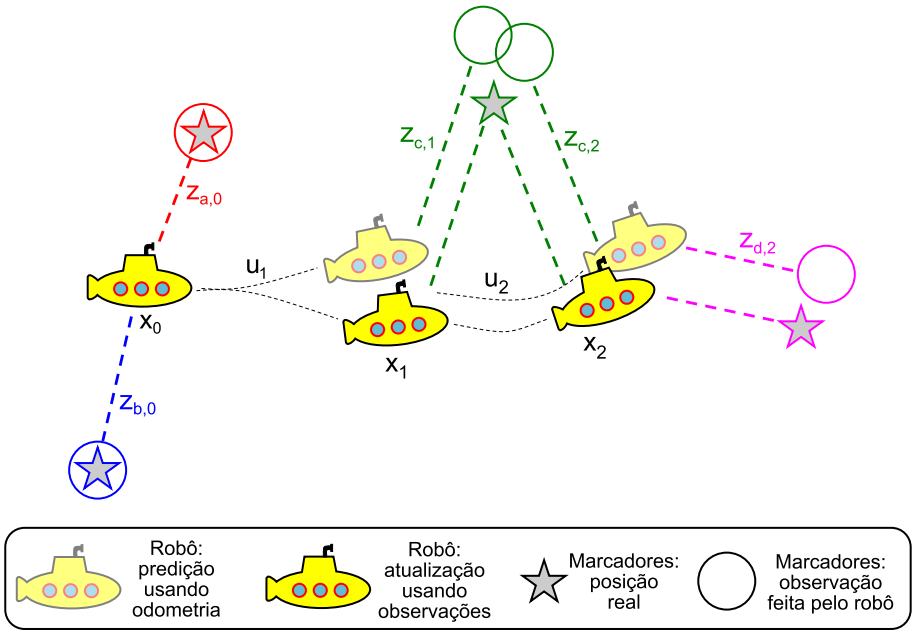


Figura 3.2: Exemplo do processo de Localização.

Problemas de localização podem ser classificados de acordo com vários critérios. Thrun et al. (2005) abordam a questão de taxionomia dos problemas de localização classificando-os conforme quatro dimensões, as quais abrangem as principais características do problema. São elas:

- Tipo de conhecimento disponível
  - Rastreamento de posição

    Para este tipo de problema assume-se que inicialmente o robô conhece sua posição, e a incerteza sobre a sua localização se dá através do ruído gerado durante a movimentação do robô. Este é um problema considerado local, pois a incerteza é local, ou seja, restrita a região próxima ao robô.
  - Localização global

    Neste caso a posição inicial do robô não é conhecida, o robô é colocado em qualquer parte do ambiente e terá pouca ou nenhuma informação sobre a sua localização. Uma vez que não se tem conhecimento da localização do robô, não se pode gerar uma região de incerteza, assim a incerteza está inicialmente distribuída por todo o ambiente. Este problema é mais complexo para resolver do que o anterior.
  - Problema do robô raptado

    Nesta variação do problema de localização, além da necessidade da localização global, o robô pode ser raptado e recolocado em um local diferente no ambiente durante a execução do processo. Isso torna o problema ainda mais difícil, já que o robô pode não perceber o rapto e continuar com o processo de localização como se nada houvesse acontecido. O fato de o robô desconhecer

sua posição é ruim, entretanto, considerar como verdadeiro um conhecimento incorreto é ainda pior.

- Variação do ambiente: estático versus dinâmico
  - Ambientes estáticos

    Ambientes estáticos são ambientes que não sofrem modificações nas posições dos objetos (obstáculos). Neste tipo de ambiente a única posição que se modifica é a do próprio robô.
  - Ambientes dinâmicos

    Ambientes dinâmicos são aqueles cujos obstáculos mudam a sua localização ou configuração no decorrer do tempo. O principal interesse neste tipo de ambiente são as mudanças persistentes, que afetam a leitura dos sensores. O robô precisa extrair informações relevantes do conjunto de dados capturados pelos sensores e descartar dados que não contribuem para o processo de localização, como por exemplo, mobílias ou pessoas. Claramente, o processo de localização em ambientes dinâmicos é mais difícil que em ambientes estáticos.
- Controle de movimentos do robô
  - Abordagens passivas

    Em abordagens passivas o robô é controlado de forma independente do processo de localização. Sendo assim, não é possível movimentar o robô de forma que facilite a localização.
  - Abordagens ativas

    Algoritmos que usam abordagem ativa, controlam os movimentos do robô de forma a minimizar erros de localização.
- Número de robôs envolvidos no processo
  - Um robô

    Neste tipo de abordagem, todos os dados coletados para o processo de localização são provenientes de um único robô.
  - Multi robôs

    Uma segunda opção pode ser adotada quando os robôs envolvidos no processo são capazes de detectar uns aos outros, dessa forma é possível usar a informação sobre a localização relativa dos robôs observados para estimar a localização de cada robô. Nesse tipo de problema ainda existe um acréscimo na complexidade dada a necessidade de tratar questões de comunicação entre robôs.

O problema abordado por essa pesquisa é o de auto localização global em ambientes estáticos, tratado através de uma abordagem passiva utilizando um único robô. O ambiente alvo é subaquático, isso incrementa a dificuldade em relação aos sensores utilizados, por exemplo, o GPS é um sensor bastante utilizado em aplicações terrestres, entretanto não está disponível para ambientes subaquáticos.

A utilização de marcadores, como *beacons* ativos ou passivos é considerada uma abordagem elegante para o tratamento de problemas de localização. Mesclando informações de sensores internos e externos o robô é capaz de se localizar mais precisamente. Esse tipo de técnica para extração de informações é semelhante a usada pelo homem, que se orienta através de estrelas ou montanhas por exemplo (SIEGWART; NOURBAKHSH, 2004).

Nesta pesquisa são usados marcadores. Esses marcadores são distinguíveis e suas posições são conhecidas pelo robô. Através desses marcadores e de sensores internos do robô é possível executar o processo de localização.

O problema utilizado como exemplo e tratado como caso de teste nesta pesquisa envolve as seguintes informações:

- Marcadores m com localização conhecida. Esses marcadores emitem um sinal único, o qual o robô é capaz de identificar e utilizar para calcular a sua distância até eles. A localização dos marcadores no instante t deve ser conhecida para calcular a localização do robô no instante t, entretanto nada impede que a localização dos marcadores varie no decorrer do tempo.
- Espaço de busca corresponde ao ambiente, e pode ser modelado como uma caixa $[\mathbf{x}] = [x] \times [y] \times [z]$, onde $[x]$, $[y]$ e $[z]$ são intervalos.
- Informação de controle $u_{1:n}$ a cada instante de tempo t, onde $t = 0..n$, as quais incluem:
  - Orientação do robô $O(\phi_o, \theta_o, \psi_o)$, onde ϕ_o , θ_o e ψ_o refere-se respectivamente a orientação do robô nos eixos x, y e z.
  - Velocidade do robô $V(\phi_v, \theta_v, \psi_v)$, onde ϕ_v , θ_v e ψ_v refere-se respectivamente a velocidade do robô nos eixos x, y e z.
- Informação de medições $z_{0:n}$ a cada instante de tempo t, onde $t = 0..n$.
  - Distância z_t^j, referente à distância entre o robô e o marcador m^j no tempo t.

3.1 Auto Localização: Abordagem Probabilística

O uso de métodos probabilísticos para resolução de problemas da robótica data da década de 90. Antes disso, algumas técnicas baseavam-se em suposições não condizentes com a realidade, tais como modelos e percepções perfeitas, livres de incerteza. Através de leis da probabilidade, a robótica probabilística lida com modelos e sensoriamentos imperfeitos (THRUN, 2002).

Um conceito básico dos métodos probabilísticos é a crença. Uma crença é a representação do conhecimento do robô em relação ao estado do ambiente no qual está inserido. A representação de crenças, usadas em robótica probabilística, se dá normalmente através de distribuições de probabilidade. Uma distribuição de crença é uma probabilidade posterior, essa posterior é feita sobre uma variável de estado e é condicionada às informações disponíveis. Pode-se descrever a posterior do estado x_t como a Equação 3.1.

$$bel(x_t) = p(x_t|z_{1:t}, u_{1:t}) \tag{3.1}$$

Na Equação 3.1 a crença é calculada depois que a medida z_t é incorporada aos dados. Pode ser útil em algumas ocasiões, calcular a posterior antes que essa medida seja incorporada. Nesse caso a posterior seria denotada como mostra a Equação 3.2.

$$\overline{bel}(x_t) = p(x_t|z_{1:t-1}, u_{1:t}) \tag{3.2}$$

$\overline{bel}(x_t)$ é normalmente chamada de predição, quando tratada em um contexto de filtragem probabilística. E correção é o nome associado ao cálculo de $bel(x_t)$ a partir de $\overline{bel}(x_t)$.

Uma técnica genérica para o cálculo de crenças é o filtro de Bayes. O filtro de Bayes é um método recursivo para estimar a crença do estado do robô que funciona em duas etapas:

1º *Predição*: $\overline{bel}(x_t) = \int p(x_t|x_{t-1}, u_t)\ bel(x_{t-1})\ dx_{t-1}$
A primeira estimativa do estado atual do robô é feita pela integral do produto de duas distribuições: a probabilidade de transição do estado anterior x_{t-1} para o estado atual x_t dada a última ação u_t; e a crença sobre o estado anterior do robô $bel(x_{t-1})$.

2º *Correção*: $bel(x_t) = \eta\ \ p(z_t|x_t)\ \overline{bel}(x_t)$
Enquanto a predição do estado do robô é feita baseada na última ação u_t, a correção é baseada na última observação z_t. A crença sobre o estado do robô é corrigida utilizando-se a probabilidade condicional de observação das leituras z_t dado que o estado atual do robô é x_t. η é uma constante que garante a normalização do resultado.

A seguir a derivação do filtro de Bayes é apresentada (THRUN et al., 2005). Usando a regra de Bayes podemos reescrever $bel(x_t)$ da seguinte forma:

$$\begin{aligned} bel(x_t) = p(x_t|z_{1:t}, u_{1:t}) &= \frac{p(z_t|x_t, z_{1:t-1}, u_{1:t})\ p(x_t|z_{1:t-1}, u_{1:t})}{p(z_t|z_{1:t-1}, u_{1:t})} \\ &= \eta\ \ p(z_t|x_t, z_{1:t-1}, u_{1:t})\ p(x_t|z_{1:t-1}, u_{1:t}) \end{aligned}$$

Sabe-se que as observações z_t dependem apenas do estado que o robô se encontra no momento em que elas foram feitas, logo toda a informação referente as observações e ações anteriores podem ser descartadas no cálculo da probabilidade de z_t. Portanto, podemos simplificar

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

Logo, temos que

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

Mas sabemos que $p(x_t|z_{1:t-1}, u_{1:t}) = \overline{bel}(x_t)$, então podemos substituir para obtermos a equação final de correção:

$$bel(x_t) = \eta \;\; p(z_t|x_t)\; \overline{bel}(x_t) \tag{3.3}$$

Para obtermos a recursividade do filtro de Bayes, devemos condicionar a crença do estado x_t àquela obtida no instante anterior. Isto pode ser feito utilizando o Teorema da Probabilidade Total, que diz o seguinte:

$$p(x) = \int p(x|y)\; p(y)\; dy$$

Então, reescrevemos $\overline{bel}(x_t)$

$$\overline{bel}(x_t) = p(x_t|z_{1:t-1}, u_{1:t}) = \int p(x_t|x_{t-1}, z_{1:t-1}, u_{1:t})\; p(x_{t-1}|z_{1:t-1}, u_{1:t})\; dx_{t-1}$$

Assumindo estado completo, x_t depende apenas do estado anterior x_{t-1}, da última ação u_t e das últimas observações z_t. Logo, todas as outras informações podem ser descartadas do primeiro termo da integral

$$\overline{bel}(x_t) = \int p(x_t|x_{t-1}, u_t)\; p(x_{t-1}|z_{1:t-1}, u_{1:t})\; dx_{t-1}$$

Sabendo que x_{t-1} independe de u_t, reescrevemos o segundo termo da integral da seguinte forma:

$$\overline{bel}(x_t) = \int p(x_t|x_{t-1}, u_t)\; p(x_{t-1}|z_{1:t-1}, u_{1:t-1})\; dx_{t-1}$$

Porém $p(x_{t-1}|z_{1:t-1}, u_{1:t-1}) = bel(x_{t-1})$, então substituimos para obter a equação final de predição:

$$\overline{bel}(x_t) = \int p(x_t|x_{t-1}, u_t)\; bel(x_{t-1})\; dx_{t-1} \tag{3.4}$$

Filtros bayesianos podem ser divididos em duas categorias, uma delas inclui os métodos que dependem de uma decomposição do espaço de estados, onde cada parte é relacionada com uma probabilidade cumulativa. Outra categoria representa os métodos que aproximam o espaço de estados através de amostras, também chamadas de partículas. Abordagens de filtro de partículas se tornaram populares a partir dos anos 2000, devido principalmente à geração de implementações bastante simples para sistemas não lineares complexos (THRUN et al., 2005).

O filtro de partículas, que é a implementação não paramétrica do filtro de Bayes, apresentou sucesso primeiramente em tratar problemas de localização. Este método foi capaz de tratar problemas até então não solucionados, tais como localização global e problema do robô raptado (THRUN, 2002).

3.1.1 Auto Localização Global Utilizando Filtro de Partículas

Localização de Monte Carlo (MCL), comumente chamado de filtro de partículas foi proposto por Dellaert et al. (1999). Dentre as principais vantagens do método estão a capacidade de representação de distribuições multimodais, o aumento da precisão de acordo com os recursos computacionais disponíveis e a fácil implementação, além de não necessitar muitas suposições paramétricas (DELLAERT et al., 1999) (THRUN, 2002) (THRUN et al., 2005).

O filtro de partículas representa uma crença $bel(x)$ através de um conjunto de amostras. Essas amostras são chamadas de partículas e representam posições específicas $x(x, y, z)$ associadas a um fator de importância w. O algoritmo inicia espalhando M amostras usando o conhecimento inicial $bel(x_0)$ sobre as possíveis posições do robô. O fator de importância inicial de cada amostra é igual a $1/M$, isto é, todas as amostras são equiprováveis como candidatas a representar a posição correta do robô. A partir da crença $bel(x_{t-1})$ o filtro de partículas constrói a crença $bel(x_t)$ de forma recursiva (PRESTES; RITT; FUHR, 2008) (THRUN et al., 2005).

Dado um conjunto de medidas z_t e controles u_t capturados pelo robô no tempo t a crença sobre a localização do robô é calculada através de um conjunto de partículas (DELLAERT et al., 1999) (THRUN et al., 2005). Um conjunto com M partículas pode ser denotado por:

$$\chi_t = \{x_t^{[1]}, x_t^{[2]}, ..., x_t^{[M]}\} \tag{3.5}$$

Cada partícula x representa uma probabilidade posterior dada por:

$$p(x_t|u_t, z_t) \tag{3.6}$$

O filtro de partículas é composto basicamente pelos passos de predição, atualização de medição e reamostragem. O Algoritmo 2 mostra como o problema de localização é resolvido utilizando filtro de partículas. A Tabela 3.1 descreve as funções utilizadas no algoritmo.

Algoritmo 2: Localização usando Filtro de Partículas

Data: $map, u_{1:n}, z_{0:n}$
Result: $\mathbb{S}$

```
1 S = ∅
2 P = createParticle(m, map);
3 for t=0 : n do
4 |   move(P, u);
5 |   weight(P);
6 |   resampling(P);
7 |   S = S ∪ avgParticle(P);
8 end
9 return S
```

Quando o filtro de partículas é utilizado para resolver o problema de auto localização global, cada partícula representa um possível estado do robô em um determinado instante

Funções	O que faz
$createParticle(M, \mathbb{X})$	cria M partículas distribuídas aleatoriamente sobre a área representada por $\mathbb{X}$
$weight(\mathbb{P})$	atribui um fator de importância para cada partícula do conjunto $\mathbb{P}$ de acordo com a probabilidade $p(z\|x)$.
$resampling(\mathbb{P})$	cria um novo conjunto de partículas a partir do conjunto $\mathbb{P}$ usando o método da roleta
$move(\mathbb{P}, u)$	movimenta cada partícula do conjunto $\mathbb{P}$ de acordo com um modelo de movimento e os controles u
$avgParticle(\mathbb{P})$	retorna uma posição no ambiente baseada na média ponderada das partículas do conjunto $\mathbb{P}$

Tabela 3.1: Funções usadas na abordagem probabilística

de tempo. Na inicialização do filtro a incerteza sobre a localização do robô está distribuída por todo o ambiente (linha 2). Em consequência disso, quando a população de partículas é criada, partículas são distribuídas sobre todo o espaço de busca.

A cada iteração, as partículas são movidas (linha 4), classificadas (linha 5) e reamostradas (linha 6). Na etapa de movimentação, as partículas são movidas utilizando informações de orientação e velocidade, de acordo com os controles u que descrevem o movimento executado pelo robô, baseando-se na probabilidade descrita na Equação 3.7.

$$p(x_t|u_t, x_{t-1}) \tag{3.7}$$

Depois de movidas, as partículas são classificadas de acordo com um fator de importância associado a cada uma. Esse fator é determinado pela diferença entre as observações feitas pelo robô no ambiente e as observações feitas pela partícula no mapa. Quanto mais parecidas essas observações z são, maior é o valor associado à partícula, consequentemente, maior é a probabilidade de que a partícula represente a posição real do robô. O peso de cada partícula é descrito de acordo com a probabilidade apresentada na equação 3.8.

$$p(z_t|x_t) \tag{3.8}$$

Depois da classificação das partículas de acordo com os pesos, o conjunto de partículas passa pelo processo de reamostragem. Nesse processo, um dos algoritmos mais frequentemente utilizado é o algoritmo da roleta (o pseudo-código desse algoritmo é presentado no Algoritmo 5), no qual partículas com baixo fator de importância tem maior probabilidade de serem descartadas.

A partir de uma população de partículas, a localização do robô é normalmente considerada como a partícula de maior probabilidade, ou como a média ponderada de todas as partículas da população (linha 7). Entretanto, essa abordagem não provê garantias de que o resultado é correto, nem o quão perto está da real posição (LANGERWISCH; WAGNER, 2012).

A Figura 3.3 mostra um exemplo de funcionamento do algoritmo de localização usando filtro de partículas. Cada traço preto representa uma partícula, i.e. uma hipótese sobre a posição do robô, e a altura do traço representa o peso da partícula. Em (a), a distribuição das probabilidades é uniforme, correspondente à máxima incerteza, pois

o robô está inicializando o processo de localização e ainda não possui informações suficientes para melhor distribuir as probabilidades. Assim, as amostras são distribuídas aleatoriamente pelo espaço de busca. Já em (b), o robô obteve sua primeira leitura (o robô observou uma porta), logo as probabilidades são redefinidas e os pesos das amostras são calculados. As amostras localizadas em posições onde é possível visualizar uma porta recebem um peso maior, e terão mais probabilidade de sobrevivência no processo de reamostragem.

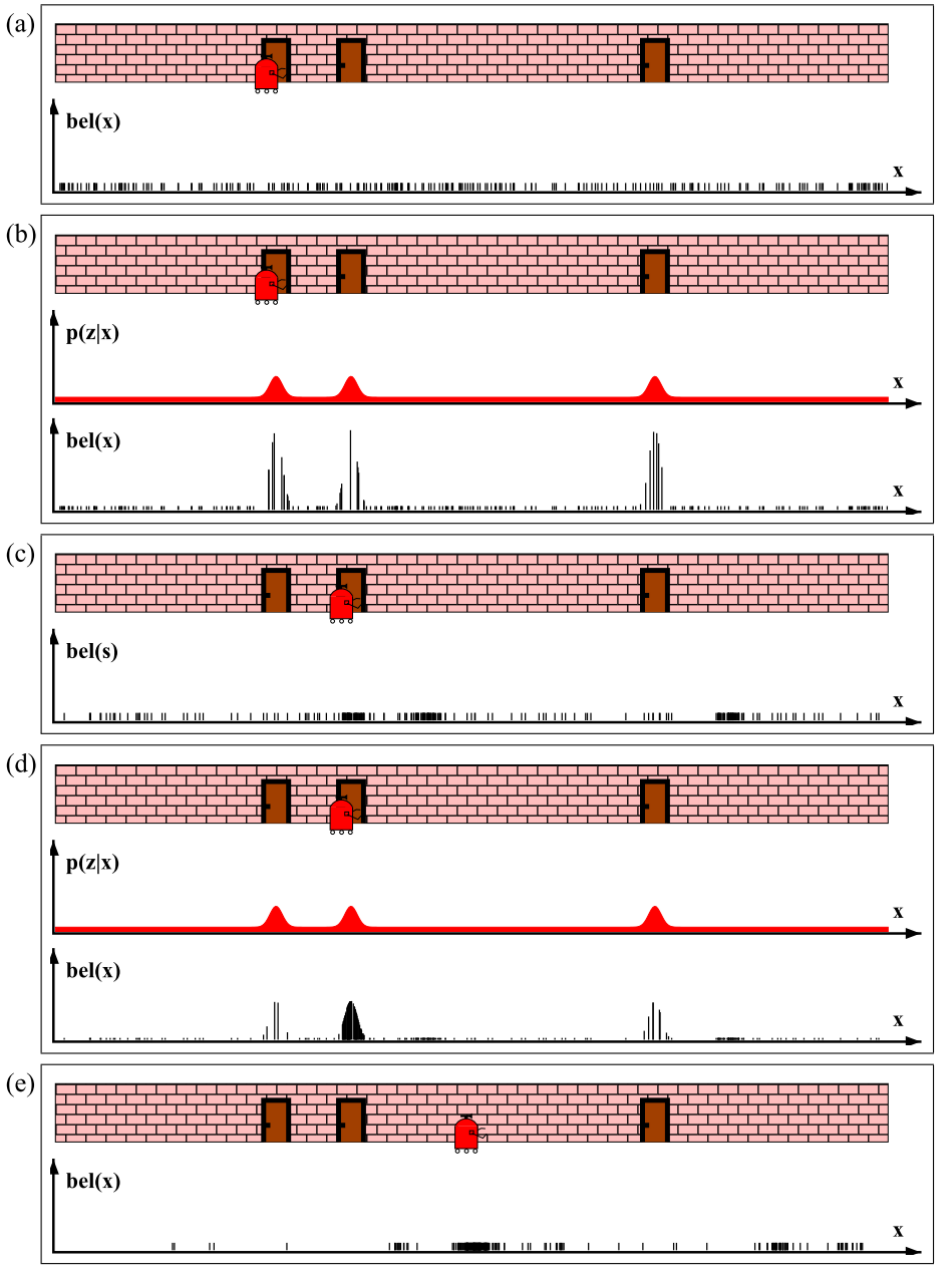


Figura 3.3: Exemplo de localização com o algoritmo de localização de Monte Carlo. Extraída de (THRUN et al., 2005)

Em (c), pode-se ver a disposição das amostras após os passos de reamostragem e movimentação. Na próxima imagem, (d), o robô observou uma porta e com essa informação o peso das amostras é recalculado. E finalmente, em (e), após um novo processo

de reamostragem e movimentação das amostras, pode-se perceber que a maioria delas se acumularam em torno da real localização do robô.

Ainda que o advento da robótica probabilística tenha trazido benefícios, existem também algumas desvantagens, como a ineficiência computacional e a necessidade de aproximação. A ineficiência computacional surge da necessidade de considerar toda a densidade de probabilidade, e o fato de a robótica tratar mundos contínuos gera a necessidade de aproximações (THRUN et al., 2005).

O filtro de partículas está sujeito a erros de aproximação por diferentes causas (THRUN et al., 2005), tais como:

- A aleatoriedade no processo de reamostragem. Suponha um caso onde o estado do robô não mude, e o robô não receba medições do ambiente. O conjunto de partículas inicial será gerado e as partículas serão espalhadas por todo o conjunto de possíveis estados. A transição é determinística, uma vez que $x_t = x_{t-1}$. Nesse contexto, após repetitivas reamostragens, a diversidade do conjunto vai desaparecer, ou seja, M cópias idênticas vão compor o conjunto de partículas. Esse exemplo mostra uma significativa limitação do filtro de partículas, pois o processo de reamostragem induz a perda de diversidade. Mesmo que a variedade do conjunto inicial de partículas diminua, é necessário que a variedade do conjunto de partículas resultantes usadas para estimar crenças verdadeiras aumente. Controlar essa variação (diversidade) é essencial.

- Diferença entre a distribuição proposta e a distribuição alvo. As partículas do conjunto são geradas a partir de uma distribuição proposta que desconsidera as medições. Entretanto, a distribuição alvo depende das medições. Para que o filtro de partículas seja eficiente, espera-se que as distribuições alvo e proposta combinem. Para exemplificar, considere um robô com sensores livres de ruído. Dessa forma, a probabilidade $p(z|x)$ para quase todos os estados de x será zero, com exceção daqueles estados que combinem perfeitamente. Considerando esse exemplo, pode-se perceber um sério problema, já que a distribuição proposta dificilmente gerará uma amostra cujas medidas coincidirão perfeitamente com z. Resumindo, se o robô possuir sensores imprecisos, mas o movimento for preciso, as distribuições alvo e proposta serão similares e o filtro será eficiente. Por outro lado, caso os sensores forem precisos e o movimento for impreciso, as distribuições serão diferentes e o filtro será arbitrariamente ineficiente.

- Outra desvantagem é o problema de privação de partículas. Para calcular a estimativa em um espaço altamente dimensional talvez não existam partículas localizadas na vizinhança do estado correto. Isso normalmente ocorre pelo uso de um número insuficiente de partículas para cobrir todas as regiões de alta probabilidade. Esse problema também pode ocorrer devido à reamostragem aleatória, pois uma infeliz série de números aleatórios pode eliminar amostras localizadas na vizinhança do estado verdadeiro. A cada execução da reamostragem aumentam-se as chances de isso ocorrer, assim, caso o método seja executado por tempo suficiente, eventualmente será gerada uma estimativa arbitrariamente incorreta.

Mesmo com essas desvantagens, diferentes problemas são tratados com sucesso através do uso de filtro de partículas, tais como localização (PRESTES; RITT; FUHR, 2008)

(SABBI; HUBER, 2006) e SLAM (GIL et al., 2010) (MAFFEI et al., 2013).

3.2 Auto Localização: Abordagem Intervalar

Nesta seção são descritos dois métodos intervalares comumente utilizados para resolução do problema de auto localização global: SIVIA e contratores.

No processo de auto localização global, o ambiente no qual o robô esta inserido é definido como uma caixa e usado como espaço de busca inicial para o algoritmo. Assim como o espaço de busca inicial, as demais variáveis do problema como orientação $[\mathbf{o}]$, velocidade $[\mathbf{v}]$ e distância $[z]$, também são representadas de forma intervalar. As variáveis são transformadas em intervalos, os quais incluem a incerteza inerente à informação obtida através das medições dos sensores.

$$[\mathbf{o}] = [-k_o * \sigma_o + \tilde{\mathbf{o}}, \tilde{\mathbf{o}} + k_o * \sigma_o] \tag{3.9}$$
$$[\mathbf{v}] = [-k_v * \sigma_v + \tilde{\mathbf{v}}, \tilde{\mathbf{v}} + k_v * \sigma_v] \tag{3.10}$$
$$[z] = [-k_z * \sigma_z + \tilde{z}, \tilde{z} + k_z * \sigma_z] \tag{3.11}$$

Onde $\tilde{\mathbf{o}}, \tilde{\mathbf{v}}$ e $\tilde{z}$ são as variáveis reais informadas pelo robô, antes da transformação em variáveis intervalares. $\sigma = (\sigma_o, \sigma_v, \sigma_z)$ é o desvio padrão do erro associado as informações, esse erro varia de acordo com os sensores e são normalmente informados pelo fabricante. E $k = (k_o, k_v, k_z)$ é um parâmetro escolhido para definir o tamanho do intervalo de confiança. O parâmetro k é frequentemente definido como o valor 3, isso ocorre devido a representação dos erros dos sensores ser comumente tratada através de uma distribuição normal. Assim dado um desvio padrão σ associado ao erro de uma variável x o intervalo $[-3 * \sigma + x; +3 * \sigma + x]$ abrange a incerteza da variável x, cobrindo $99,73\%$ do valor coberto pela distribuição normal.

3.2.1 Auto Localização Global Utilizando SIVIA

O método SIVIA (MEIZEL et al., 2002), apresentado previamente na Seção 2.5.1, é uma das técnicas utilizadas no tratamento do problema de auto localização global. O funcionamento da abordagem de localização usando SIVIA é descrito no Algoritmo 3. Funções apresentadas no Algoritmo 3 são descritas na Tabela 3.2.

Como visto anteriormente, o resultado do método SIVIA é representado através de um conjunto de caixas não sobrepostas. Para o método de auto localização, esse conjunto de caixas representará uma região do ambiente, mais especificamente, a região onde o robô está localizado (linha 4). O conjunto de caixas é criado através de um processo de bissecção e seleção de caixas. A seleção das caixas que vão compor a solução é definida através de uma função intervalar de teste. Para o problema tratado a função de teste pode ser descrita como apresentado na Equação 3.12. Ou seja, uma hipótese de localização do robô, dado por uma caixa $[\mathbf{x}]$, faz parte da solução quando as distâncias do robô para os marcadores estão totalmente contidas nos intervalos das distâncias observadas (ou quando a caixa é indeterminada, mas pequena o bastante). A hipótese é descartada quando não há nenhuma correspondência entre as distâncias computadas e as distâncias medidas. E ela

Algoritmo 3: Localização usando SIVIA

Data: $f, [\mathbf{x}], \varepsilon, u_{1:n}, z_{0:n}$
Result: $\mathbb{S}$
1 $\mathbb{S} = \emptyset$
2 **for** *t=0 : n* **do**
3 $\quad \mathbb{D} =$ conjunto de medidas $z_{0:n}$ no tempo i;
4 $\quad siviaResult = SIVIA(f, \mathbb{D}, [\mathbf{x}], \varepsilon)$;
5 $\quad [\mathbf{bb}] = hull(siviaResult)$;
6 $\quad \mathbb{S} = siviaResult$
7 $\quad moveI([\mathbf{bb}], [\mathbf{u}])$;
8 $\quad [\mathbf{x}] = [\mathbf{bb}]$
9 **end**
10 $return\ \mathbb{S}$

Funções	O que faz
$hull(\mathbb{X})$	cria uma caixa, com menor tamanho possível capaz de conter todas as caixas do conjunto $\mathbb{X}$
$moveI([\mathbf{bb}], [\mathbf{u}])$	movimenta a caixa $[\mathbf{bb}]$ de acordo com um modelo de movimento intervalar definido e os controles $[\mathbf{u}]$

Tabela 3.2: Funções usadas na abordagem intervalar

é indeterminada nos casos restantes.

$$[t]([\mathbf{x}]) = \begin{cases} 1, & \text{se} \quad \bigcap_{l=1}^{j} \sqrt{(x_1 - {}_{\mathrm{x}}m^l)^2 + (x_2 - {}_{\mathrm{y}}m^l)^2 + (x_3 - {}_{\mathrm{z}}m^l)^2} \quad \subset \quad [z]^l \\ 1, & \text{se} \quad w([\mathbf{x}])) < \varepsilon \\ 0, & \text{se} \quad \bigcap_{l=1}^{j} \sqrt{(x_1 - {}_{\mathrm{x}}m^l)^2 + (x_2 - {}_{\mathrm{y}}m^l)^2 + (x_3 - {}_{\mathrm{z}}m^l)^2} \quad \cap \quad [z]^l = \emptyset \\ [0;1], & \text{se} \quad \bigcap_{l=1}^{j} \sqrt{(x_1 - {}_{\mathrm{x}}m^l)^2 + (x_2 - {}_{\mathrm{y}}m^l)^2 + (x_3 - {}_{\mathrm{z}}m^l)^2} \quad \begin{cases} \not\subset & [z]^l \\ \cap & [z]^l \neq \emptyset \end{cases} \end{cases} \tag{3.12}$$

Onde j corresponde ao número de marcadores observados no ambiente.

Após a definição do conjunto de caixas que representa a localização do robô, é criada uma caixa $[\mathbf{bb}]$ contendo esse conjunto. Essa caixa delimitadora (*bounding box*) é a menor caixa possível que contenha todo o conjunto solução. Quando um modelo de movimento é aplicado à caixa $[\mathbf{bb}]_{t_0}$, os movimentos do robô e a incerteza inerente são integrados à caixa. Nessa nova caixa $[\mathbf{bb}]_{t_1}$, gerada a partir do modelo de movimento, estarão contidas todas as posições possíveis para o robô que estava contido anteriormente na caixa $[\mathbf{bb}]_{t_0}$. Neste contexto, a próxima iteração do algoritmo pode usar $[\mathbf{bb}]_{t_1}$ como espaço de busca (linha 8).

Para a movimentação da caixa de acordo com as informações do robô, foi adotado um modelo de movimento que utiliza os dados de orientação (ϕ, θ, ψ) e velocidade $([\mathbf{v}])$ extraídos dos sensores e uma matriz de rotação 3D $[\mathbf{R}]_{([\phi],[\theta],[\psi])}$ (linha 7). Desta forma, pode-se definir o movimento da caixa a partir da função de inclusão apresentada na Equação 3.13.

$$[\mathbf{bb}]_{t_1} = [f]([\mathbf{bb}]_{t_0}) = [\mathbf{bb}]_{t_0} + [\mathbf{R}]_{([\phi],[\theta],[\psi])} \times [\mathbf{v}] \tag{3.13}$$

A cada iteração a localização do robô é armazenada, assim ao final da execução pode-se definir todo o caminho percorrido pelo robô. A principal vantagem dessa abordagem é

a redução do espaço de busca. A área de incerteza sobre a localização do robô é a menor possível considerando que nenhuma solução factível é descartada. Entretanto, o custo para utilização do SIVIA é exponencial dado o número de dimensões (JAULIN et al., 2001), sendo essa a principal desvantagem do método.

3.2.2 Auto Localização Global Utilizando Contratores

Assim como mostrado na Seção 3.2.1, os dados do problema (espaço de busca inicial, informações sobre movimentação e distâncias dos marcadores) são definidos na forma intervalar. Ao contrário do método utilizando SIVIA, que a cada passo gera como resultado um conjunto de caixas, o método utilizando contratores gera uma única caixa. Um pseudocódigo de localização utilizando contratores pode ser visto no Algoritmo 4.

Esta abordagem, trata o problema como um CSP (descrito na Seção 2.6), onde a equação 3.14 é utilizada como restrição para cada um dos marcadores visíveis $m^{(1..j)}$. Esse conjunto $\mathbb{F}$ de equações irá representar as restrições do problema e pode ser descrito como:

$$\sqrt{(x_1 - {}_{\mathrm{x}}m^l)^2 + (x_2 - {}_{\mathrm{y}}m^l)^2 + (x_3 - {}_{\mathrm{z}}m^l)^2} \quad \subset \quad z^l \qquad \forall\, l = 1..j \tag{3.14}$$

Algoritmo 4: Localização usando Contratores

Data: $f, [\mathbf{x}], u_{0:n}, z_{0:n}$
Result: $\mathbb{S}$

1 $\mathbb{S} = \emptyset$
2 **for** *i=0 : n* **do**
3 $\quad \mathbb{F} =$ conjunto de restrições baseadas em f e $z_{0:n}$ no tempo i, Equação 3.14;
4 $\quad [\mathbf{x}] = Contract([\mathbf{x}], \mathbb{F})$;
5 $\quad \mathbb{S} = [\mathbf{x}]$
6 $\quad moveI([\mathbf{x}], [\mathbf{u}])$;
7 **end**
8 $return\ \mathbb{S}$

A cada iteração o espaço de busca é reduzido por contratores (linha 4), a partir das restrições do problema. A caixa resultante das contrações é movida de acordo com a função de inclusão descrita na Equação 3.13 e apresentada na tabela 3.2, e será definida como espaço de busca na próxima iteração (linha 6). Isso é possível devido à utilização de um modelo de movimento baseado em uma função de inclusão, dessa forma todas as soluções factíveis estarão contidas na nova caixa e ela poderá ser considerada como espaço de busca na próxima iteração.

A vantagem do uso de contratores está relacionada à complexidade do processo. Enquanto o SIVIA tem um custo exponencial, o uso de contratores tem custo polinomial. Contudo, contratores normalmente produzem resultados menos precisos que os resultados gerados pelo SIVIA.

3.3 Trabalhos Relacionados

Nesta seção são apresentados alguns trabalhos que utilizam abordagens probabilísticas e intervalares. Os trabalhos estão classificados em subseções de acordo com a abordagem utilizada.

3.3.1 Abordagens Probabilísticas

O uso de abordagens probabilísticas para tratar problemas da robótica tem se mostrado como a principal opção de diversos pesquisadores. Existindo uma vasta gama de bibliografia. Nesta seção são apresentadas resumidamente algumas dessas pesquisas.

- Prestes, Ritt e Fuhr (2008) apresentam uma proposta para lidar com o problema de auto localização global em ambientes esparsos. Os autores propõem uma combinação do planejador de caminhos baseado em problemas de valor de contorno (BVP) com o algoritmo de localização de Monte Carlo. A estratégia proposta usa informações sobre a estrutura do ambiente no processo de distribuição das partículas, espalhando-as apenas em partes relevantes do ambiente, ao contrário do filtro de partículas tradicional, que distribui as partículas aleatoriamente no espaço livre.

  O ambiente é representado usando uma grade, e através de resultados parciais do processo de esqueletização do ambiente, o método é capaz de determinar as células do ambiente que facilitam a localização do robô. Uma das principais vantagens citadas pelos autores é a distribuição das partículas em regiões mais prováveis de conter a posição do robô. Alem disso, a utilização de limites dinâmicos no planejamento de caminhos faz com que o robô possa agir de maneira inteligente e gerar diferentes comportamentos durante a navegação.

- Sabbi e Huber (2006) tratam do problema de rastreamento de objetos de acordo com suas características visuais. Os autores propõem duas abordagens utilizando câmeras estereoscópicas e filtro de partículas. Na primeira abordagem são usados dois conjuntos de partículas, um para cada câmera (direita e esquerda) e o resultado é estabelecido de acordo com restrições estereoscópicas sobre esses conjuntos. Na segunda abordagem, o filtro de partículas é usado considerando um espaço tridimensional.

  O autor apresenta resultados que indicam que o filtro de restrições estereoscópicas encontram os objetos mais rapidamente, entretanto os erros de rastreamento são ligeiramente mais altos quando comparado com o filtro 3d. Por outro lado o filtro 3d é mais preciso, mas demanda um pouco mais de tempo para localizar o objeto.

- Li et al. (2011) tratam do problema de localização de fontes de odor. Os autores usam robôs em ambientes externos com correntes de ar que variam no decorrer do tempo. O algoritmo proposto é baseado em filtro de partículas e trata o problema em tempo real. Para avaliar o método, ele foi comparado com um método bayesiano baseado em inferência. A partir dos experimentos, na comparação dos métodos, o método proposto se mostrou mais robusto.

- Maffei et al. (2013) propõe um método para o problema de SLAM chamado SDP-SLAM baseado em uma estratégia de submapas. O método proposto combina ca-

racterísticas de dois métodos, SegSLAM e DP-SLAM, e é próprio para aplicações cujo ambiente alvo seja estruturado. Do SegSLAM, o método proposto aproveita o conceito do uso de segmentos do ambiente contendo múltiplos mapas. Uma estrutura de dados otimizada para armazenamento dos mapas das partículas também é usada. O aspecto distribuído dessa estrutura, derivado do método DP-SLAM, permite que o ambiente seja segmentado em vários submapas. A avaliação do SDP-SLAM foi feita em ambientes reais e simulados e mesmo usando menos partículas o método mostrou melhores soluções que DP-SLAM. Comparando o método proposto com o SegSLAM, as avaliações mostraram que o SDP-SLAM pesquisa soluções com alinhamento de erros mais baixo.

- Arturo et al. (2010) propõem uma abordagem para o problema de SLAM baseado em características para um time de veículos autônomos cooperativos. O método é baseado no uso de câmeras estéreo capazes de observar *landmarks* visuais no ambiente. Os robôs se movem e, de acordo com as observações feitas, constroem o mapa do ambiente usando filtro de partículas. Após os testes e avaliações demonstrou-se que a abordagem proposta é adequada para pequenos times.

3.3.2 Abordagens Intervalares

A análise de intervalos é constantemente aplicada em problemas da robótica para tratar incertezas que são inevitáveis em decorrência do uso de sensores para medições. Dentre as principais vantagens da utilização de uma abordagem intervalar estão a flexibilidade (pode ser utilizada para diferentes problemas) e a garantia (resultados são matematicamente garantidos, dada a correta modelagem das incertezas) (MERLET, 2011). Diferentes problemas da robótica são tratados através de abordagens intervalares e algumas dessas soluções são descritas nessa seção.

- Langerwisch e Wagner (2012) propuseram um método intervalar para o problema de localização de robôs móveis. O método é aplicado em um ambiente 2D, interno e estruturado. O mapa utilizado é baseado em características. No processo de localização são utilizadas informações do laser e odometria. O algoritmo utilizado é o RSIVIA (JAULIN, 2009), cuja ideia fundamental é a mesma do algoritmo SIVIA. As modificações do algoritmo foram feitas visando maior robustez contra leituras inconsistentes retornadas pelos sensores. A principal contribuição do trabalho é a aplicação do RSIVIA em um problema que utiliza um mapa baseado em característica, com laser rotativo e múltiplas medidas por iteração. Nos testes feitos pelo autor, foi possível perceber que a real posição do robô está sempre contida no conjunto solução, o qual pode ser computado em tempo real.

- Guyonneau et al. (2012) tratam o problema do robô raptado através de uma abordagem intervalar. Para isso, são utilizadas informações provenientes de sensores de alcance e odometria, além de um mapa discretizado do ambiente alvo. O algoritmo proposto permite lidar com os problemas de rastreamento de posição, localização global e detecção de situações de rapto do robô. Os autores tratam do possível aparecimento de leituras inconsistentes retornadas pelos sensores através da técnica de relaxamento de intersecção.

  O problema abordado pelo artigo é tratado como um problema de satisfação de restrições e utiliza contratores para encontrar a solução para o problema. Prevendo que

existam situações em que os contratores não conseguem reduzir as variáveis até um tamanho mínimo desejável, processos de bissecção e seleção de caixas são utilizados nesse resultado a fim de melhorá-lo. Os autores concluem que a utilização de abordagens baseadas em análise de intervalos são alternativas interessantes para as clássicas abordagens probabilísticas.

- Jaulin (2009) propõe uma solução intervalar para o problema de SLAM em ambientes subaquáticos. Ele trata o problema de SLAM como um problema de satisfação de restrições e utiliza propagação de intervalos. As leituras do ambiente são feitas através de um sonar, entretanto é necessário um operador humano para o reconhecimento dos *seamarks*. O sucesso sobre o uso de análise de intervalos para resolução do problema pode ser comprovada através de experimentos reais.

- Em outro trabalho de Jaulin (2011), o problema de SLAM é tratado sem o uso de *landmarks*, utilizando um sonar omnidirecional para detectar obstáculos. Mesmo tendo obtido resultados satisfatórios, alguns problemas são apontados, tais como, a não adequação para aplicações em tempo real e a presença de leituras inconsistentes, que podem gerar erros ou até mesmo impossibilitar uma solução. Outro problema mencionado pelo autor é a presença de obstáculos dinâmicos, que na solução proposta são tratados como leituras inconsistentes, não tendo uma representação no mapa do ambiente.

- Aubry et al. (2013) usam sensores proprioceptivos para detectar e localizar ciclos feitos pelo robô durante sua trajetória. O método foi testado com sucesso em um experimento real em duas dimensões. Ele apresentou um custo computacional menor que métodos de detecção que se baseiam em imagens extraídas do ambiente e reduziu o número de falsas detecções.

- Pepy et al. (2010) propuseram o algoritmo Box-RRT com o objetivo de planejar um caminho seguro de um ponto origem até um ponto destino. Para isso, combinou-se *Rapidly-exploring Random Trees* (RRT) com análise de intervalos. O algoritmo se mostrou capaz de encontrar caminhos seguros, evitando obstáculos. Entretanto, quando as incertezas envolvidas no sistema excedem um determinado limite, um caminho seguro pode não ser encontrado.

3.3.3 Abordagem Híbrida para o Problema de Localização

Abdallah, Gning e Bonnifait (2008) propuseram o *Box Particle Filter* (BPF), uma extensão do algoritmo filtro de partículas que lida com dados intervalares através da análise de intervalos. Diferentemente do filtro de partículas tradicional que usa partículas para representação de posições pontuais, o BPF usa caixas de partículas. Essas caixas podem representar uma partícula com localização indeterminada dentro da caixa, ou um conjunto de partículas distribuídas cobrindo toda a área da caixa.

No filtro de partículas, eficiência e precisão são dependentes principalmente do número de partículas utilizadas. Para algumas aplicações o número de partículas necessário é muito grande, impossibilitando implementações em tempo real. A utilização de dados intervalares no método proposto por Abdallah, Gning e Bonnifait o torna mais eficiente, uma vez que usa um número muito menor de caixas de partículas quando comparado

com o número de partículas usadas no filtro de partículas tradicional, isso reduz o custo computacional requerido.

O funcionamento do método pode ser descrito em uma sequência de passos. São eles:

- Inicialização

  O espaço de busca é dividido em N caixas $[x^{(i)}]_{i=1}^{N}$ não sobrepostas, cada uma associada a um fator de importância inicialmente equivalentes. Essas caixas substituem as partículas usadas no filtro de partículas tradicional.

- Propagação ou predição

  Conhecendo as caixas $[x^{(i)}]_{i=1}^{N}$ e as ações $[u_t]$ no instante t, as caixas no instante $t+1$ são construídas a partir da equação de propagação $[x_{t+1}^{(i)}] = [f]([x_t^{(i)}], [u_t])$, sendo $[f]$ uma função de inclusão.

- Atualização de medição

  Nesta etapa as novas medidas são usadas para ajuste dos pesos e contração das caixas.

  - Inovação

    O processo de inovação corresponde a encontrar a intersecção entre a caixa que representa as observações do robô e a predição da caixa de partículas.

  - Probabilidade

    Nesta etapa é definida a probabilidade de cada uma das caixas $[x^{(i)}]_{i=1}^{N}$. Quando as medições das caixas de partículas não tem intersecção com as medições do robô, a caixa de partículas relacionada a ela é penalizada. Quando a medição está incluída na medição real, a caixa de partículas relacionada a ela deve ser favorecida.

  - Contração

    Cada caixa de partículas, quando propagada, agrega as incertezas do sistema. Isso faz com que as caixas aumentem de tamanho. Para reduzir essas caixas, eliminando partes não consistentes, são usados contratores.

  - Atualização de pesos

    Para atualizar o peso das caixas de partículas é feita a multiplicação de todos os pesos anteriormente assumidos pela caixa pela probabilidade da caixa de partículas. Após a definição do peso de todas as caixas, é feita uma normalização desses valores.

- Estimativa

  Usando caixas de partículas o resultado pode ser estimado escolhendo o centro da caixa com peso mais alto, ou través de uma média ponderada usando a equação $x_k = \sum_{i=1}^{N} w_k^i C_k^i$, onde w_k^i é o peso e C_k^i é o centro da caixa i.

- Reamostragem

  A reamostragem é feita de acordo com o peso das caixas, isto é, caixas com pesos maiores tem mais probabilidade de sobreviver que caixas com pesos menores. Para

obter soluções mais precisas, é possível dividir caixas em várias subcaixas (através de bissecções), refinando a solução em torno de regiões de alta probabilidade e eliminando caixas com pesos baixos.

O problema tratado com o método foi a localização de um veículo terrestre com dados de gps, giroscópio e odômetro. Na comparação da solução encontrada pelo método BPF com a solução do filtro de partículas pode-se notar um desempenho equivalente em relação à qualidade da solução encontrada, a precisão dos resultados são bastante similares. Uma vez que a etapa de reamostragem elimina caixas com baixo fator de importância, a característica probabilística do filtro de partículas é herdada pelo BPF. Entretanto, o BPF usa apenas 10 caixas, enquanto o filtro de partículas tradicional precisou de 3000 partículas. Esse número reduzido de caixas de partículas impactou no tempo de execução do método, tornando o BPF consideravelmente mais rápido que o filtro de partículas. Dadas as vantagens apresentadas pelo método ele foi usado em outras pesquisas (DANDACH et al., 2012) (GNING; RISTIC; MIHAYLOVA, 2011) (GNING; RISTIC; MIHAYLOVA, 2012) (PETROV et al., 2012).

4 UMA ABORDAGEM HÍBRIDA PARA O PROBLEMA DE AUTO LOCALIZAÇÃO GLOBAL

O problema tratado nesta pesquisa é o de auto localização global. Para formular uma proposta de solução foram estudados alguns dos métodos tradicionalmente aplicados a esse problema. A partir desse estudo optou-se pelo desenvolvimento de um método híbrido com base em duas abordagens diferentes: probabilística e intervalar.

O filtro de partículas é um método probabilístico que ganhou popularidade no tratamento de problemas de localização, pois é um filtro não paramétrico, capaz de lidar com crenças multimodais. Essas características tornaram o filtro de partículas um dos componentes da hibridização proposta.

Ainda que apresente diversas características positivas, o filtro de partículas também possui algumas desvantagens (como apresentado no Capítulo 3.1). Na busca de um método complementar, que contribua para localização de soluções mais precisas e evite uma convergencia errada, optou-se pelo uso de análise de intervalos.

O princípio básico da hibridização proposta é a limitação da área de incerteza sobre a posição do robô através da análise de intervalos. É a partir dessa informação que o filtro de partículas poderá atuar. A fim de manter as garantias provenientes da análise de intervalos, não são usadas informações probabilísticas nas computações intervalares. Porém, os cálculos probabilísticos são limitados por definições intervalares. Assim, é possível descobrir a posição do robô com o uso de probabilidades, dentro de uma região limitada e matematicamente garantida pela análise de intervalos. Com a abordagem proposta pretende-se alcançar benefícios como:

- *Alta cobertura da região de incerteza.* Dada a redução do espaço de busca, as partículas podem ser melhor distribuídas. O método intervalar define os limites onde é possível que o robô esteja localizado, e as partículas são forçadas a permanecer dentro desse limite. Dessa forma, sem aumentar o número de partículas usadas pelo método, é possível aumentar a cobertura da área de real interesse.

- *Rápida detecção de má convergência.* Se o método convergir para um local incorreto, a abordagem intervalar produz como solução conjuntos vazios, os quais são facilmente detectados e sinalizam a ocorrência de erros. Nesses casos, o método pode iniciar um processo a fim de evitar uma convergencia incorreta, no método proposto, esse processo é a reinicialização do método.

- *Um resultado com delimitação da incerteza matematicamente garantida*. A abordagem intervalar não descarta nenhuma solução factível. Assim, a solução intervalar é matematicamente garantida, uma vez que o problema tenha sido corretamente modelado. O resultado do filtro de partículas está dentro dos limites do resultado intervalar, então a incerteza associada às partículas também pode ser definida a partir desses limites garantidos.

- *No caso do resultado intervalar não prover informação suficiente sobre a localização do robô, mais informações podem ser extraídas utilizando as informações das partículas*. A região definida pelo método intervalar pode ser muito grande e prover pouca informação sobre a localização do robô, dado que essa abordagem não descarta nenhuma solução factível. Usando partículas nessa região é possível extrair informações probabilísticas sobre a possível localização do robô dentro da região.

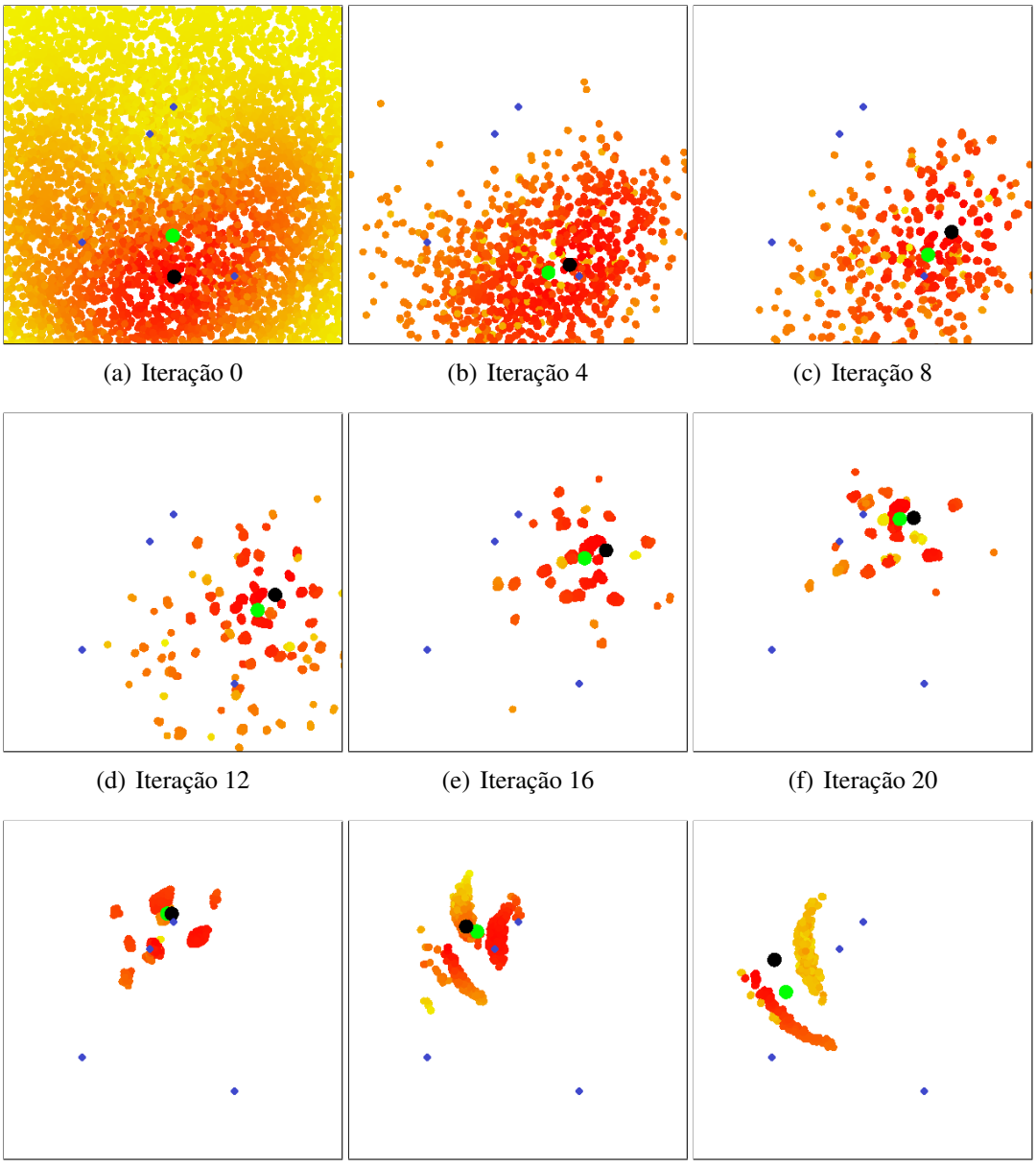
Foi necessário uma implementação do filtro de partículas a partir das restrições do problema abordado, tanto como parte dos métodos propostos, quanto para comparação dos experimentos realizados. Na Seção 4.1 são apresentados alguns detalhes dessa implementação. A primeira proposta de hibridização foi a combinação do filtro de partículas com o SIVIA. Tendo em vista o custo computacional associado a essa técnica, uma segunda hibridização foi proposta. Nessa hibridização optou-se por usar filtro de partículas com contratores, cujo custo computacional é menor. Contratores, assim como o SIVIA, não descartam quaisquer soluções factíveis para o problema, contudo, normalmente não alcançam resultados tão precisos. Todos os métodos foram implementados utilizando a linguagem de programação C++ e a biblioteca IBEX para computações intervalares. A biblioteca IBEX é um projeto acadêmico *open-source* utilizado por outros grupos de pesquisa de robótica intervalar.

4.1 Implementação do Filtro de Partículas de Acordo com as Especificidades do Problema Tratado

O FP implementado segue as mesmas etapas básicas descritas na Seção 3.1.1. A cada iteração, o método executa cada uma das etapas que compõem a iteração presente no filtro de partículas (Algoritmo 2). O comportamento das partículas em algumas dessas iterações pode ser visto na Figura 4.1, que apresenta a evolução do processo de localização utilizando apenas o filtro de partículas. Na figura, os transponders são representados em circulos azuis, as partículas em uma escala de cores do amarelo ao vermelho de acordo com seu fator de importância, a posição real do robô é representada em preto e a média ponderada das partículas é representada em verde.

- Criação da população

  A população de partículas é criada e espalhada aleatóriamente sobre a região de incerteza. No método tradicional isso significa espalhar as partículas por todo o ambiente. Essa etapa de execução é apresentada na Figura 4.1(a). Nos métodos propostos, essa etapa sofre uma modificação, a qual é apresentada nas próximas seções.



(a) Iteração 0 (b) Iteração 4 (c) Iteração 8

(d) Iteração 12 (e) Iteração 16 (f) Iteração 20

(g) Iteração 24 (h) Iteração 28 (i) Iteração 32

Figura 4.1: Iterações do Filtro de Partículas.

- Movimentação das partículas

  As partículas são movidas de acordo com os movimentos executados pelo robô, essa movimentação é dada pela probabilidade $p(x_t|u_t, x_{t-1})$. Onde u_t é o movimento executado pelo robô no instante t o qual inclui informações de velocidade e angulação do robô. O parâmetro x_{t-1} é a posição da partícula no instante $t-1$ e x_t é a posição a ser estimada. O cálculo para definir a nova posição das partículas é dado na Equação 4.1, onde $\mathbf{R}_{(\phi,\theta,\psi)}$ é uma matriz de rotação 3d. Para calcular a movimentação das partículas, cada uma delas recebe uma alteração nos valores de u_t, um valor é inserido à informação a fim de tratar possíveis erros das leituras.

$$\begin{aligned} u_t &= u_t + Erro \\ x_t &= x_{t-1} + \mathbf{R}_{(\phi,\theta,\psi)} * (u_t) \end{aligned} \tag{4.1}$$

  O erro associado a cada parâmetro relacionado ao movimento do robô é um erro aleatório de distribuição normal obtido através da Equação 4.2, onde a média é 0, b é a variância e $rand(-1, 1)$ é um gerador de números pseudo aleatório de distribuição uniforme entre -1 e 1 (THRUN et al., 2005).

$$Erro \ = \ \frac{b}{6} \sum_{i=1}^{12} rand(-1, 1) \tag{4.2}$$

- Pesagem das partículas

  A cada iteração, o robô observa um conjunto de marcadores e obtém a distância referente a cada um. De forma análoga, as partículas medem a distância que se encontram os marcadores dado o mapa m do ambiente. Após a aquisição dessas informações, as partículas recebem um fator de importância de acordo com a similaridade das leituras obtidas por elas no mapa m, e das leituras obtidas pelo robô sobre determinados marcadores. A Equação 4.3 mostra como é calculado o fator de importância para cada partícula x_t, sendo z_t^j a leitura de distância entre o robô e o marcador j no instante de tempo t.

$$\prod_{j=1}^{J} p(z_t^j|x_t, m) \tag{4.3}$$

  A probabilidade $p(z_t^k|x_t, m)$ pode ser calculada através da fórmula da gaussiana descrita na Equação 4.4, onde μ representa a média e σ^2 é a variância da gaussiana. Sendo que l é uma das leituras feitas pela partícula, a média é definida pela leitura correspondente feita pelo robô e a variância é um parâmetro pre-definido. A variância influencia em relação ao nível de qualidade atribuída a uma leitura.

$$f(l, \mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\left(-\frac{(l-\mu)^2}{2\sigma^2}\right)} \tag{4.4}$$

  A Figura 4.1 mostra as partículas de diferentes cores, em uma escala do amarelo ao vermelho, essa escala de cores representa graficamente o fator de importância atribuido à cada partícula. Quanto mais próximo à cor amarela, menor é o fator

de importância da partícula e quanto mais próximo ao vermelho, maior o fator de importância.

- Reamostragem

  Nesta etapa a população passa por um processo de reamostragem, onde algumas partículas serão descartadas, após esse processo uma nova população será definida. As partículas que farão parte dessa nova população são escolhidas através do algoritmo da roleta. Esse algoritmo considera que partículas com maior fator de importância tem maior probabilidade de sobreviver ao processo, um pseudocódigo é dado no Algoritmo 5.

Algoritmo 5: Roleta

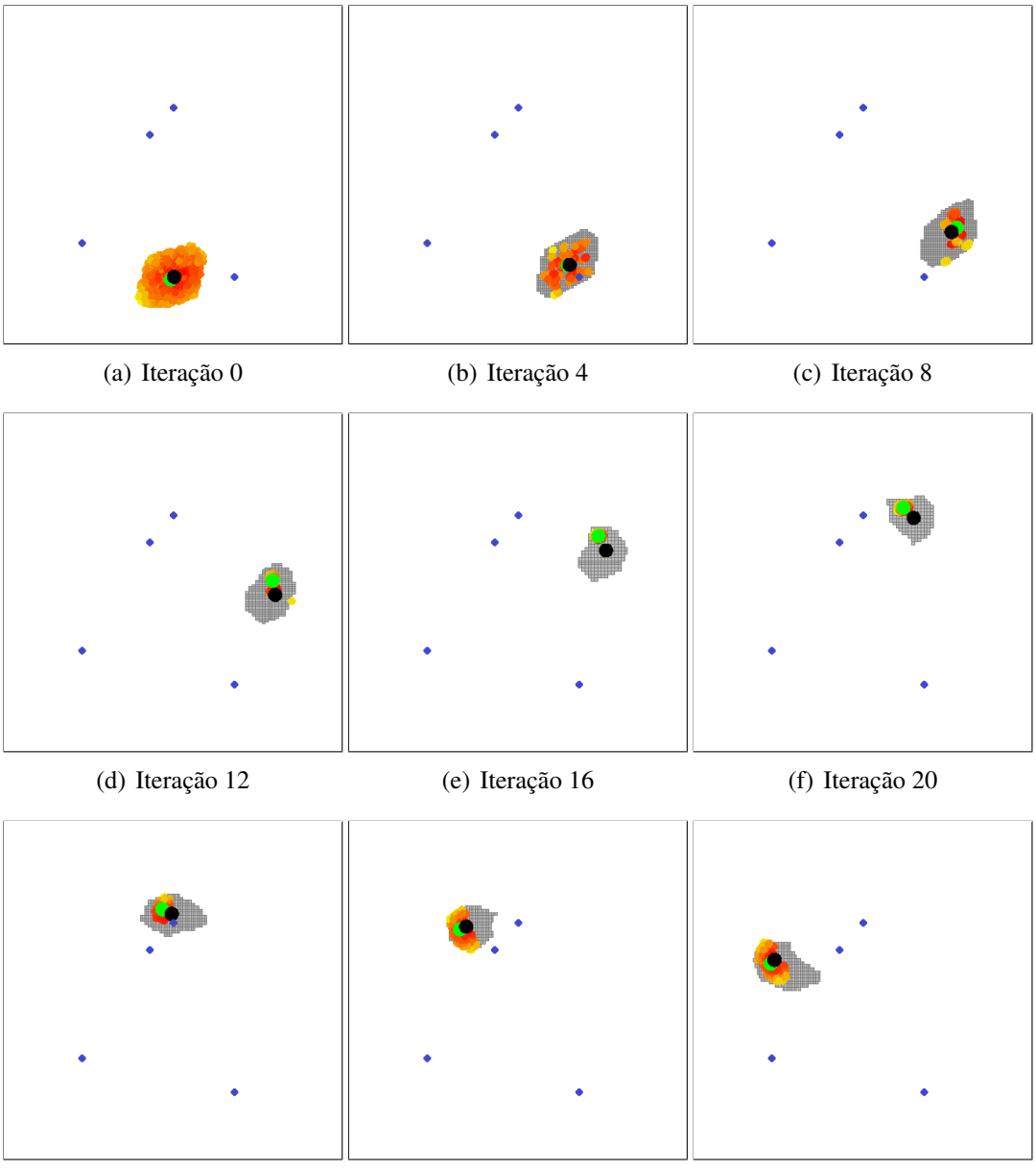
```
   Data: M, populacaoAtual
   Result: novaPopulacao
1  cont = 1;
2  a_i = Σ_1^i fator de importância da partícula i;
3  while cont ≤ M do
4  |  r = valor aleatório com probabilidade uniforme [0;1];
5  |  i=1;
6  |  while a_i < r do
7  |  |  i = i + 1;
8  |  |  a_i = a_i + fator de importância da partícula i;
9  |  end
10 |  novaPopulacao[cont] = populacaoAtual[i];
11 |  cont = cont + 1;
12 end
```

4.2 Primeira Hibridização: Filtro de Partículas com SIVIA

A primeira opção para hibridização utiliza o algoritmo SIVIA em conjunto com o filtro de partículas. O SIVIA contribui para delimitar a região de incerteza sobre a localização do robô. Essa região é então definida como espaço de busca para o filtro de partículas. Com o espaço de busca reduzido é possível obter uma melhor distribuição das partículas usadas no filtro. O Algoritmo 6 apresenta os passos de execução do método proposto. Algumas funções usadas no algoritmo são descritas nas Tabelas 3.1 e 3.2.

Inicialmente o algoritmo não possui nenhuma informação sobre a localização do robô, assim, o espaço de busca inicial abrange todo o ambiente, podendo ter limites definidos ou indefinidos, nesse caso representados por intervalos infinitos (ex.: $[-\infty; +\infty]$).

A partir de um conjunto de restrições (linha 2), as quais são criadas de acordo com os marcadores detectados no atual instante de tempo, e do conjunto de medidas (transformadas em intervalos) referentes à distância entre esses marcadores e o robô (linha 3), o algoritmo executa o método SIVIA (linha 4). Como resultado, o SIVIA vai retornar um conjunto de caixas representando a região do ambiente na qual o robô está localizado. A população é criada e partículas são distribuídas aleatoriamente sobre o espaço de busca delimitado pelo SIVIA (linha 5).



(a) Iteração 0 (b) Iteração 4 (c) Iteração 8

(d) Iteração 12 (e) Iteração 16 (f) Iteração 20

(g) Iteração 24 (h) Iteração 28 (i) Iteração 32

Figura 4.2: Iterações do Filtro de Partículas com Sivia.

Algoritmo 6: Filtro de Partículas com SIVIA

Data: $f, [\mathbf{x}], \varepsilon, u_{1:n}, z_{0:n}$
Result: $\mathbb{S}$

1 $\mathbb{S} = \emptyset$
2 $\mathbb{F} =$ conjunto de restrições f;
3 $\mathbb{D} =$ conjunto de medidas z_0
4 $siviaResult = SIVIA(\mathbb{F}, \mathbb{D}, [\mathbf{x}], \varepsilon)$;
5 $\mathbb{P} = createParticle(m, siviaResult)$;
6 **for** *t = 1 : n* **do**
7 $[\mathbf{bb}] = hull(siviaResult)$;
8 $moveI([\mathbf{bb}], [\mathbf{u}])$;
9 $[\mathbf{x}] = [\mathbf{bb}]$;
10 $\mathbb{F} =$ conjunto de restrições f;
11 $\mathbb{D} =$ conjunto de medidas z_t;
12 $siviaResult = SIVIA(\mathbb{F}, \mathbb{D}, [\mathbf{x}], \varepsilon)$;
13 $move(\mathbb{P}, u)$;
14 **for** *j = 0 : m* **do**
15 **if** $p^j \cap siviaResult = \emptyset$ **then**
16 $\mathbb{P} = \mathbb{P} \setminus p^j$;
17 $\mathbb{P} = \mathbb{P} \cup createParticle(1, siviaResult)$;
18 **end**
19 **end**
20 $weight(\mathbb{P})$;
21 $resampling(\mathbb{P})$;
22 $\mathbb{S} = avgParticle(\mathbb{P})$;
23 **end**
24 $return\ \mathbb{S}$

Em uma sequencia de iterações, a região definida pelo SIVIA é limitada por uma caixa (linha 7), essa caixa é a menor possível que seja capaz de conter todas as caixas obtidas pelo SIVIA. Considerando os movimentos executados pelo robô o método agrega essa informação aos cálculos. A caixa que delimita a região de incerteza é movida de acordo com uma função de inclusão (linha 8), e torna-se o espaço de busca para a próxima iteração do método (linha 9). A função de inclusão é apresentada na Equação 3.13.

A partir de um conjunto de restrições (linha 10) e medidas (linha 11), o algoritmo executa o método SIVIA (linha 12). A população atual é atualizada, movendo as partículas de acordo com as informações procedentes do robô (linha 13). Todas as partículas são avaliadas, nessa avaliação as partículas localizadas fora do espaço de busca são descartadas. Para cada partícula descartada, uma nova partícula é inserida aleatoriamente no espaço de busca (linhas 14 - 19).

Nos próximos passos, as partículas passam pelas etapas de pesagem (linha 20) e reamostragem (linha 21), comuns para o filtro de partículas. O processo de reamostragem é baseado no tradicional método da roleta. A partir da nova população é feita uma média ponderada da localização das partículas em relação aos fatores de importância associados a elas, essa posição resultante é definida como a localização do robô naquele instante e é armazenada no conjunto solução (linha 22).

A Figura 4.2 apresenta um exemplo de propagação do método híbrido de filtro de particulas e SIVIA. Além de mostrar a posição real do robô (preto), os transponders (azul), as partículas (tons de amarelo/vermelho) e a média ponderada das partículas (verde), a figura mostra em cinza o conjunto de pequenas caixas criadas pelo SIVIA. Podemos observar durante toda a progressão do filtro que as partículas sempre ficam contidas dentro da área demarcada pelas caixas, tendendo a se concentrar muito mais rapidamente do que no exemplo mostrado na Figura 4.1 do filtro de partículas tradicional.

Espera-se com esse método híbrido uma maior precisão nos resultados, dado que o método intervalar adotado reduz a região de incerteza. Entretanto, essa redução pode não ser suficiente para justificar o custo computacional associado à utilização do SIVIA, que será mostrado na seção de experimentos. Por esse motivo, uma segunda proposta de hibridização foi adotada, desta vez utilizando contratores.

4.3 Segunda Hibridização: Filtro de Partículas com Contratores

A segunda proposta de hibridização é composta também pelo filtro de partículas, mas desta vez com uma abordagem intervalar diferente, usando contratores. O uso de contratores foi motivado pela fácil redução de domínios a partir de restrições do problema e do baixo custo computacional dessa técnica. O Algoritmo 7 descreve o funcionamento do método, algumas funções utilizadas no algoritmo são descritas nas Tabelas 3.1 e 3.2.

Algoritmo 7: Filtro de Partículas com Contratores

Data: $f, [\mathbf{x}], u_{1:n}, z_{0:n}$
Result: $\mathbb{S}$

$\mathbb{S} = \emptyset$
$\mathbb{F} =$ conjunto de restrições f usando z_0;
[**x**]= $Contract([\mathbf{x}], \mathbb{F})$;
$\mathbb{P}$ = $createParticles(m, [\mathbf{x}])$;
for *t = 1 : n* **do**
    $moveI([\mathbf{x}], [\mathbf{u}])$;
    $\mathbb{F} =$ conjunto de restrições f usando z_t;
    [**x**]= $Contract([\mathbf{x}], \mathbb{F})$;
    $move(\mathbb{P}, u)$;
    for *j = 0 : m* **do**
        if $p^j \cap [\boldsymbol{x}] = \emptyset$ **then**
            $\mathbb{P} = \mathbb{P} \setminus p^j$;
            $\mathbb{P} = \mathbb{P} \cup createParticles(1, [\mathbf{x}])$;
        end
    end
    $weight(\mathbb{P})$;
    $resampling(\mathbb{P})$;
    $\mathbb{S} = avgParticle(\mathbb{P})$;
end
$return\ \mathbb{S}$

O espaço de busca inicial é definido por uma caixa, essa caixa inicialmente abrange

todo o ambiente, uma vez que a incerteza é global. São definidas restrições baseando-se nos marcadores visualizados e nas distâncias entre eles e o robô (linha 2). A partir dessas restrições o método reduz o espaço de busca utilizando contratores (linha 3). As partículas são criadas e espalhadas aleatoriamente pelo espaço de busca previamente definido (linha 4), ou seja, na caixa gerada pelos contratores. Com o espaço de busca reduzido, as partículas ficam restritas aos limites impostos pela caixa definida como espaço de busca.

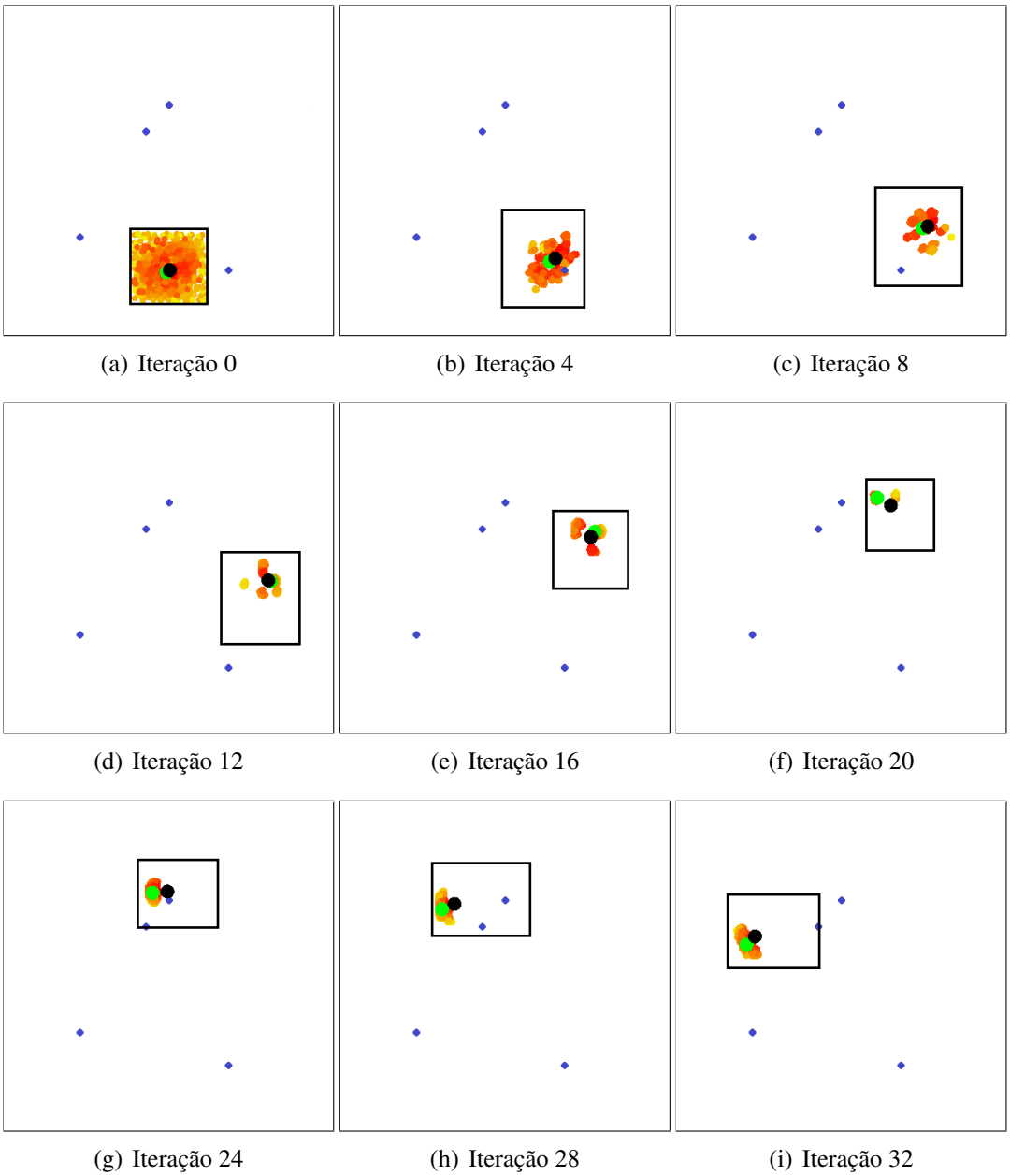
A cada nova informação de movimento e leituras do robô, é executada uma sequencia de iterações. Os movimentos realizados pelo robô são considerados no processo de localização, assim, a caixa que representa o atual espaço de busca é atualizada de acordo com o movimento e será o espaço de busca inicial para a próxima iteração (linha 6).

A partir de um conjunto de restrições (linhas 7) o método reduz o espaço de busca utilizando contratores (linha 8). Considerando os movimentos do robô, as partículas são atualizadas (linha 9).

As partículas da população passam por uma avaliação e as que estiverem fora dos limites do espaço de busca são descartadas. Para cada partícula descartada, uma nova partícula é gerada aleatoriamente no espaço de busca e incorporada à população atual (linhas 10 - 15).

Após essas etapas, acontecem os processos de pesagem (linha 16) e reamostragem (linha 17) das partículas da população atual. A partir dessa população é feita a média da localização das partículas (linha 18), essa posição média é considerada a localização do robô e é armazenada no conjunto solução.

Por fim, a Figura 4.3 apresenta um exemplo de propagação do método usando filtro de partículas e contratores. Dessa vez, o espaço de busca das partículas é definido pela caixa com contorno preto mostrada em cada passo da figura. Quando comparado ao método híbrido usando SIVIA, mostrado na Figura 4.2, podemos ver que usando contratores o espaço de soluções é sempre um pouco maior. Porém essa diferença é muita pequena para encorajar a escolha do SIVIA, uma vez que contratores possuem um custo computacional consideravelmente menor.



(a) Iteração 0 (b) Iteração 4 (c) Iteração 8

(d) Iteração 12 (e) Iteração 16 (f) Iteração 20

(g) Iteração 24 (h) Iteração 28 (i) Iteração 32

Figura 4.3: Iterações do Filtro de Partículas com contratores.

5 EXPERIMENTOS E DISCUSSÕES

Neste capítulo são apresentados e discutidos os resultados de experimentos realizados a fim de avaliar a abordagem proposta. Os resultados gerados pelos métodos híbridos são comparados entre eles e com o filtro de partículas tradicional. Os experimentos foram feitos através da simulação de ambientes subaquáticos contendo marcadores distinguíveis cujas posições são conhecidas em todos os instantes de tempo. O robô utilizado nos experimentos é capaz de extrair informações sobre sua velocidade linear e angulação, além de detectar os marcadores e a distância que está deles.

A extração de dados necessários para a execução dos experimentos foi feita através de simulações. O simulador utilizado durante a pesquisa foi o Simulador MORSE (ECHEVERRIA et al., 2011), o qual disponibiliza diferentes modelos de robôs. O modelo escolhido foi um submarino genérico, equipado com um sensor *loch-doppler* e um sensor giroscópio, capazes de extrair informações sobre velocidade linear e ângulos de euler respectivamente. *Transponders* foram usados como marcadores, que durante a simulação emitem sinais distinguíveis perceptíveis pelo robô. A Figura 5.1 apresenta a interface da simulação utilizada para a extração de dados.

Foram propostos três ambientes de teste 3D com dimensões de 400m x 400m x 400m, sendo o espaço de busca inicial representado por

$$[\mathbf{x}] = [-200; 200] \times [-200; 200] \times [-400; 0].$$

Os ambientes são diferenciados pela quantidade de informações disponíveis para o processo de localização. Os ambientes 1, 2 e 3 possuem respectivamente 2, 4 e 8 *transponders* distribuídos de forma parcialmente aleatória. Mais especificamente, o ambiente foi dividido em partes iguais e os *transponders* foram posicionados aleatóriamente em cada uma dessas partes a fim de posicionar marcadores em todo o ambiente.

A Tabela 5.1 apresenta alguns parâmetros usados nos testes: o ruído dos sensores e o parâmetro k que define o tamanho dos intervalos das medições (como mostrado na Seção 3.2).

Durante as simulações o robô percorreu três trajetórias diferentes. Na primeira trajetória, apresentada na Figura 5.2, o robô simula uma trajetória circular descendente, como se estivesse monitorando 360° de uma estrutura. Isso pode ser feito, por exemplo, para monitoramento de dutos subaquáticos e plataformas petrolíficas. A segunda trajetória, apresentada na Figura 5.3, visa a cobertura do espaço de busca. Esse tipo de trajetória pode ser feita em casos de vigilância de ambientes ou, por exemplo, para busca de minas

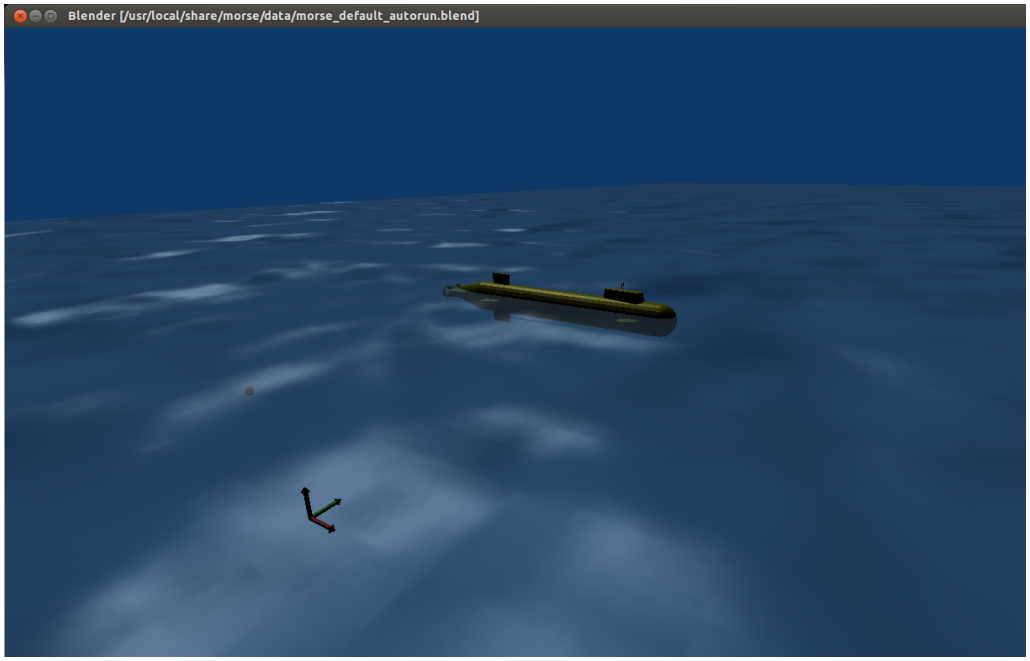


Figura 5.1: Interface - Ambiente de simulação MORSE

Tabela 5.1: Parâmetros para os experimentos

Medições	**Ruído** (σ=desvio padrão)	k
Sensor *loch-doppler*	0.04 m	k_v= 3
Sensor giroscópio	0.02°	k_o= 3
Distância dos *transponders*	0.3 m	k_d= 3

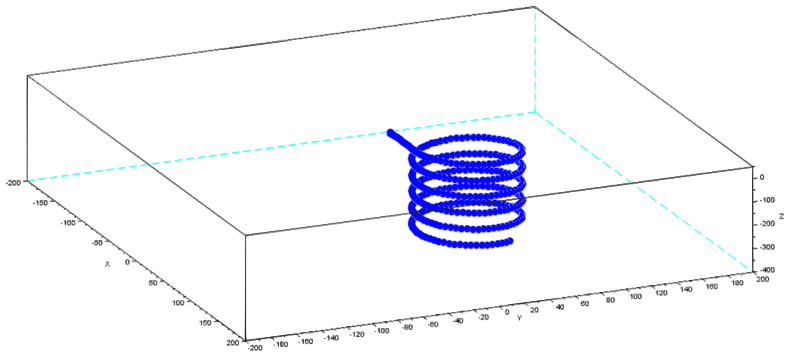


Figura 5.2: Trajetória 1

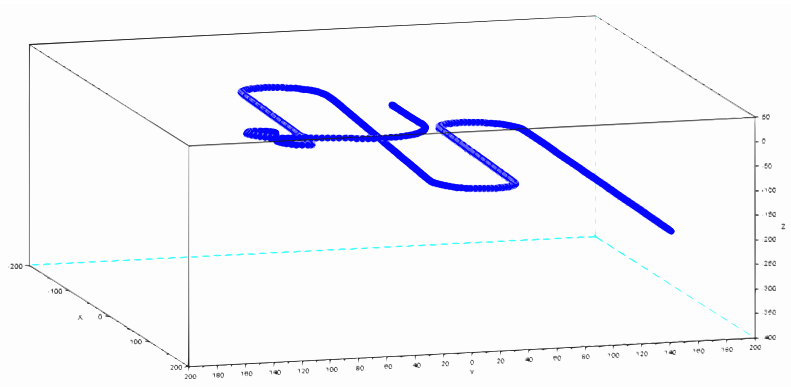


Figura 5.3: Trajetória 2

subaquáticas. A terceira trajetória, mostrada na Figura 5.4, visa alcançar determinados pontos do espaço. O robô vai de ponto a ponto, por exemplo, para coleta de amostras ou transporte de materiais.

Em suma, os testes são executados sobre 3 trajetórias diferentes, e cada uma considerando 3 ambientes diferentes. Sendo assim, 9 configurações de casos de teste são geradas para testar os métodos propostos e o filtro de partículas. Cada configuração foi executada 10 vezes, totalizando 90 testes com cada método. Em todos os testes a população de partículas usada possuia 5000 amostras.

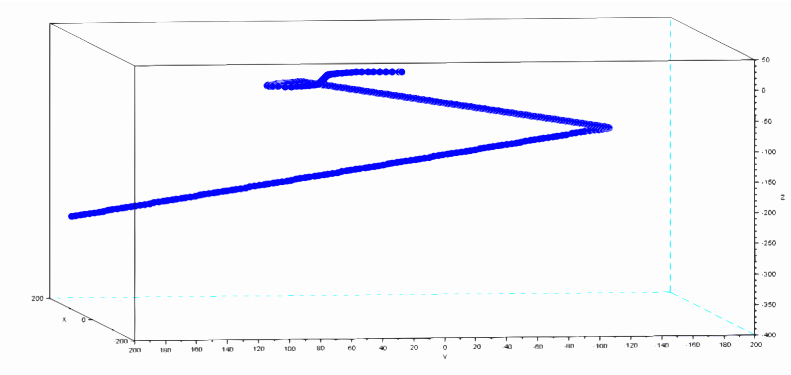


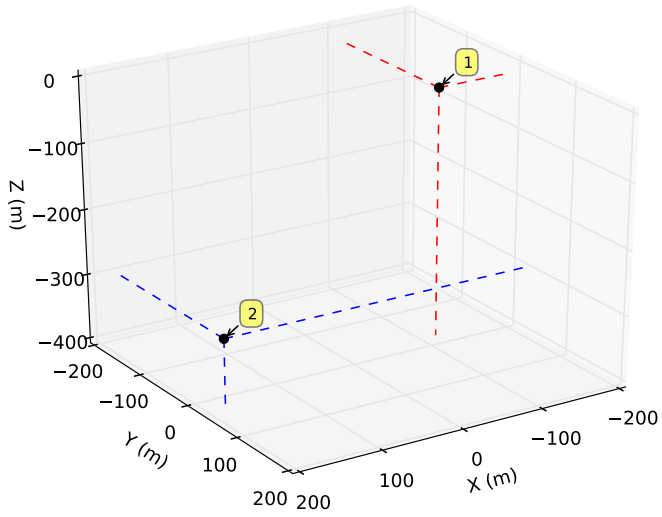
Figura 5.4: Trajetória 3

A fim de comparar o comportamento e a precisão dos métodos, nas próximas seções são apresentados alguns gráficos. Os gráficos de linha, mostram o comportamento do erro sobre a localização do robô no decorer das iterações. São apresentados a média das execuções (em preto) e o desvio padrão (em cinza) do erro entre a posição do robô dada pelos métodos testados (média ponderada das partículas) e a posição real do robô extraída da simulação.

Os gráficos de caixa, evidenciam a concentração dos valores de erro e a significativa diferença entre os métodos propostos e o filtro de partículas tradicional. Nos gráficos de caixa, o erro é representado no eixo vertical e as informações presentes nos gráficos são quartis e mediana. Para cada configuração testada, são mostrados a mediana (quadrado em preto) e os quartis superior e inferior dos valores de erro medidos durante os experimentos. Também são mostrados os valores máximos e mínimos de erro (traço "-" em preto), descontando *outliers*, que são valores afastados da região interquartil (retângulo em cinza) por uma distância maior que 1.5 vezes a amplitude interquartil.

5.1 Experimentos no Ambiente 1

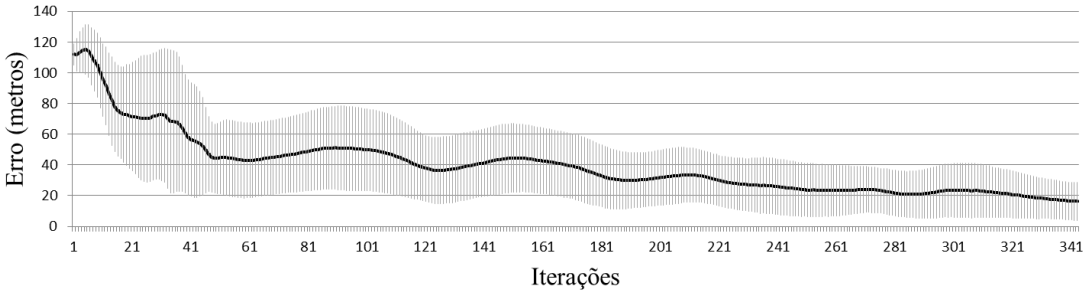
O primeiro ambiente possui apenas dois *transponders*, dispostos conforme mostra a figura 5.5. Por se tratar de um cenário com pouca informação, a abordagem intervalar não reduz muito o espaço de soluções, obtendo regiões grandes que são pouco significativas para o processo de localização. Entretando, essa redução do espaço de busca, mesmo que pequena, pode ajudar a acelerar a convergência do filtro de partículas.



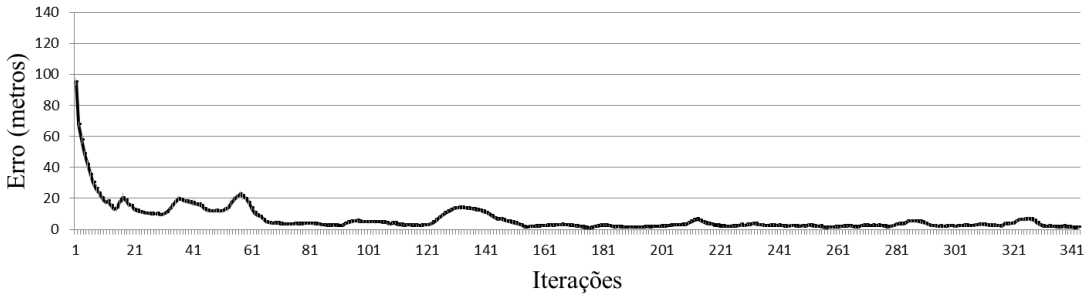
Transponders	
1	(-112; -10; -20)
2	(180; 30; -300)

Figura 5.5: Localização dos *transponders* no Ambiente 1

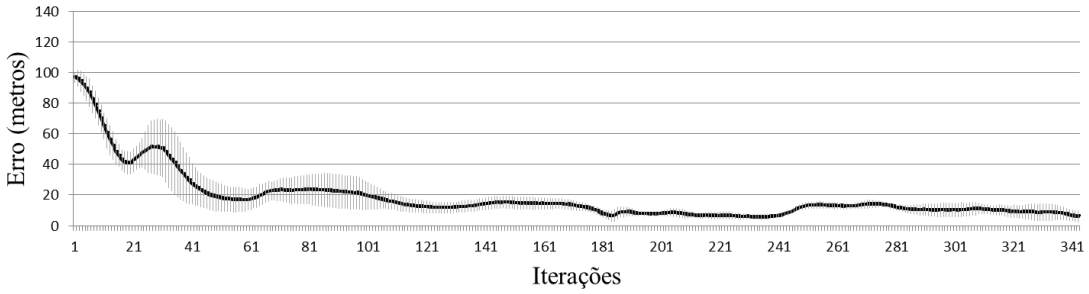
As Figuras 5.6, 5.7 e 5.8 mostram os erros associados aos métodos durante os testes com as três trajetórias propostas. Analisando os resultados das trajetórias 1 e 2, mostrados respectivamente nas Figuras 5.6 e 5.7, pode-se perceber que o erro médio cai mais



(a) Filtro de partículas.



(b) Híbrido (SIVIA).



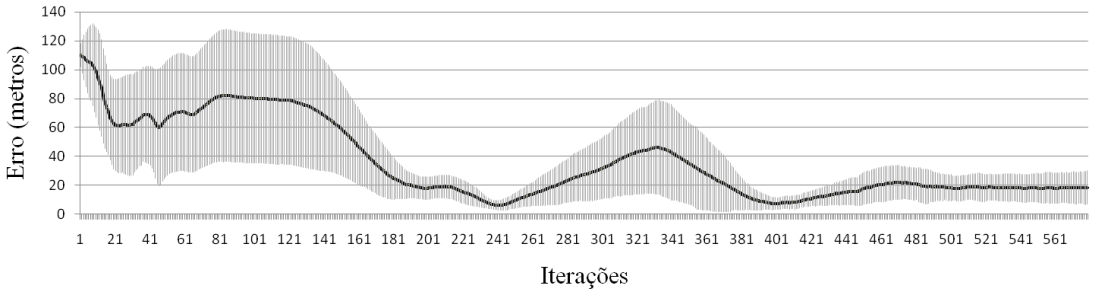
(c) Híbrido (contratores).

Figura 5.6: Erros - Trajetória 1 ambiente 1.

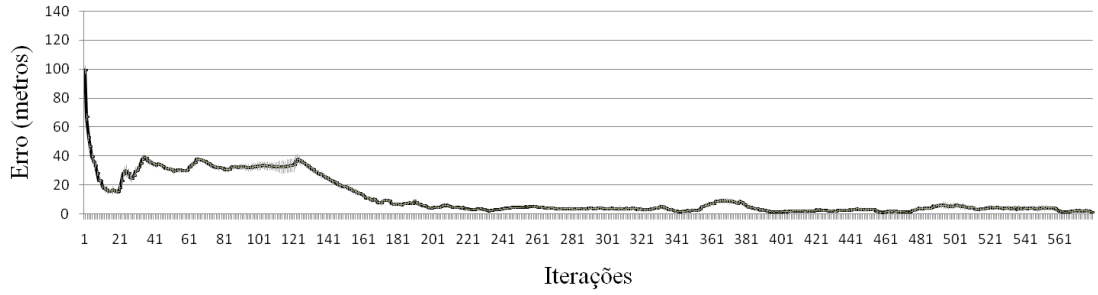
rapidamente nos gráficos referentes aos métodos híbridos do que nos gráficos referentes ao filtro de partículas. Além disso, os métodos híbridos apresentam uma grande redução no desvio padrão.

Os resultados encontrados nos testes executados no ambiente 1 com a trajetória 3 são apresentados na Figura 5.8. Para esse cenário de teste pode-se perceber que não houve melhoria na precisão dos resultados do método híbrido baseado no SIVIA. Este foi o único caso de teste onde o método híbrido baseado em contratores obteve melhores resultados que o método baseado no SIVIA.

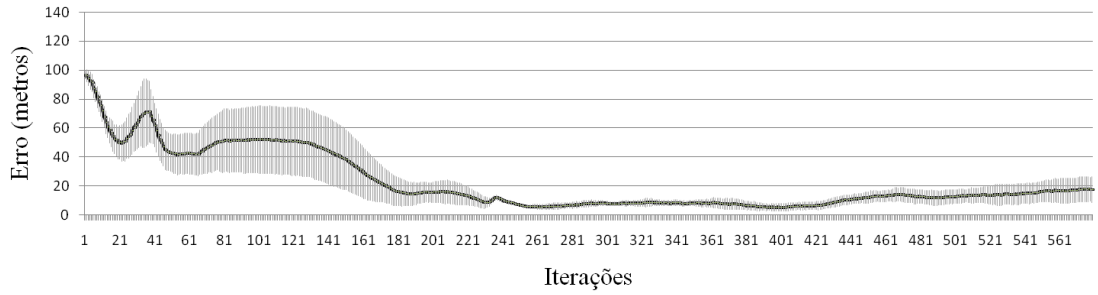
A ocorrência de tal discrepância parece resultar de uma escolha desafortunada da trajetória em relação ao posicionamento dos *transponders*. Conforme dito antes, esta configuração com apenas dois *transponders* não provê informação suficiente para uma localização precisa do robô. Este problema deixa de ocorrer nos outros dois ambientes, que possuem um número maior de *transponders*.



(a) Filtro de partículas.

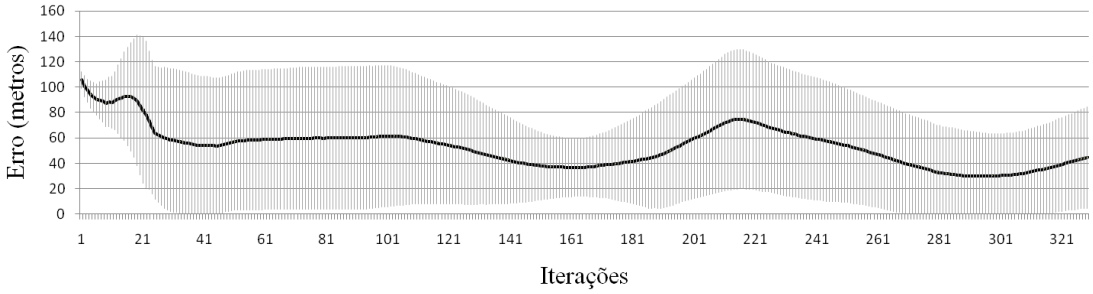


(b) Híbrido (SIVIA).

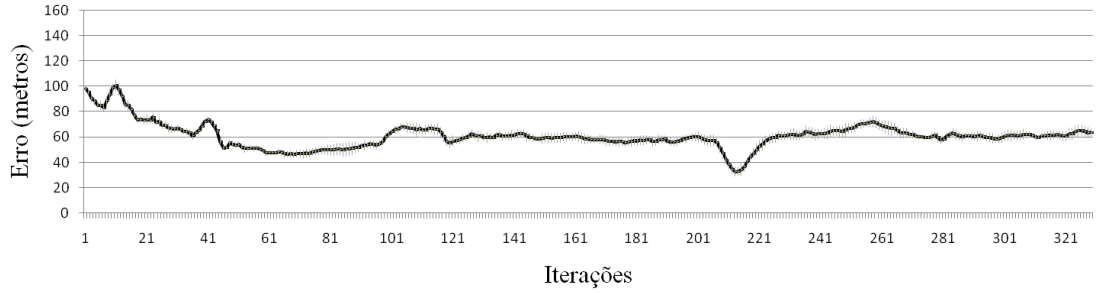


(c) Híbrido (contratores).

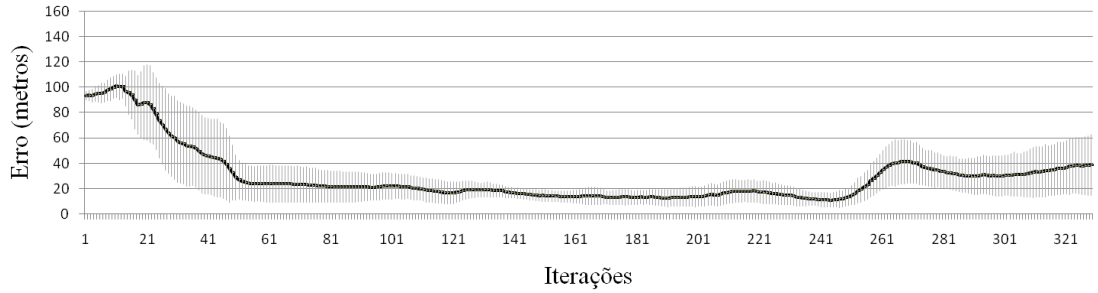
Figura 5.7: Erros - Trajetória 2 ambiente 1.



(a) Filtro de partículas.



(b) Híbrido (SIVIA).



(c) Híbrido (contratores).

Figura 5.8: Erros - Trajetória 3 ambiente 1.

Uma comparação compacta entre os resultados obtidos no ambiente 1 é apresentada na Figura 5.9 através de um diagrama de caixa. Podemos perceber a melhoria obtida com os métodos híbridos em comparação ao filtro de partículas puro, com a ressalva do desempenho do SIVIA na trajetória 3.

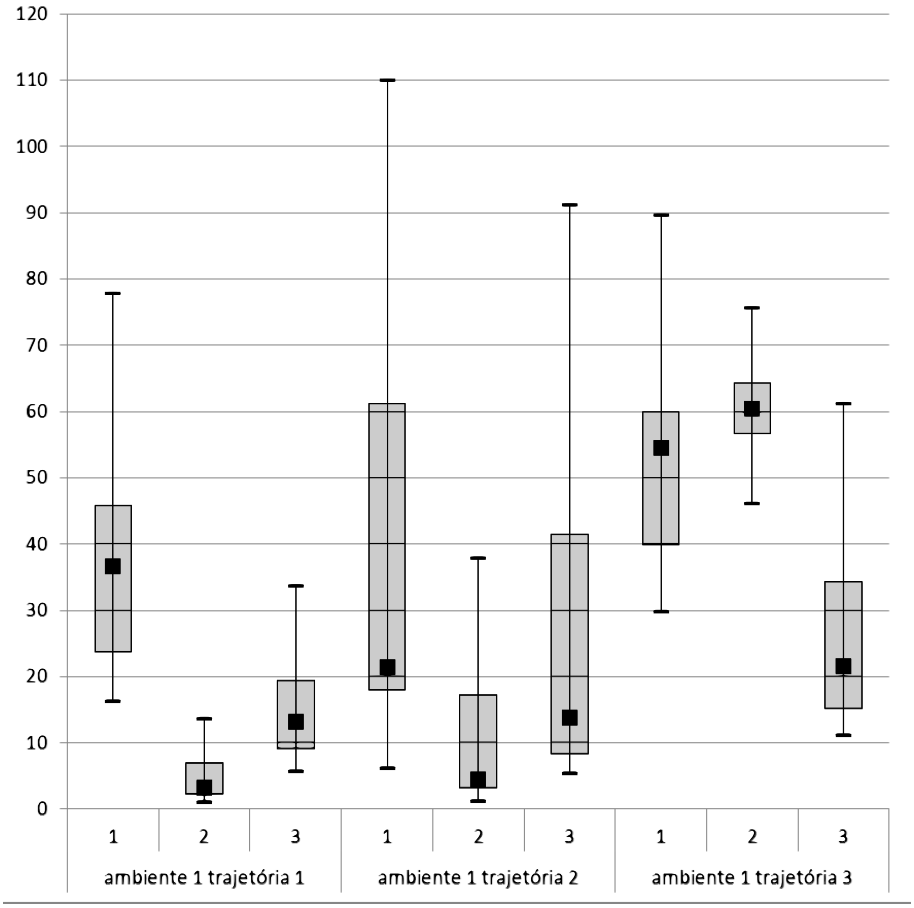
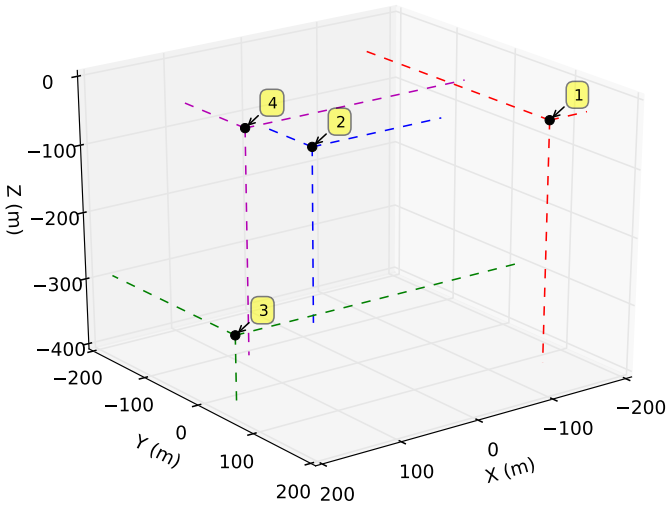


Figura 5.9: Diagrama de caixa para os dados do ambiente 1. Onde 1, 2 e 3 representam respectivamente os métodos filtro de partículas, método híbrido com sivia e método híbrido com contratores.

O gráfico da Figura 5.9 mostra claramente o ganho significativo em relação a precisão dos métodos híbridos nos testes feitos no primeiro ambiente. Considerando os erros máximos, o filtro de partículas alcança valores como 77.7, 110.0 e 89.5, para trajetórias 1, 2 e 3 respectivamente. Em contrapartida, os métodos híbridos apresentam valores como 13.5, 37.8 e 75.6 para o método híbrido com SIVIA e 33.7, 91.2 e 61.2 para o método híbrido com contratores. Avaliando as medianas, o filtro de partículas apresenta valores como 36.6, 21.3 e 54.4, para trajetórias 1, 2 e 3 respectivamente. Enquanto métodos híbridos apresentam valores como 13.5, 37.8 e 75.6 para o método híbrido com SIVIA e 3.2, 4.4 e 60.4 para o método híbrido com contratores.

5.2 Experimentos no Ambiente 2

O segundo ambiente possui 4 *transponders*, dispostos conforme mostra a Figura 5.10. Sendo assim, neste caso de teste é disponibilizado o dobro de informações dos primeiros



Transponders	
1	(-150; 130; -40)
2	(-25; -115; -130)
3	(180; 30; -300)
4	(90; -75; -60)

Figura 5.10: Localização dos *transponders* no Ambiente 2

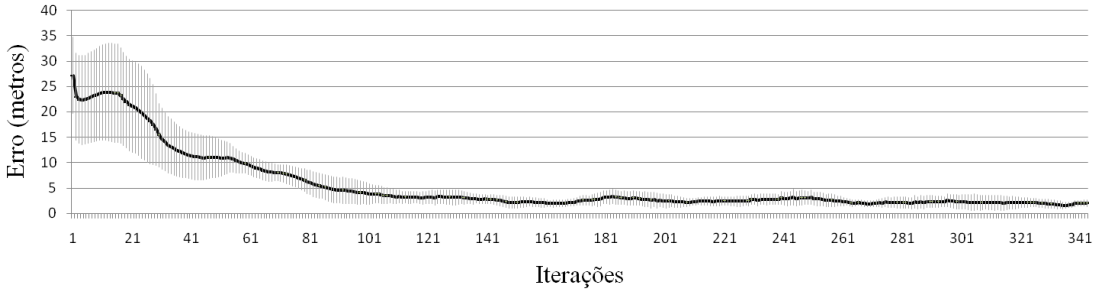
testes. Com tal aumento, já se torna possível observar diferenças mais acentuadas nos resultados obtidos.

Nas Figuras 5.11, 5.13 e 5.15 são apresentados os erros de localização encontrados nos testes executados no segundo ambiente. Ambos os métodos híbridos mostraram grande ganho quanto à precisão do resultado encontrado quando comparados com o filtro de partículas tradicional. Comparando as abordagens híbridas entre sí, fica claro que a abordagem que utiliza o SIVIA apresenta uma solução mais precisa, sendo que o erro sobre a localização do robô não ultrapassa 0.5 metros em todas as trajetórias testadas.

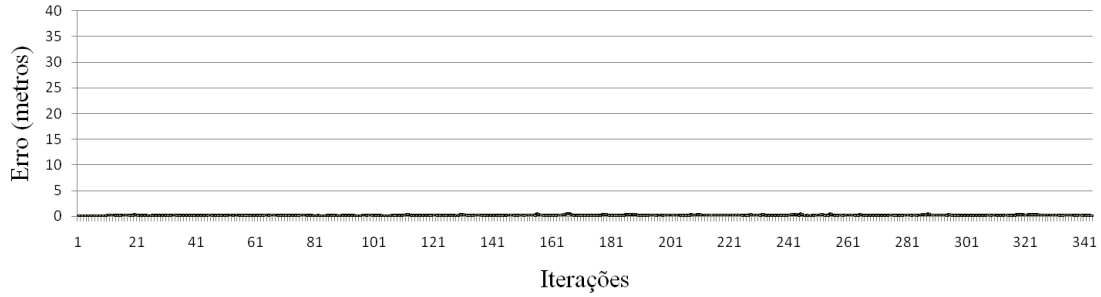
O diagrama de caixa para os resultados obtidos no ambiente 2 é apresentado na Figura 5.17. Se compararmos com o diagrama de caixas do ambiente 1 (Figura 5.9) veremos que a precisão aumentou consistentemente em todos os métodos. Além disso, a superioridade dos dois métodos híbridos se mostra evidente neste ambiente.

Visto que a escala usada para comparação dos métodos apresentados nas Figuras 5.11, 5.13 e 5.15 dificulta a visualização do comportamento do erro para os métodos híbridos, as Figuras 5.12, 5.14 e 5.16 apresentam as mesmas informações sob uma escala diferenciada.

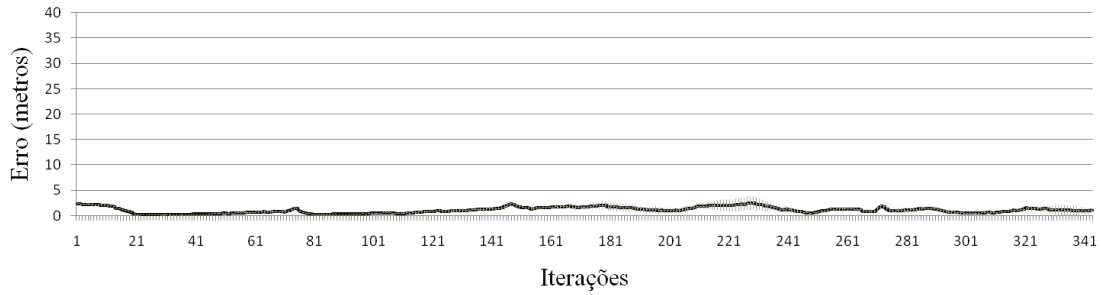
O gráfico da Figura 5.17 mostra claramente o ganho significativo em relação a precisão dos métodos híbridos. Considerando os erros máximos, o filtro de partículas alcança valores como 9.3, 9.5 e 26.1, para trajetórias 1, 2 e 3 respectivamente. Em contrapartida, os métodos híbridos apresentam valores como 0.29, 0.32 e 0.37 para o método híbrido com SIVIA e 2.4, 2.7 e 4.3 para o método híbrido com contratores. Avaliando as medianas, o filtro de partículas apresenta valores como 2.7, 3.3 e 5.9, para trajetórias 1, 2 e 3 respectivamente. Enquanto métodos híbridos apresentam valores como 0.16, 0.13 e 0.14 para o método híbrido com SIVIA e 1.03, 0.61 e 1.54 para o método híbrido com contratores.



(a) Filtro de partículas.

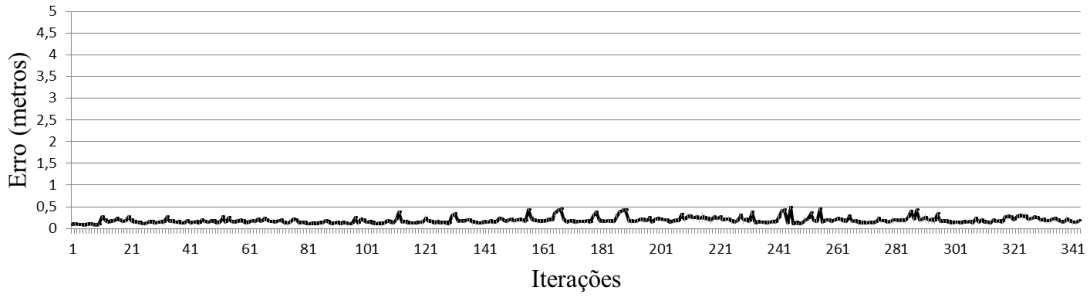


(b) Híbrido (SIVIA).

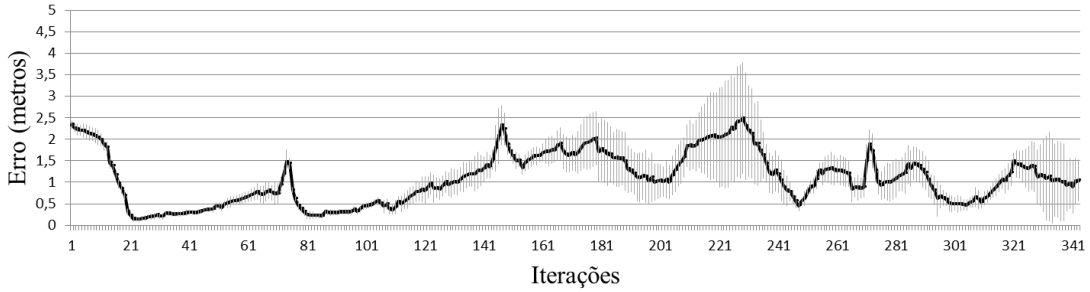


(c) Híbrido (contratores).

Figura 5.11: Erros - Trajetória 1 ambiente 2.

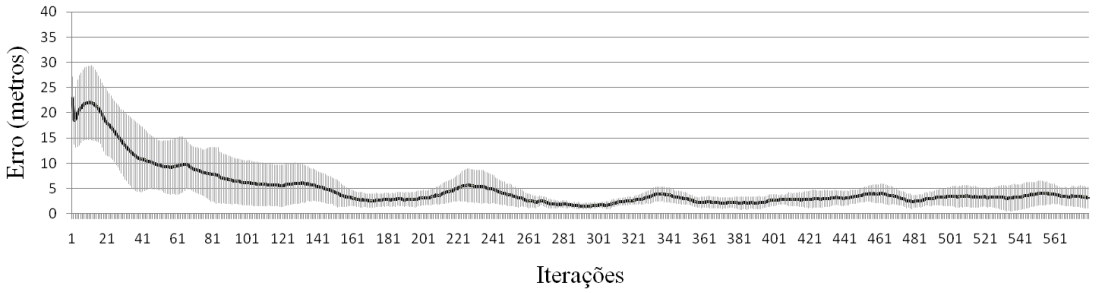


(a) Híbrido (SIVIA).



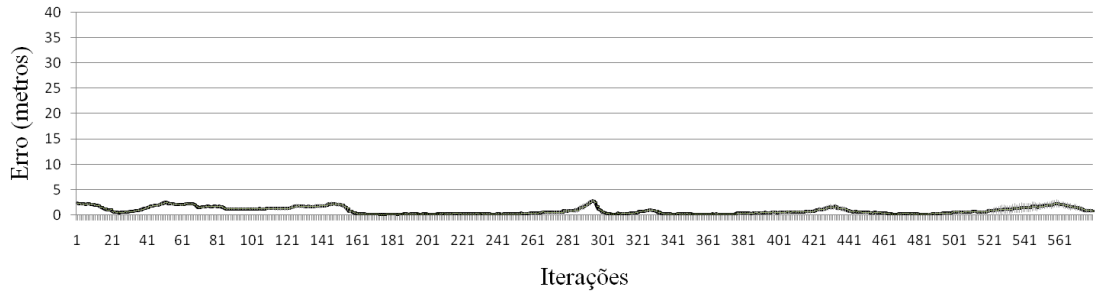
(b) Híbrido (contratores).

Figura 5.12: Erros - Trajetória 1 ambiente 2 - escala diferenciada.



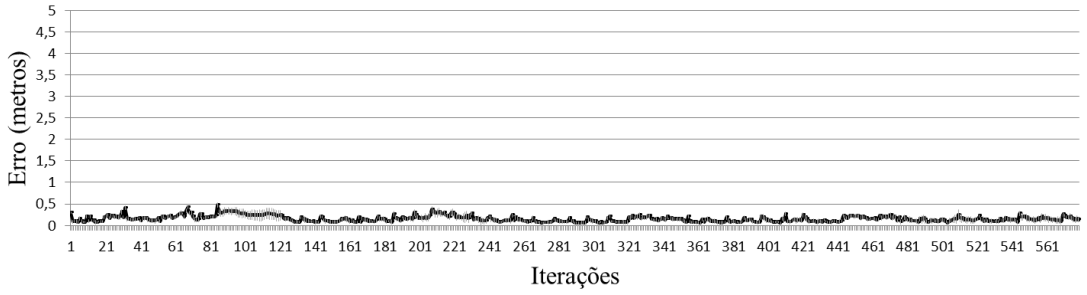
(a) Filtro de partículas.

(b) Híbrido (SIVIA).

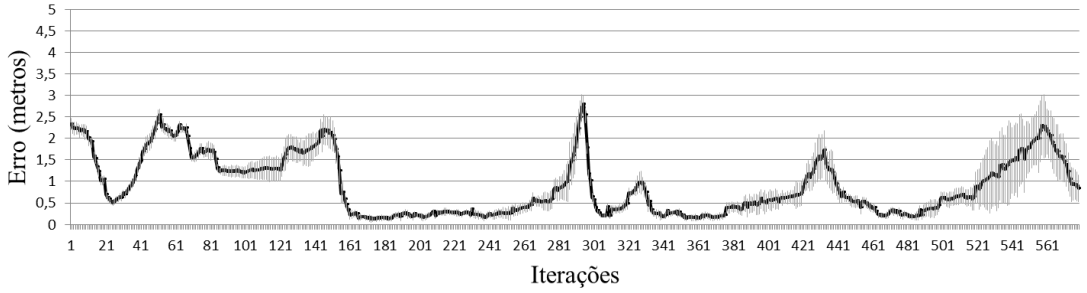


(c) Híbrido (contratores).

Figura 5.13: Erros - Trajetória 2 ambiente 2.



(a) Híbrido (SIVIA).

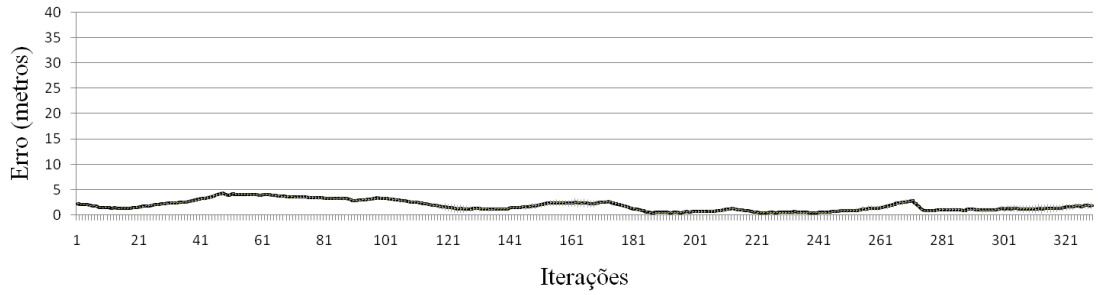


(b) Híbrido (contratores).

Figura 5.14: Erros - Trajetória 2 ambiente 2 - escala diferenciada.

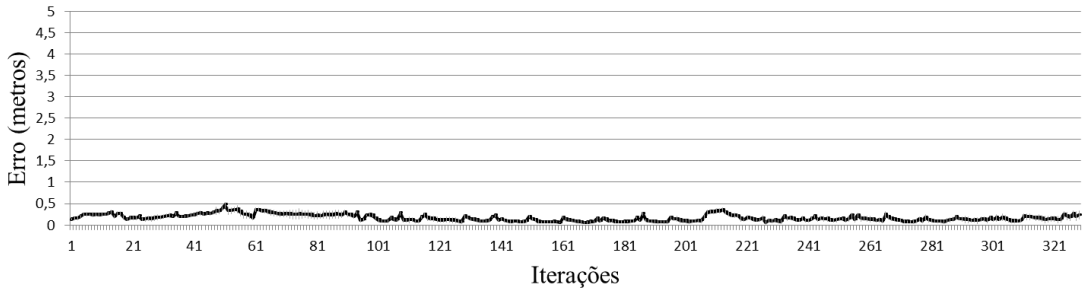
(a) Filtro de partículas.

(b) Híbrido (SIVIA).

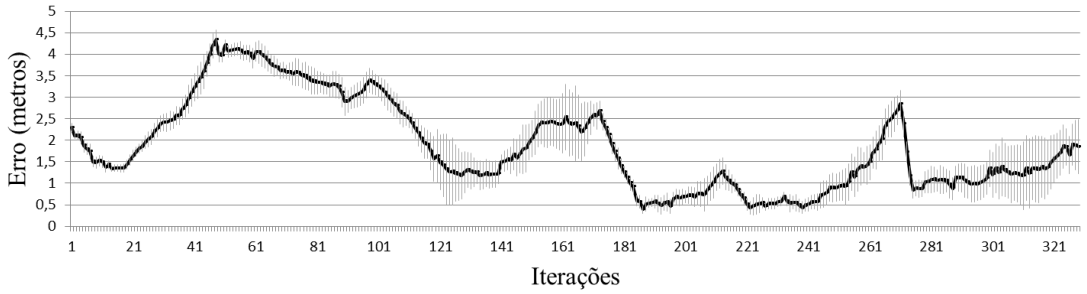


(c) Híbrido (contratores).

Figura 5.15: Erros - Trajetória 3 ambiente 2.



(a) Híbrido (SIVIA).



(b) Híbrido (contratores).

Figura 5.16: Erros - Trajetória 3 ambiente 2 - escala diferenciada.

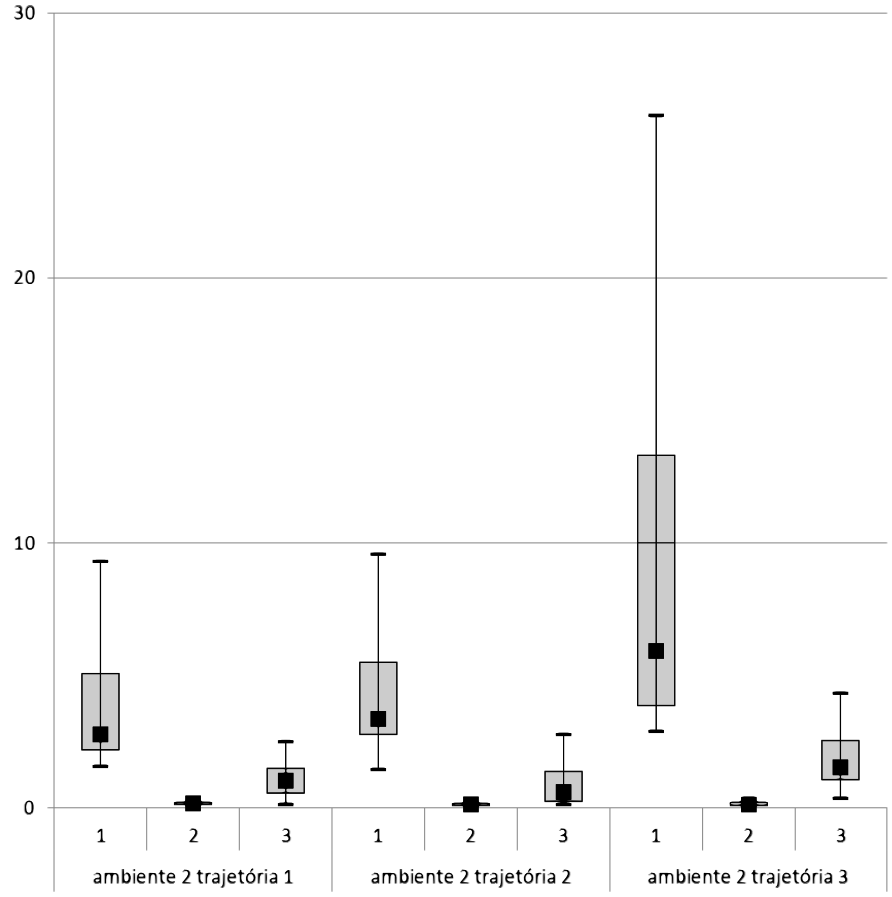
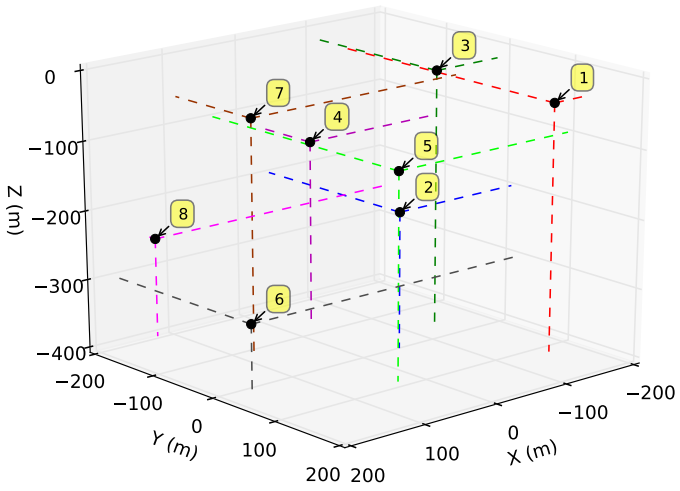


Figura 5.17: Diagrama de caixa para os dados do ambiente 2. Onde 1, 2 e 3 representam respectivamente os métodos filtro de partículas, método híbrido com sivia e método híbrido com contratores.

5.3 Experimentos no Ambiente 3

O terceiro ambiente contém 8 *transponders*, dispostos conforme mostra a Figura 5.18. Os resultados dos erros no processo de localização para os três métodos segundo as trajetórias 1, 2 e 3, são apresentados respectivamente nas Figuras 5.19, 5.21 e 5.23. Como é possível observar, os resultados obtidos nesse ambiente são muito semelhantes aos resultados dos testes no segundo ambiente. Isso pode ser explicado pelo fato de que, o aumento do número de *transponders* do ambiente 1 para o ambiente 2 implicou em ganho significativo de informação, enquanto que do ambiente 2 para o ambiente 3 implicou basicamente em redundância. De qualquer forma, os métodos híbridos mostraram resultados mais precisos que o filtro de partículas tradicional. E, mais uma vez, o método utilizando SIVIA mostrou superioridade sobre o método utilizando contratores.



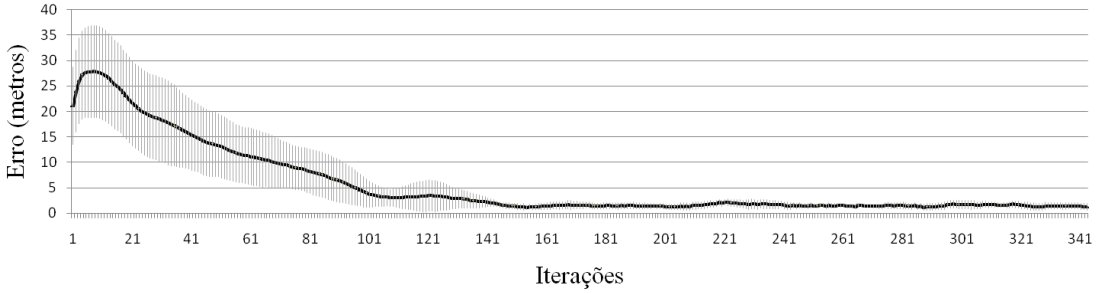
Transponders	
1	(-150; 130; -40)
2	(-38; 15; -200)
3	(-112; -10; -20)
4	(-25; -115; -130)
5	(40; 100; -100)
6	(180; 30; -300)
7	(90; -75; -60)
8	(130; -190; -255)

Figura 5.18: Localização dos *transponders* no Ambiente 3

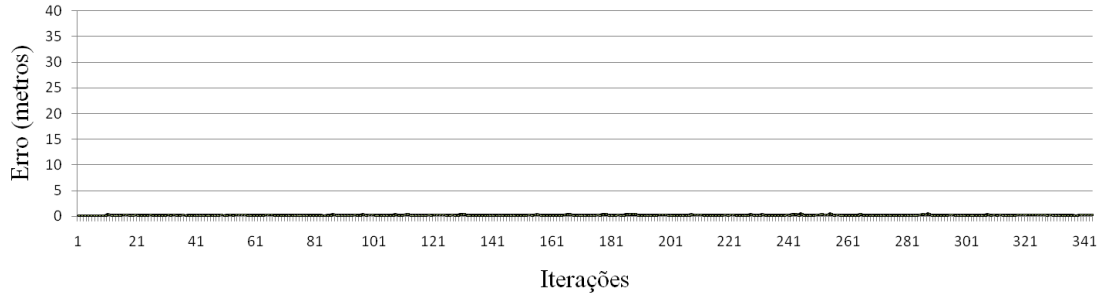
Visto que a escala usada para comparação dos métodos apresentados nas Figuras 5.19, 5.21 e 5.23 dificulta a visualização do comportamento do erro para os métodos híbridos, as Figuras 5.20, 5.22 e 5.24 apresentam as mesmas informações sob uma escala diferenciada.

O diagrama de caixa para os resultados obtidos no ambiente 3 é apresentado na Figura 5.25. Apesar de uma leve melhoria em praticamente todos os experimentos, os resultados ficaram semelhantes aos obtidos no ambiente 2.

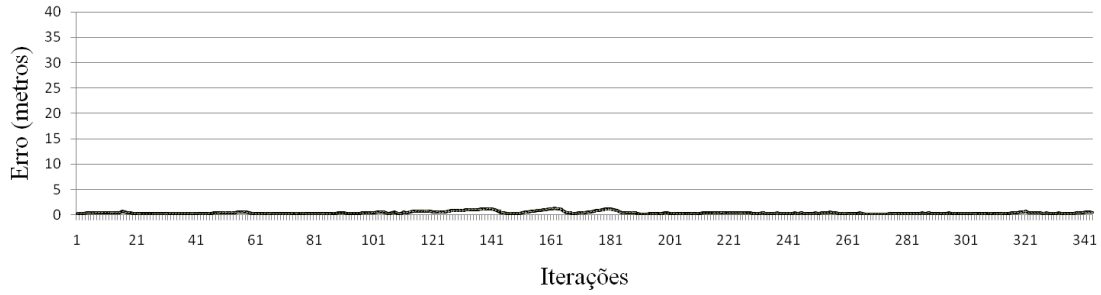
O gráfico da Figura 5.25 mostra claramente o ganho significativo em relação a precisão dos métodos híbridos. Considerando os erros máximos, o filtro de partículas alcança valores como 15.6, 4.5 e 31.3, para trajetórias 1, 2 e 3 respectivamente. Em contrapartida, os métodos híbridos apresentam valores como 0.24, 0.24 e 0.29 para o método híbrido com SIVIA e 0.71, 0.87 e 1.32 para o método híbrido com contratores. Avaliando as medianas, o filtro de partículas apresenta valores como 1.7, 2.3 e 7.2, para trajetórias 1, 2 e 3 respectivamente. Enquanto métodos híbridos apresentam valores como 0.14, 0.105 e 0.104 para o método híbrido com SIVIA e 0.30, 0.27 e 0.32 para o método híbrido com



(a) Filtro de partículas.

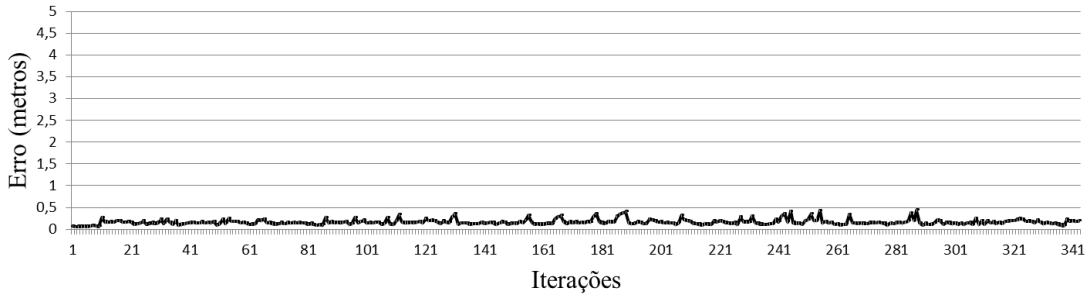


(b) Híbrido (SIVIA).

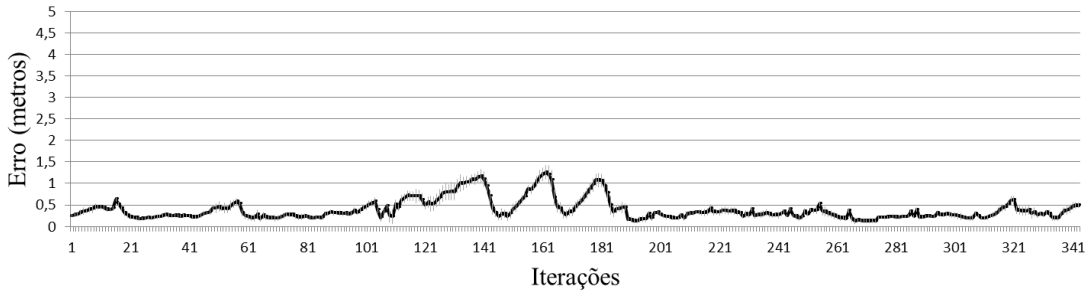


(c) Híbrido (contratores).

Figura 5.19: Erros - Trajetória 1 ambiente 3.



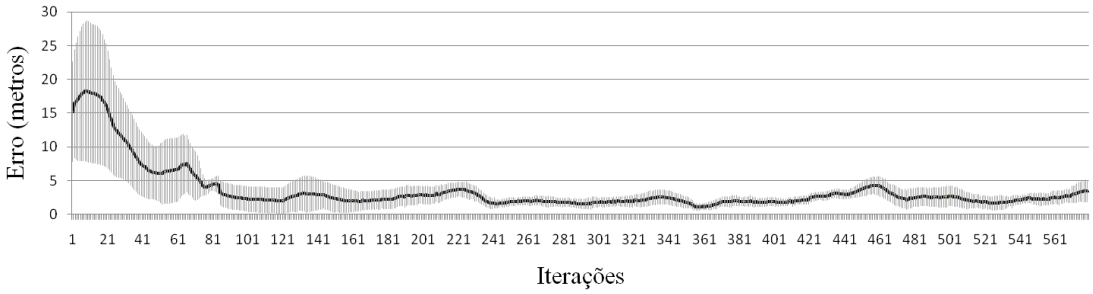
(a) Híbrido (SIVIA).



(b) Híbrido (contratores).

Figura 5.20: Erros - Trajetória 1 ambiente 3 - escala diferenciada.

contratores.

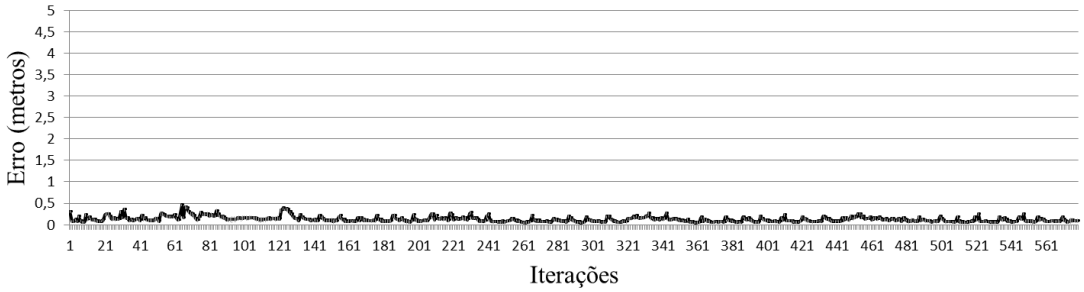


(a) Filtro de partículas.

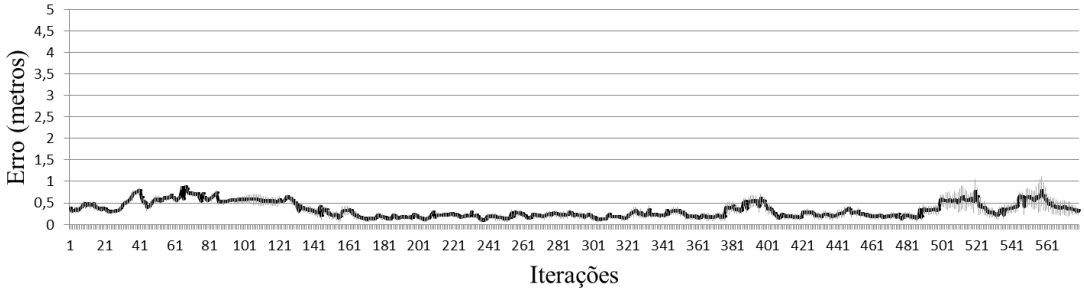
(b) Híbrido (SIVIA).

(c) Híbrido (contratores).

Figura 5.21: Erros - Trajetória 2 ambiente 3.

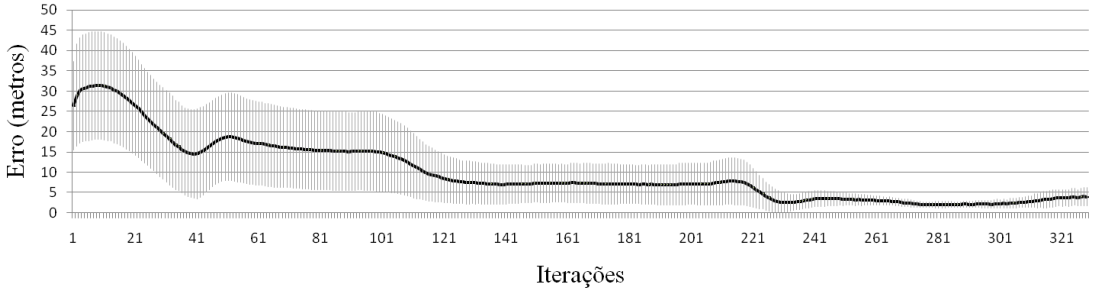


(a) Híbrido (SIVIA).

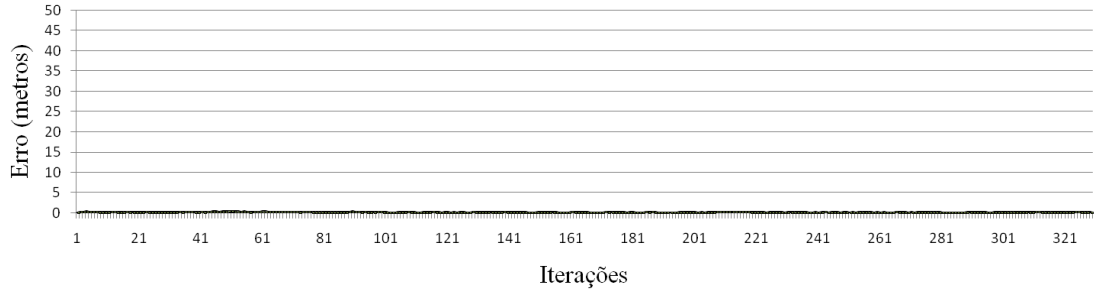


(b) Híbrido (contratores).

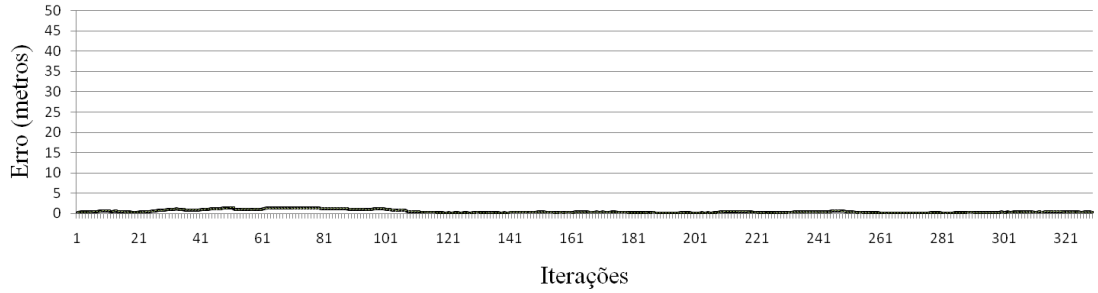
Figura 5.22: Erros - Trajetória 2 ambiente 3 - escala diferenciada.



(a) Filtro de partículas.

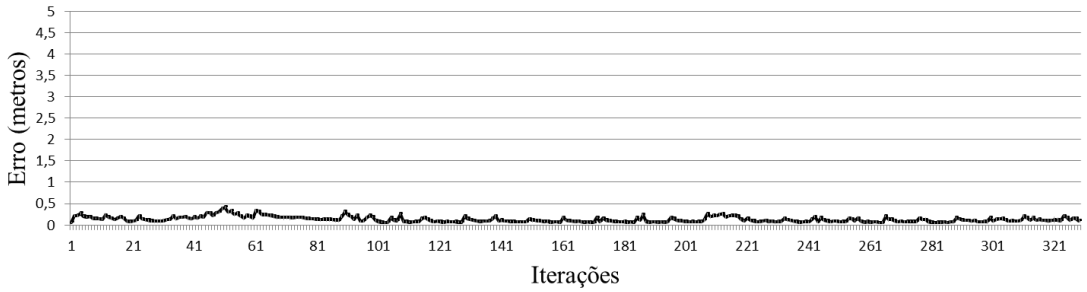


(b) Híbrido (SIVIA).

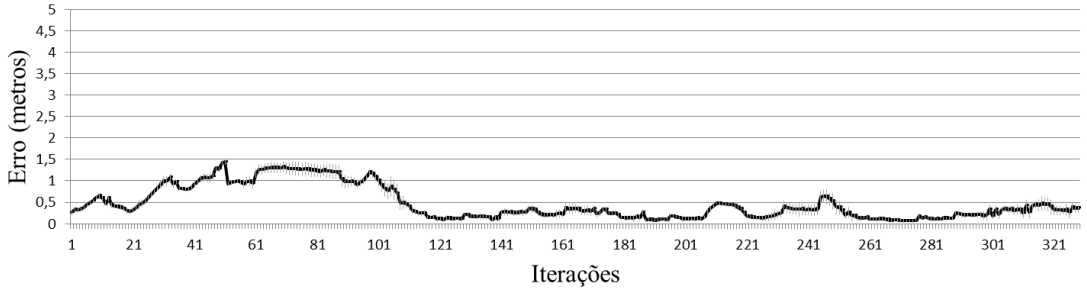


(c) Híbrido (contratores).

Figura 5.23: Erros - Trajetória 3 ambiente 3.



(a) Híbrido (SIVIA).



(b) Híbrido (contratores).

Figura 5.24: Erros - Trajetória 3 ambiente 3 - escala diferenciada.

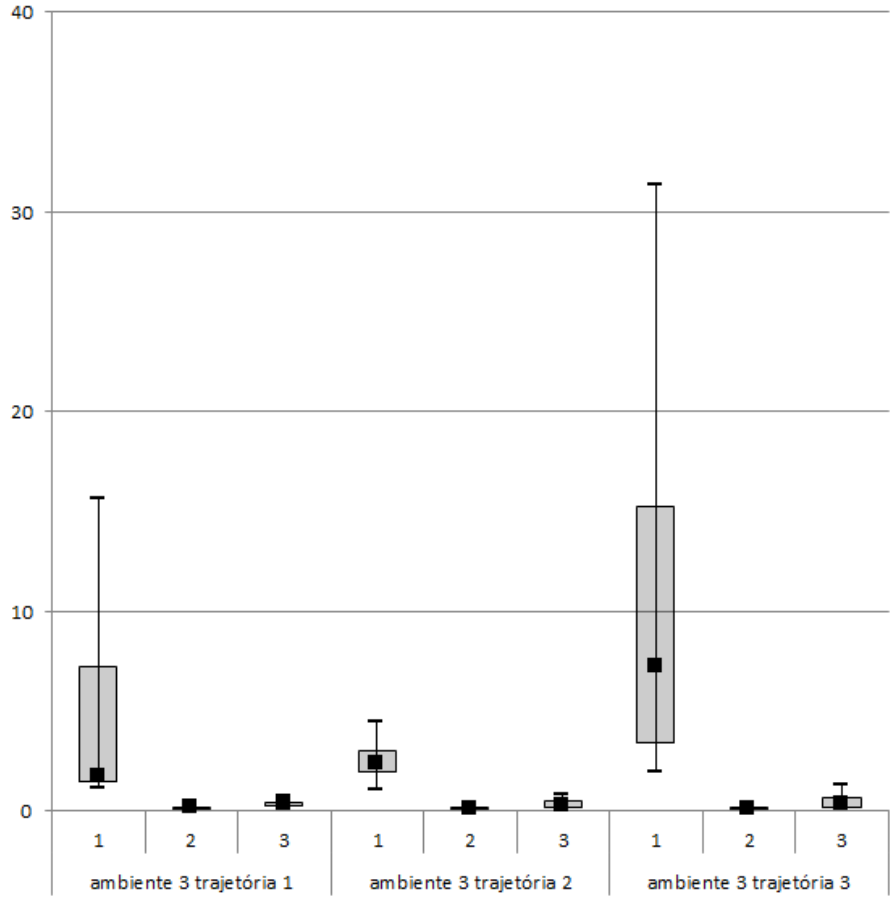


Figura 5.25: Diagrama de caixa para os dados do ambiente 3. Onde 1, 2 e 3 representam respectivamente os métodos filtro de partículas, método híbrido com sivia e método híbrido com contratores.

5.4 Discussão

É importante destacar que ao analisar os gráficos apresentados neste capítulo seja dada devida atenção às escalas dos mesmos. Uma vez que os resultados são bastante diferentes, os gráficos foram adaptados para melhor acomodar as informações. De modo geral, os métodos híbridos mostraram resultados mais precisos que o filtro de partículas. Esse ganho na precisão é bastante significativo e se torna mais evidente quando há mais informações disponíveis para o processo de localização. Comparando os gráficos do ambiente 1 com os gráficos do ambiente 3, por exemplo, é possível observar que o aumento do número de *transponders* melhorou significativamente os resultados gerados. Entretanto, tal como foi observado nos testes, há um limite para acrescentar *transponders* no ambiente e gerar melhorias nos resultados. Em contrapartida, o uso de um número insuficiente de *transponders* prejudica o potencial do método. Este número varia de acordo com o tamanho do ambiente e o alcance dos *transponders*.

É importante salientar que a posição dos *transponders* também pode impactar no resultado do método. Não existe um padrão predefinido, entretanto é preferível que os *transponders* fiquem afastados uns dos outros e cubram o ambiente o mais uniformemente possível. Lembrando que dependendo do tipo de ambiente tratado, pode existir obstrução de sinal. Nos testes apresentados nessa pesquisa esse problema não é considerado, uma vez que não existem obstáculos e todos os *transponders* são visíveis de qualquer ponto do ambiente.

Além do ganho significativo na precisão, outra característica evidenciada nos resultados apresentados é o desvio padrão reduzido dos métodos híbridos em relação ao filtro de partículas. Isso ocorre principalmente devido a limitação no espalhamento das partículas definida pela análise de intervalos.

Os métodos híbridos prospostos apresentaram, de modo geral, resultados mais precisos que o filtro de partículas tradicional, considerando a média de erros encontrados durante a execução. Dentre os métodos híbridos, o método que utiliza o SIVIA mostrou-se mais apto a encontrar resultados mais precisos. Isso é esperado, já que a região delimitada é normalmente menor que a encontrada pelos contratores. O teste envolvendo a trajetoria 3 no primeiro ambiente é uma exceção, tendo sido o único caso de teste onde o método híbrido baseado em contratores obteve melhores resultados que o método baseado no SIVIA. Neste caso, o método com contratores mostrou resultados mais precisos, entretanto o tamanho da área de incerteza definida pelos contratores é maior ou igual à área encontrada pelo SIVIA. Assim, essa diferença entre os métodos híbridos é devido ao comportamento do filtro de partículas.

O resultado apresentado na Figura 5.8 evidencia que, apesar da redução da área de incerteza, não existe uma garantia de que o resultado do método híbrido será melhor que o resultado do filtro de partículas. Existe apenas uma delimitação da incerteza. Essa delimitação geralmente aumenta a precisão dos resultados. Entretanto, as partículas ainda podem convergir para um lugar errado dentro da região delimitada. Conforme mostrado na seção 3.1.1, o filtro de partículas é suscetível a problemas de convergência. Se há pouca informação presente no ambiente, como é o caso do ambiente 1, a pesagem das partículas pode não ser muito adequada. Isto gera um problema na etapa de reamostragem, podendo amplificar a convergência para soluções boas localmente, mas erradas globalmente. Como cada um dos métodos testados geram espaços de busca de tamanhos

	Contratores	**SIVIA**
ambiente 1 trajetória 1	4,27%	3489,29%
ambiente 1 trajetória 2	3,92%	2385,10%
ambiente 1 trajetória 3	4,45%	7628,61%
ambiente 2 trajetória 1	6,90%	111,31%
ambiente 2 trajetória 2	5,78%	108,77%
ambiente 2 trajetória 3	5,81%	112,91%
ambiente 3 trajetória 1	9,11%	85,27%
ambiente 3 trajetória 2	9,22%	88,96%
ambiente 3 trajetória 3	9,62%	87,96%

Tabela 5.2: Tempo de execução extra

distintos, a distribuição das partículas é diferente para cada um deles. Logo, a convergência do filtro de partículas pode ocorrer (se ocorrer) em momentos diferentes para cada método. Foi provavelmente essa fraqueza do filtro de partículas em ambientes com pouca informação, somada com a pequena redução da incerteza, que favoreceu o surgimento de resultados anômalos nessa configuração específica de trajetória e *transponders*.

Como visto até aqui, os métodos híbridos propostos neste trabalho apresentaram ganhos substanciais quanto a precisão de localização. Como exemplo disso pode-se citar o resultado apresentado na Figura 5.23, nessa comparação o filtro de partículas apresenta erros mínimo e máximo de 1.98 e 31.3 respectivamente, sendo o erro ao final do processo de 3.97. Esses erros são superiores aos do método híbrido com SIVIA, o qual apresentou erros mínimo e máximo de 0.41 e 0.04, sendo o erro ao final do processo de 0.09. Mesmo com erros superiores aos do método que utiliza SIVIA, o método híbrido com contratores também apresentou erros muito menores que o FP, sendo 0.06 e 1.43 os erros mínimo e máximo e 0.35 o erro apontado ao final do processo.

Contudo, é preciso levar em consideração o aumento no custo computacional resultante da inclusão da análise intervalar durante o processamento. A Tabela 5.2 mostra o percentual de tempo extra necessário para a execução dos testes de cada um dos métodos híbridos em comparação ao filtro de partículas. Percebe-se que o método híbrido usando SIVIA, apesar de obter os melhores resultados de localização, introduz um *overhead* muito grande no processamento. Isto pode ser visto especialmente no ambiente 1, onde a área de incerteza é muito grande devido à pouca informação no ambiente. Por outro lado, o método com contratores também mostra melhoria significante de localização, mesmo que não seja tão boa quanto a do método utilizando SIVIA, mas com uma adição de tempo muito pequena comparado ao filtro de partículas. Portanto, concluimos que o uso do método híbrido usando contratores seria o mais adequado para o problema de localização.

Na Tabela 5.3 é apresentada uma comparação a nível qualitativo considerando aspectos de precisão do resultado, tempo gasto para execução do método, cobertura da região de incerteza e velocidade e qualidade na detecção de má convergencia do método.

	Filtro de Partículas	Filtro de Partículas+ SIVIA	Filtro de Partículas + Contratores
Precisão	−	++	+
Tempo		−−	
Cobertura da região de incerteza	−	++	+
Detecção de má convergência	−	++	++

Tabela 5.3: Avaliação qualitativa dos métodos testados

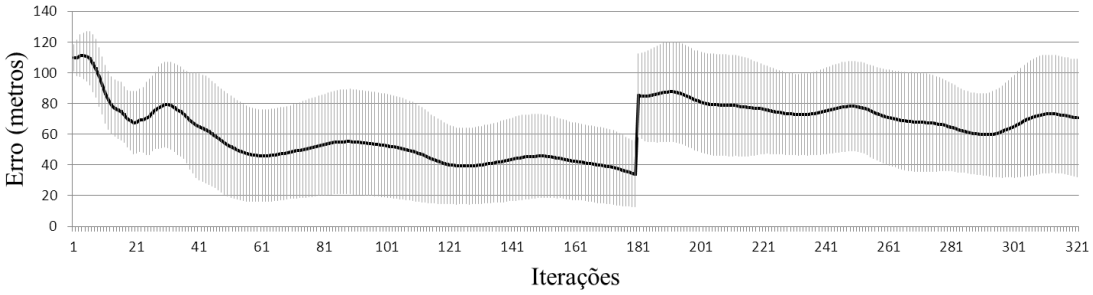
5.5 Experimentos extras: Problema do Robô Raptado

Após o sucesso nos experimentos com os métodos propostos aplicados ao problema de localização global, foram feitos testes utilizando os mesmos métodos para o problema do robô raptado. Os parâmetros foram mantidos e as trajetórias levemente alteradas, gerando um deslocamento inesperado na posição do robô.

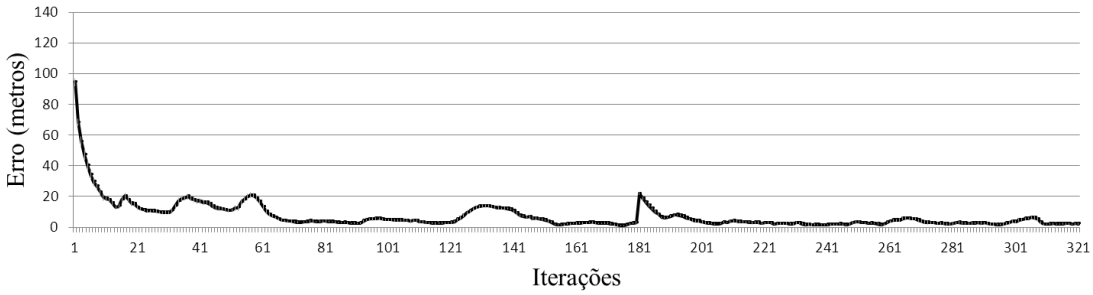
Os métodos híbridos, por usarem análise intervalar, conseguem identificar e se recuperar com facilidade quando o robô é raptado. Se o robô é retirado da caixa (ou conjunto de caixas) em que estava situado, a próxima iteração do método de análise intervalar resulta em um conjunto vazio. Quando isso ocorre, o espaço de busca é expandido de volta para o seu tamanho original, pois deve haver a garantia de que a solução está contida nele, e as partículas são redistribuídas de acordo com os novos limites definidos através da análise de intervalos (SIVIA ou contratores). Após encontrar a região que engloba a nova localização do robô, o método prossegue naturalmente como um problema de localização global.

Já o filtro de partículas tradicional não é capaz de detectar, por si só, esse tipo de situação com facilidade. Quando o robô sofre um deslocamento repentino no ambiente, o que muda é o resultado da avaliação das partículas. Isso afeta a reamostragem, por exemplo, causando uma convergência rápida (se um grupo de partículas se destacar muito) ou uma replicação uniforme (se todas as partículas tiverem pesos parecidos). O problema é que as partículas continuarão restritas à região do espaço próxima de onde se encontravam e o erro não será corrigido. Logo, para tratar o problema do robô raptado é necessária a inclusão de um mecanismo externo ao filtro de partículas.

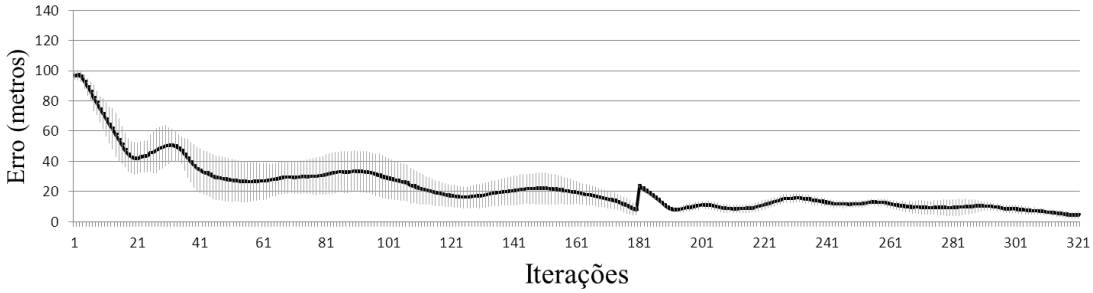
Após todos os testes, se percebeu um comportamento semelhante entre todas as trajetórias. Sendo assim, serão mostrados apenas os gráficos que apresentam o comportamento do erro sobre a primeira trajetória. A Figura 5.26 mostra os resultados no primeiro ambiente. Através desses gráficos é fácil perceber que o rápto ocorreu próximo a iteração 181, uma vez que os três métodos sofrem um aumento brusco no erro. Analisando as Figuras



(a) Filtro de partículas.



(b) Híbrido (SIVIA).



(c) Híbrido (contratores).

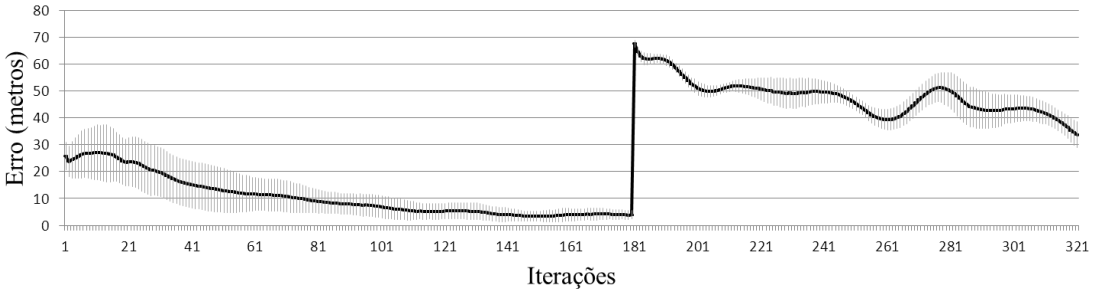
Figura 5.26: Erros - Trajetória 1 ambiente 1 - Problema do Robô Raptado.

5.26(b) e 5.26(c) pode-se perceber que os métodos híbridos rapidamente se recuperam após o rápto.

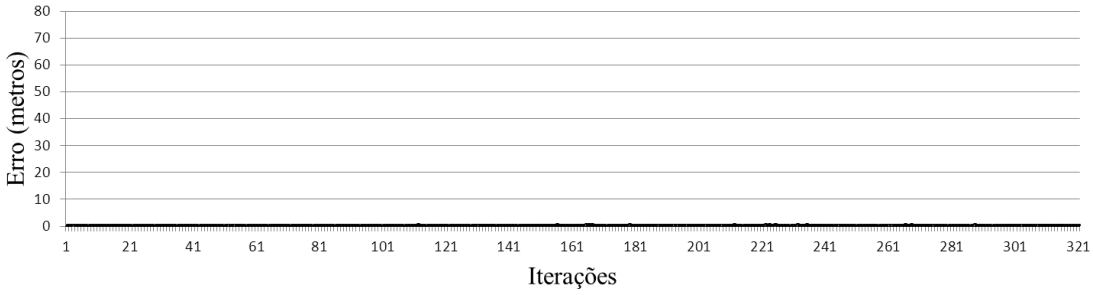
A Figura 5.27 apresenta os gráficos de erro encontrados nos testes no segundo ambiente. Com mais informações, os erros encontrados são menores que os encontrados no primeiro ambiente. Contudo, o filtro de partículas novamente não consegue se recuperar após o rápto.

Visto que a escala usada para comparação dos métodos apresentados na Figura 5.27 dificulta a visualização do comportamento do erro para os métodos híbridos, a Figura 5.28 apresenta as mesmas informações sob uma escala diferenciada. Nesses gráficos é possível ver o pico causado pelo rápto usando o método híbrido com contratores. Entretanto, no método que utiliza o SIVIA o rápto é quase imperceptível.

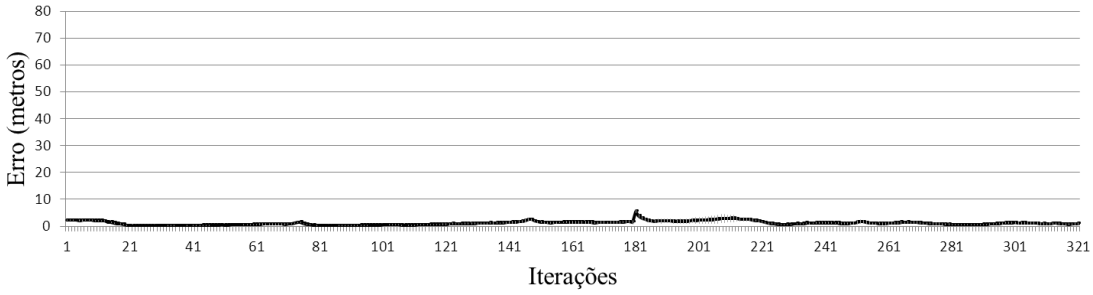
A Figura 5.29 apresenta o comportamento do erro para os diferentes métodos testados. Os resultados observados nesses gráficos são muito semelhantes aos gráficos do



(a) Filtro de partículas.

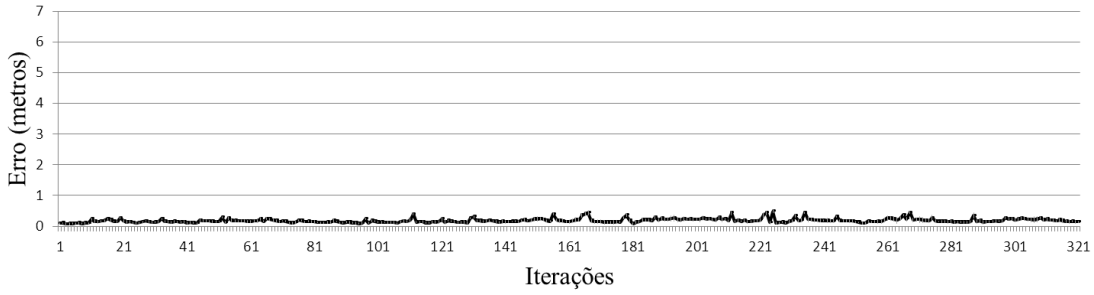


(b) Híbrido (SIVIA).

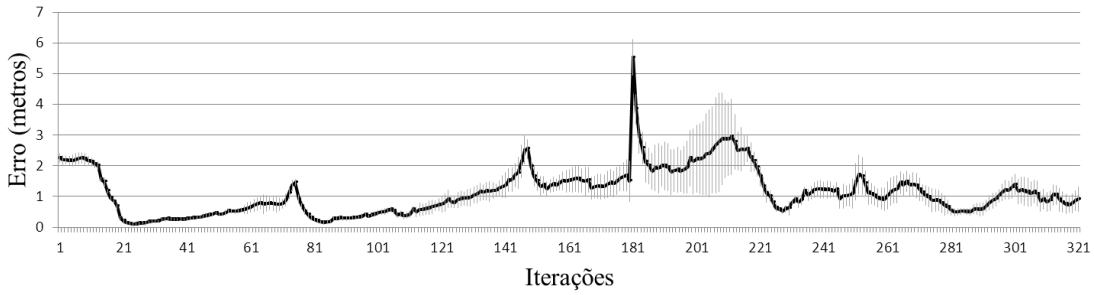


(c) Híbrido (contratores).

Figura 5.27: Erros - Trajetória 1 ambiente 2 - Problema do Robô Raptado.



(a) Híbrido (SIVIA).



(b) Híbrido (contratores).

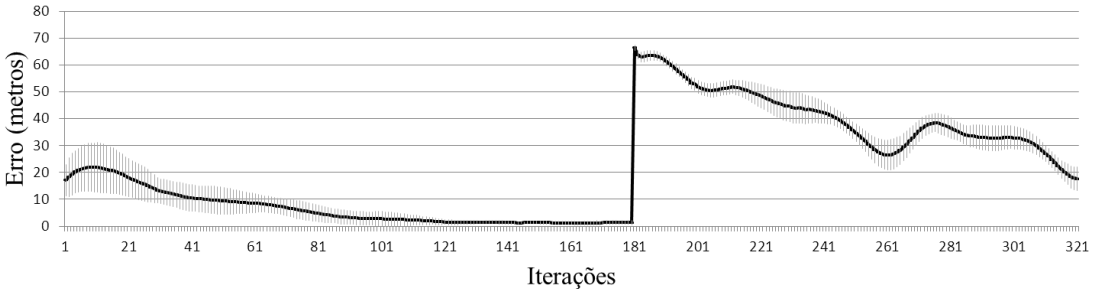
Figura 5.28: Erros - Trajetória 1 ambiente 2 - escala diferenciada - Problema do Robô Raptado.

ambiente anterior. Apesar do grande pico de erro apontado pelo FP, o momento do rapto é frequentemente imperceptível para quem observa o comportamento do erro durante a execução dos métodos híbridos. Avaliando por exemplo as informações contidas no gráfico da Figura 5.29, o erro apresentado pelo FP momentos antes do rapto estava próximo a 1.31 e instante do rapto alcançou um pico de 66.4. Considerando os mesmos instantes de tempo, o método híbrido com SIVIA apresentou respectivamente 0.17 e 0.10 e o método híbrido com contratores 0.89 e 0.11. O momento do rapto, neste exemplo, é imperceptível quando analisados os gráficos de erro dos métodos híbridos, sendo que, queda no erro dos métodos pode ter sido causada por uma correção na convergência das partículas dentro da região delimitada pela análise de intervalos (SIVIA e contratores).

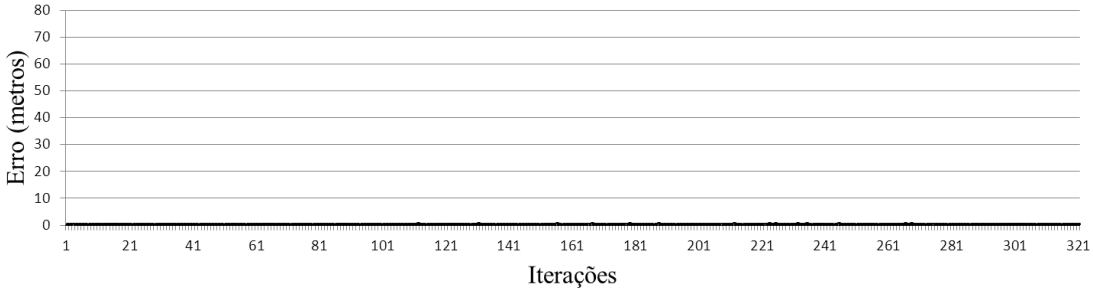
Visto que a escala usada para comparação dos métodos apresentados na Figura 5.29 dificulta a visualização do comportamento do erro para os métodos híbridos, a Figura 5.30 apresenta as mesmas informações sob uma escala diferenciada.

Os diagramas de caixa para os resultados obtidos nos três ambientes testados são apresentados nas Figuras 5.31, 5.32 e 5.33. Se compararmos os diagramas de caixa referentes ao problema do robô raptado, pode-se perceber resultados semelhantes aos obtidos para o problema de localização global. Onde os métodos híbridos se destacam com melhores resultados.

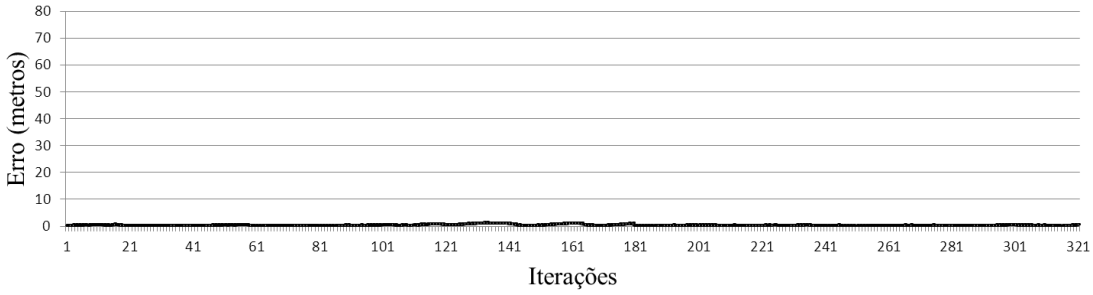
Em todos os ambientes de teste os métodos híbridos apresentaram valores de erro inferiores aos do filtro de partículas tradicional. Alguns dos resultados mais significantes podem ser vistos no gráfico da Figura 5.33. Os erros máximos do FP, no terceiro ambiente, alcançam valores como 66.4, 161.9 e 26.6 (para trajetórias 1, 2 e 3 respectivamente). Em



(a) Filtro de partículas.

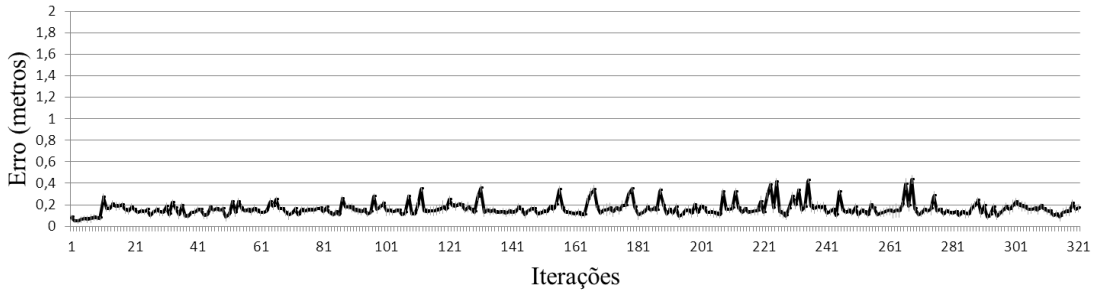


(b) Híbrido (SIVIA).

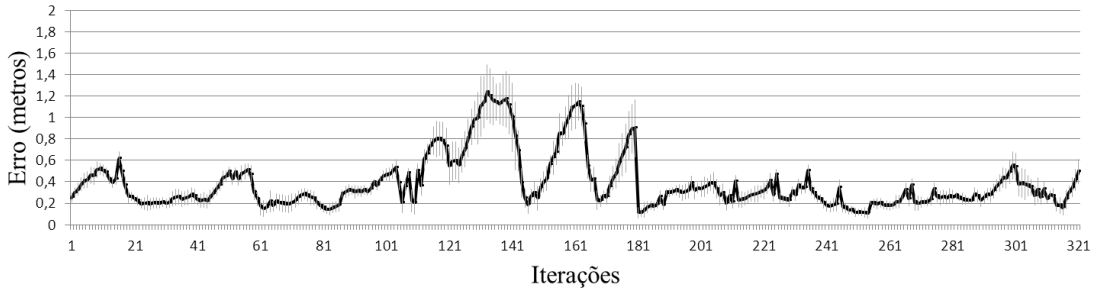


(c) Híbrido (contratores).

Figura 5.29: Erros - Trajetória 1 ambiente 3 - Problema do Robô Raptado.



(a) Híbrido (SIVIA).



(b) Híbrido (contratores).

Figura 5.30: Erros - Trajetória 1 ambiente 3 - escala diferenciada - Problema do Robô Raptado.

comparação os métodos híbridos apresentaram valores como 0.23, 0.32 e 0.31 (híbrido com SIVIA) e 0.73, 0.87 e 1.42 (híbrido com contratores).

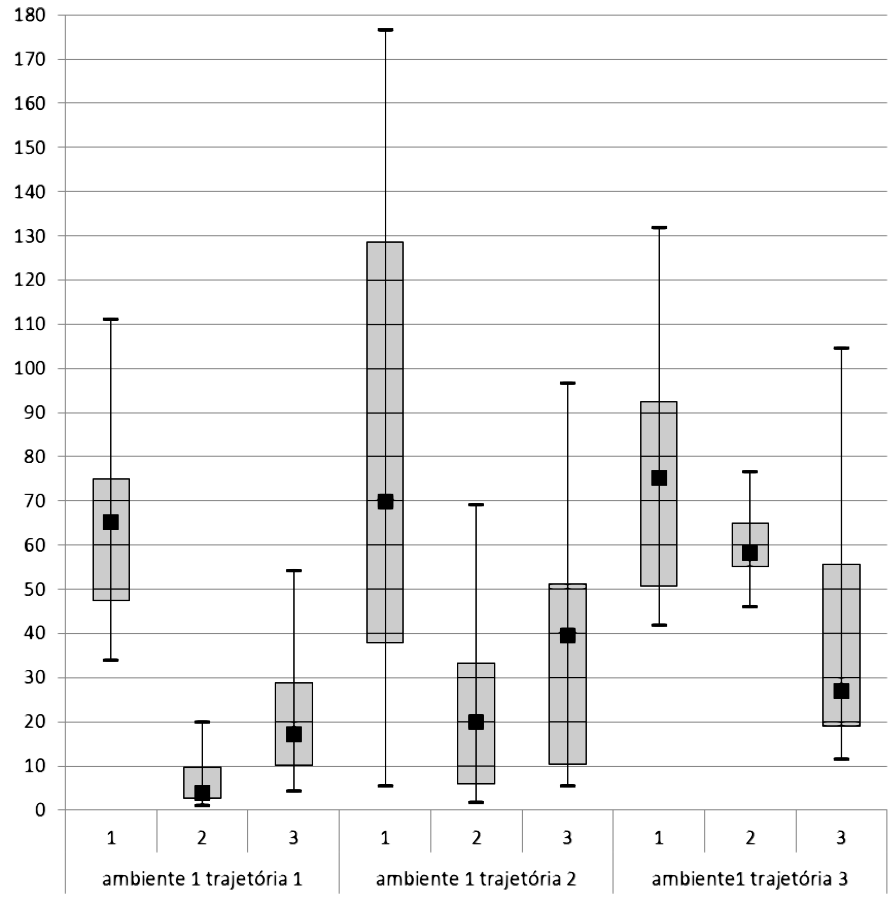


Figura 5.31: Diagrama de caixa para os dados do ambiente 1 para o problema do robô raptado. Onde 1, 2 e 3 representam respectivamente os métodos filtro de partículas, método híbrido com sivia e método híbrido com contratores.

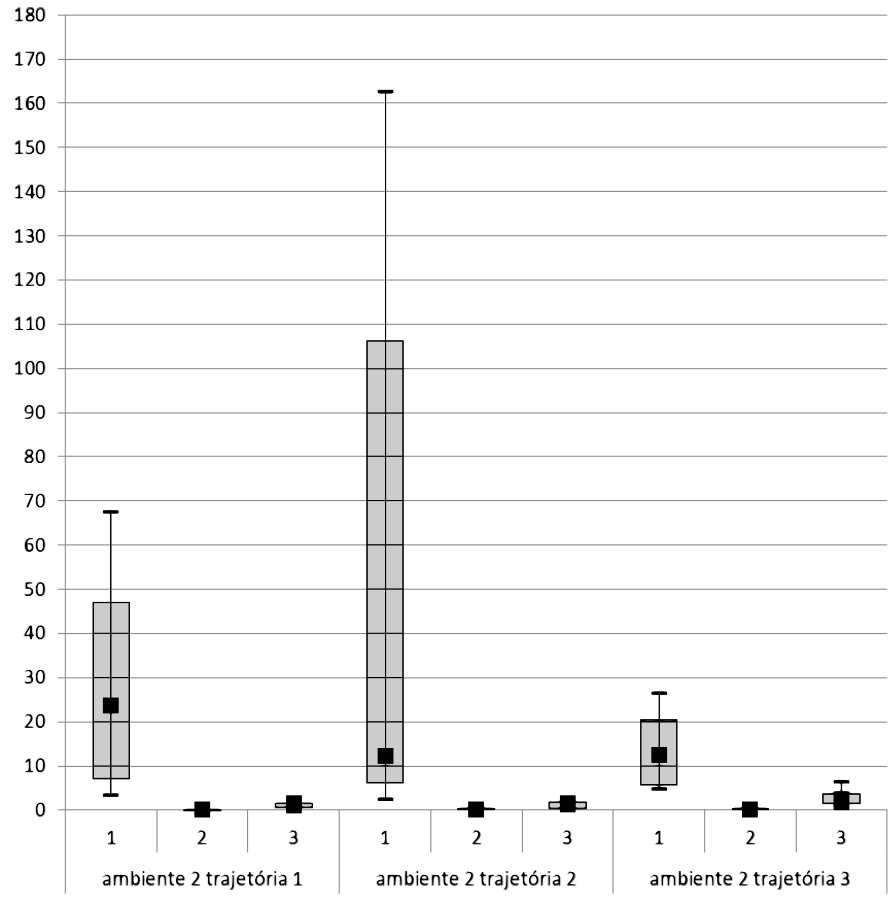


Figura 5.32: Diagrama de caixa para os dados do ambiente 2 para o problema do robô raptado. Onde 1, 2 e 3 representam respectivamente os métodos filtro de partículas, método híbrido com sivia e método híbrido com contratores.

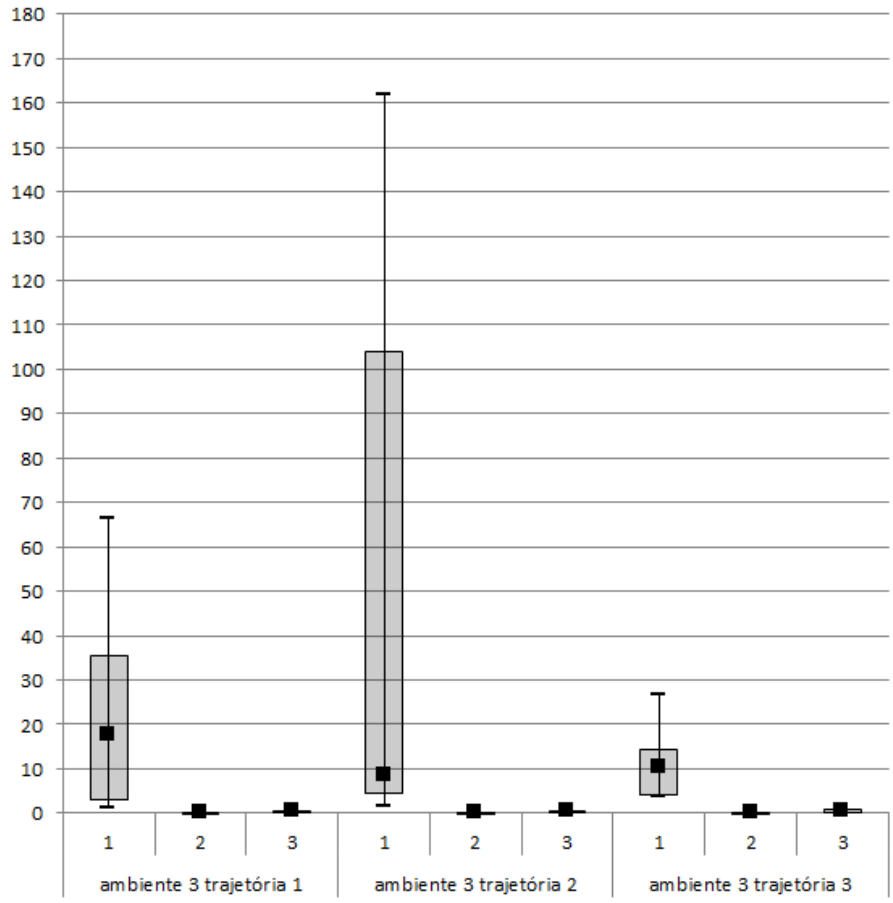


Figura 5.33: Diagrama de caixa para os dados do ambiente 3 para o problema do robô raptado. Onde 1, 2 e 3 representam respectivamente os métodos filtro de partículas, método híbrido com sivia e método híbrido com contratores.

6 CONCLUSÃO

Esta pesquisa apresentou duas hibridizações do filtro de partículas com técnicas intervalares. Os métodos foram aplicados ao problema de auto localização global, e os experimentos apresentados no Capítulo 5.

As técnicas intervalares, SIVIA e contratores foram empregadas nos experimentos como limitadores da incerteza associada ao problema. Com a incerteza inerente ao problema delimitada, e normalmente menor que a incerteza inicial, esperávamos que as partículas se concentrassem em regiões de real interesse e que fossem melhor aproveitadas.

A primeira hibridização apresentada utilizou o SIVIA com o filtro de partículas. Apesar do importante aumento na precisão dos resultados, houve um acréscimo significativo no tempo de execução requerido. Para a utilização desse método é preciso avaliar se o tempo extra requerido é compensado pela melhoria na precisão dos resultados.

Devido a esse aumento excessivo no tempo de execução, foi proposta uma segunda hibridização, utilizando filtro de partículas com contratores. Apesar de não apresentar resultados tão precisos como o SIVIA, o uso de contratores requereu pouco tempo extra para execução e mesmo assim, apresentou uma precisão consideravelmente maior que o filtro de partículas tradicional isoladamente.

Embora suscetível a erros, os resultados dos métodos mostraram aumento na precisão da localização do robô. Nessas hibridizações, a incerteza, antes global, ficou limitada a uma região representada por uma caixa ou um conjunto de caixas. A ação do filtro de partículas foi limitado a esse espaço, que é matematicamente garantido pela análise de intervalos. O resultado estimado pelas partículas dentro dessa região é probabilístico. Sendo assim, ainda é possível ocorrer erros comuns ao filtro de partículas. Porém, mesmo no pior caso, os erros não excederão os limites impostos pela análise de intervalos.

Outro benefício da abordagem híbrida é a facilidade na detecção de erros, como leituras inconsistentes ou má convergência do método. Técnicas intervalares normalmente geram como resultado conjuntos vazios, que no contexto do problema abordado, é um forte indicativo de erro. Foi possível observar esse benefício nos experimentos referentes ao problema do robô raptado, nos quais ambos os métodos híbridos conseguiram operar adequadamente, enquanto que o filtro de partículas tradicional não obteve resultados satisfatórios.

Os benefícios iniciais esperados com o uso de um método híbrido foram alcançados. Os métodos tornam possível uma maior cobertura da região de incerteza, uma vez que

essa região pode ser reduzida. Devido aos testes serem controlados através de simulação, foi possível visualizar que os resultados intervalares sempre continham a real posição do robô. Isso era esperado, tendo em vista que esses limites foram obtidos usando análise de intervalos.

Os métodos utilizados na hibridização se complementam. As características herdadas da abordagem intervalar proporcionam redução do espaço de amostras. Essa redução é útil para métodos como filtro de partículas, pois concentra as partículas em regiões de alta probabilidade, o que leva a uma maior precisão na localização do robô. Além disso, como o espaço de amostras é menor, uma menor quantidade de partículas poderia ser usada. Por outro lado, os métodos apresentados usando análise de intervalos sempre geram regiões onde todas as possíveis localizações do robô são equiprováveis. O filtro de partículas ajuda a definir mais precisamente onde, na região definida pela análise de intervalos, está o robô.

Nossos experimentos forneceram evidências de que a abordagem híbrida proposta é capaz de fornecer resultados melhores do que os obtidos usando cada uma das técnicas isoladamente.

Finalmente, cabe salientar que os métodos apresentados podem ser usados, sem quaisquer alterações, em diferentes situações, desde que as informações básicas descritas na seção 3 sejam respeitadas. Isso inclui o uso de robôs aéreos ou terrestres de diferentes configurações. Os marcadores podem ser detectados através de ondas sonoras, características visuais ou mesmo através de odor. Além disso, existe a necessidade de conhecer a localização dos marcadores a cada instante de tempo, contudo, não é obrigatório que os marcadores sejam fixos.

6.1 Trabalhos Futuros

Algumas modificações podem tornar o método melhor ou mais genérico. Uma das melhorias possíveis é o uso de marcadores não-distinguíveis. Isso elimina uma restrição importante do método, porém aumenta a dificuldade do problema. Neste caso, quando uma leitura for feita, o método não saberá a qual marcadores ela está associada e para não perder a garantia de solução, deverá tratá-la em função de todos os marcadores.

Outra opção a ser seguida é a aplicação dos métodos híbridos para testes em ambientes reais. Entretanto, dada a quantidade de ruídos nas leituras e a dinamicidade de um ambiente real, é necessário o tratamento de *outliers*. Isso pode acarretar em muitas dificuldades, pois o método de eliminação de *outliers* deverá ser bom o bastante, para ter certeza de que não haverá descarte de soluções factíveis.

Por fim, uma extensão natural das abordagens propostas é a utilização no tratamento do problema de SLAM. Na resolução de tal problema, as posições dos marcadores não são conhecidas. A partir da movimentação do robô e das leituras extraídas do ambiente, o robô precisa se localizar e localizar os demais elementos do mapa, sejam marcadores ou obstáculos. Filtros de partículas possuem uma variante usada no SLAM, chamada filtro de partículas Rao-Blackwellized (THRUN et al., 2005), que é muito difundida na literatura. Ele funciona de forma bastante similar ao filtro de partículas apresentado neste trabalho. A diferença principal é que cada partícula carrega consigo um mapa do ambiente, ou no

caso em questão, estimativas de localização de todos os marcadores. A parte do processo associada a estimativa da posição do robô funciona da mesma forma que na localização, através de etapas de amostragem, pesagem e reamostragem de partículas. Logo, com algumas modificações os métodos híbridos podem ser adaptados para este problema.

REFERÊNCIAS

ARNOLD, D. N. **Some disasters attributable to bad numerical computing**. Disponível em: <http://www.ima.umn.edu/ arnold/disasters/ >.

BENHAMOU, F.; GOUALARD, F.; GRANVILLIERS, L.; PUGET, J.-F. Revising Hull and Box Consistency. In: INT. CONF. ON LOGIC PROGRAMMING. **Proceedings...** MIT press, 1999. p.230–244.

CHABERT, G. **Ibex - An Interval Based EXplorer**. Disponível em: <www.ibex-lib.org >.

DANDACH, H.; ABDALLAH, F.; DE MIRAS, J.; CHARARA, A. Vehicle dynamics estimation using Box Particle Filter. In: CONTROL AUTOMATION ROBOTICS & VISION (ICARCV), 2012 12TH INTERNATIONAL CONFERENCE ON. **Proceedings...** [S.l.: s.n.], 2012. p.118–123.

DELLAERT, F.; FOX, D.; BURGARD, W.; THRUN, S. Monte carlo localization for mobile robots. In: ROBOTICS AND AUTOMATION, 1999. PROCEEDINGS. 1999 IEEE INTERNATIONAL CONFERENCE ON. **Proceedings...** [S.l.: s.n.], 1999. v.2, p.1322–1328.

DIMURO, G. P. **Domínios intervalares da matemática computacional**. 1991. Dissertação (Mestrado em Ciência da Computação) — UFRGS.

ECHEVERRIA, G.; LASSABE, N.; DEGROOTE, A.; LEMAIGNAN, S. Modular open robots simulation engine: morse. In: ICRA. **Proceedings...** IEEE, 2011. p.46–51.

GIL, A.; REINOSO, Ó.; BALLESTA, M.; JULIÁ, M. Multi-robot visual SLAM using a Rao-Blackwellized particle filter. **Robotics and Autonomous Systems**, [S.l.], v.58, n.1, p.68–80, 2010.

GNING, A.; RISTIC, B.; MIHAYLOVA, L. A box particle filter for stochastic and set-theoretic measurements with association uncertainty. In: INFORMATION FUSION (FUSION), 2011 PROCEEDINGS OF THE 14TH INTERNATIONAL CONFERENCE ON. **Proceedings...** [S.l.: s.n.], 2011. p.1–8.

GNING, A.; RISTIC, B.; MIHAYLOVA, L. Bernoulli particle/box-particle filters for detection and tracking in the presence of triple measurement uncertainty. **Signal Processing, IEEE Transactions on**, [S.l.], v.60, n.5, p.2138–2151, 2012.

GUIBAS, L. J.; MOTWANI, R.; RAGHAVAN, P. The Robot Localization Problem. In: WORKSHOP ON ALGORITHMIC FOUNDATIONS OF ROBOTICS, 1. **Proceedings...** [S.l.: s.n.], 1995. p.269–282.

HICKEY, T.; JU, Q.; VAN EMDEN, M. H. Interval arithmetic: from principles to implementation. **J. ACM**, New York, NY, USA, v.48, n.5, p.1038–1068, Sept. 2001.

JAULIN, L. Robust set-membership state estimation; application to underwater robotics. **Automatica**, [S.l.], v.45, n.1, p.202–206, 2009.

JAULIN, L. Range-only slam with occupancy maps: a set-membership approach. **Robotics, IEEE Transactions on**, [S.l.], v.27, n.5, p.1004–1010, 2011.

JAULIN, L.; KIEFFER, M.; DIDRIT, O.; WALTER, E. **Applied interval analysis**: with examples in parameter and state estimation, robust control and robotics. [S.l.]: Springer, 2001.

JAULIN, L.; WALTER, E. Set inversion via interval analysis for nonlinear bounded-error estimation. **Automatica**, [S.l.], v.29, n.4, p.1053–1064, 1993.

KIEFFER, M.; JAULIN, L.; WALTER, E. Guaranteed recursive non-linear state bounding using interval analysis. **International Journal of Adaptive Control and Signal Processing**, [S.l.], v.16, n.3, p.193–218, 2002.

KLATTE, R.; CORLISS, G. F. **C-XSC**: a c++ class library for extended scientific computing. [S.l.]: Springer-Verlag Berlin, 1993.

KNUPPEL, O. PROFIL/BIAS-A fast interval library. **Computing**, [S.l.], v.53, n.3-4, p.277–287, 1994.

KUEVIAKOE, I.; LAMBERT, A.; TARROUX, P. Comparison of Interval Constraint Propagation Algorithms for Vehicle Localization. **A Journal of Software Engineering and Applications**, [S.l.], v.5, p.157–162, 2012.

LANGERWISCH, M.; WAGNER, B. Guaranteed mobile robot tracking using robust interval constraint propagation. In: **Intelligent Robotics and Applications**. [S.l.]: Springer, 2012. p.354–365.

MAFFEI, R.; JORGE, V.; KOLBERG, M.; PRESTES, E. Segmented DP-SLAM. In: INTELLIGENT ROBOTS AND SYSTEMS (IROS), 2013 IEEE/RSJ INTERNATIONAL CONFERENCE ON. **Proceedings...** [S.l.: s.n.], 2013. p.31–36.

MEIZEL, D.; LÉVÊQUE, O.; JAULIN, L.; WALTER, E. Initial localization by set inversion. **Robotics and Automation, IEEE Transactions on**, [S.l.], v.18, n.6, p.966–971, 2002.

MERLET, J.-P. ALIAS: an interval analysis based library for solving and analyzing system of equations. **SEA, June. Automation**, [S.l.], p.1964–1969, 2000.

MERLET, J.-P. Interval analysis and robotics. In: **Robotics Research**. [S.l.]: Springer, 2011. p.147–156.

PETROV, N.; ULMKE, M.; MIHAYLOVA, L.; GNING, A.; SCHIKORA, M.; WIENEKE, M.; KOCH, W. On the performance of the Box Particle Filter for extended object tracking using laser data. In: SENSOR DATA FUSION: TRENDS, SOLUTIONS, APPLICATIONS (SDF), 2012 WORKSHOP ON. **Proceedings...** [S.l.: s.n.], 2012. p.19–24.

PRESTES, E.; RITT, M.; FUHR, G. Improving monte carlo localization in sparse environments using structural environment information. In: INTELLIGENT ROBOTS AND SYSTEMS, 2008. IROS 2008. IEEE/RSJ INTERNATIONAL CONFERENCE ON. **Proceedings...** [S.l.: s.n.], 2008. p.3465–3470.

ROKNE, J. G. Interval arithmetic and interval analysis: an introduction. In: **Granular computing**. [S.l.]: Springer, 2001. p.1–22.

RUMP, S. INTLAB - INTerval LABoratory. In: CSENDES, T. (Ed.). **Developments in Reliable Computing**. [S.l.]: Springer Netherlands, 1999. p.77–104.

SABBI, A. S.; HUBER, M. Particle filter based object tracking in a stereo vision system. In: ROBOTICS AND AUTOMATION, 2006. ICRA 2006. PROCEEDINGS 2006 IEEE INTERNATIONAL CONFERENCE ON. **Proceedings...** [S.l.: s.n.], 2006. p.2409–2415.

SEIGNEZ, E.; LAMBERT, A. Complexity study of guaranteed state estimation applied to robot localization. In: CONTROL, AUTOMATION, ROBOTICS AND VISION, 2008. ICARCV 2008. 10TH INTERNATIONAL CONFERENCE ON. **Proceedings...** [S.l.: s.n.], 2008. p.398–405.

SIEGWART, R.; NOURBAKHSH, I. R. Autonomous mobile robots. **Massachusetts Institute of Technology**, [S.l.], 2004.

THRUN, S. Particle filters in robotics. In: EIGHTEENTH CONFERENCE ON UNCERTAINTY IN ARTIFICIAL INTELLIGENCE. **Proceedings** [S.l.: s.n.], 2002. p.511–518.

THRUN, S.; BURGARD, W.; FOX, D. et al. **Probabilistic robotics**. [S.l.]: MIT press Cambridge, 2005. v.1.

WANG, B.; HE, G.; LIU, K.; LV, H.; YIN, W.; MEI, S. Guaranteed state estimation of power system via interval constraints propagation. **Generation, Transmission Distribution, IET**, [S.l.], v.7, n.2, p.138–144, 2013.