LOCALIZAÇÃO E MAPEAMENTO SIMULTÂNEOS EM GRANDES AMBIENTES: UMA ABORDAGEM HÍBRIDA

EDUARDO DA SILVA ALVES

DISSERTAÇÃO DE MESTRADO EM ENGENHARIA ELÉTRICA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

FACULDADE DE TECNOLOGIA

UNIVERSIDADE DE BRASÍLIA

UNIVERSIDADE DE BRASÍLIA
FACULDADE DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

LOCALIZAÇÃO E MAPEAMENTO SIMULTÂNEOS EM GRANDES AMBIENTES: UMA ABORDAGEM HÍBRIDA

EDUARDO DA SILVA ALVES

DISSERTAÇÃO DE MESTRADO ACADÊMICO SUBMETIDA AO DEPARTAMENTO DE ENGENHARIA ELÉTRICA DA FACULDADE DE TECNOLOGIA DA UNIVERSIDADE DE BRASÍLIA COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM ENGENHARIA ELÉTRICA.

APROVADA POR:

_____

Prof. Geovany Araújo Borges, ENE/UnB
(Orientador)

_____

Prof. Flávio de Barros Vidal, CIC/UnB
Examinador Externo

_____

Prof. João Yoshiyuki Ishihara, ENE/UnB
Examinador Interno

BRASÍLIA, 21 DE DEZEMBRO DE 2012.

FICHA CATALOGRÁFICA

ALVES, EDUARDO DA SILVA
Localização e mapeamento simultâneos emgrandes ambientes: uma abordagem híbrida [Distrito Federal] 2012.
xi, 77p., 210 x 297 mm (ENE/FT/UnB, Mestre, Engenharia Elétrica, 2012).
Dissertação de Mestrado – Universidade de Brasília, Faculdade de Tecnologia.
Departamento de Engenharia Elétrica

1. SLAM	2. Sistemas dinâmicos híbridos
3. Robótica Móvel Terrestre	4. Filtro de Kalman Estendido
I. ENE/FT/UnB	II. Título (série)

REFERÊNCIA BIBLIOGRÁFICA

ALVES, E.S. (2012). Localização e mapeamento simultâneos emgrandes ambientes: uma abordagem híbrida, Dissertação de Mestrado em Engenharia Elétrica, Publicação PPGENE.DM-351/08, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF, 77p.

CESSÃO DE DIREITOS

AUTOR: Eduardo da Silva Alves
TÍTULO: Localização e mapeamento simultâneos emgrandes ambientes: uma abordagem híbrida.
GRAU: Mestre ANO: 2012

Eduardo da Silva Alves
Departamento de Eng. Elétrica (ENE) - FT
Universidade de Brasília (UnB)
Campus Darcy Ribeiro
CEP 70919-970 - Brasília - DF - Brasil

Aos meus pais Marizete e Francisco, à minha vozinha Divina e a um grande amigo chamado Henrique.

AGRADECIMENTOS

Sinto-me um pouco aliviado escrevendo esta seção de agradecimentos! Contudo, finalizar esta parte talvez seja mais difícil do que escrever o conteúdo de toda a dissertação. Este trabalho de mestrado durou um longo período e, entre altos e baixos, sinto-me na obrigação de agradecer a muitas pessoas.

Considero, antes de tudo, um agradecimento sincero a Deus.

Não poderia de deixar de agradecer meu orientador, o professor Geovany. Muito obrigado pela compreensão durante todo esse tempo, pois o senhor sabe das dificuldades que tive em terminar este mestrado. Também gostaria de lembrar do professor Ishihara que, de certa forma, também contribuiu para que este trabalho chegasse ao fim. Ainda mais, muito obrigado a todos os companheiros de trabalho do LARA, que sempre estiveram dispostos a me ajudar, não só nos problemas técnicos enfrentados durante este período, mas também na sua amizade e companheismo. Um obrigado especial ao Henrique. Cara, você se mostrou um amigo incrível!

Várias pessoas sofreram um pouco comigo neste período, principalmente meus familiares. Pai, muito obrigado! Sei que você sempre acreditou que isso um dia ia acabar. Obrigado pela força que você me deu durante todo este tempo. Guga, também queria te agradecer de coração. Também ao meu tio Magu e minha tia Linda que também se preocuparam muito comigo e torceram bastante para finalização deste mestrado.

Por último, mas não menos importante, queria dedicar um agradecimento mais que especial a três mulheres que durante este tempo tiveram um papel primordial na minha vida. Minha mãe, minha avó e a Fabi. Mãe eu sei que já te preocupei muito e gostaria que a conclusão deste trabalho fosse para você um presente do filho que te ama muito. Vozinha, muito obrigado pelas suas orações. Gostaria que a senhora soubesse que este trabalho é dedicado para a senhora com todo carinho do mundo.

Por fim, gostaria de agradecer à pessoa que conviveu comigo durante muito tempo, mas que neste momento não poderá ver a conclusão deste trabalho. Fabi, sei que você torceu muito para que este dia chegasse, me apoiou e se preocupou tanto quanto eu para que este mestrado fosse concluído. Você não tem idéia de quanto me ajudou em tudo! Muito obrigado!

RESUMO

LOCALIZAÇÃO E MAPEAMENTO SIMULTÂNEOS EM GRANDES AMBIENTES: UMA ABORDAGEM HÍBRIDA

Autor: Eduardo da Silva Alves

Orientador: Prof. Geovany Araújo Borges, ENE/UnB

Programa de Pós-graduação em Engenharia Elétrica

Brasília, 21 de dezembro de 2012

Mapeamento e localização simultâneos (SLAM, da sigla em inglês) é um dos assuntos mais pesquisados no campo da robótica. Este trabalho propõe uma abordagem de sistemas dinâmicos híbridos para tratar do problema de SLAM. Dentro desta perspectiva, desenvolve-se um modelo matemático para o problema de localização e mapeamento em grandes ambientes e propõe-se uma modificação no algoritmo do filtro híbrido IMM de forma a se alcançar um melhor desempenho durante o processo de estimação estocástica do vetor de estados do sistema. Este novo algoritmo, juntamente com a formulação mais tradicional FKE-SLAM e do filtro de partículas (FastSLAM), tem seu desempenho comparado por meio de dados de simulação. Sugere-se que a formulação apresentada neste trabalho pode superar os resultados disponíveis na literatura em termos de complexidade computacional.

ABSTRACT

SIMULTANEOUS LOCALIZATION AND MAPPING IN LARGE-SCALE ENVIRONMENTS: A HYBRID APPROACH

Author: Eduardo da Silva Alves

Supervisor: Prof. Geovany Araújo Borges, ENE/UnB

Programa de Pós-graduação em Engenharia Elétrica

Brasília, 21th December 2012

Simultaneous localization and mapping (the acronym, SLAM) is one of the most interesting topics in the field of robotics. This paper proposes a hybrid dynamic systems approach to address the problem of SLAM. Within this perspective, it develops a mathematical model to the problem of localization and mapping and propose a modification in the hybrid filter IMM algorithm in order to achieve better performance during the estimation process. This new algorithm, together with the formulation EKF-SLAM and FastSLAM, is compared by means of simulation data. It is suggested that the formulation presented in this paper can overcome the results available in the literature in terms of computational complexity.

SUMÁRIO

LISTA DE FIGURAS

1 INTRODUÇÃO GERAL

1.1 MOTIVAÇÃO

Mapeamento e localização simultâneos (SLAM, da sigla em inglês) é um dos assuntos mais pesquisados no campo da robótica. Uma noção bastante intuitiva do problema de SLAM pode ser mostrada através do seguinte exemplo hipotético. Considere um simples robô móvel: um conjunto de rodas conectadas a um motor e uma câmera juntamente com alguns dispositivos para controlar a velocidade e a direção da plataforma como um todo. Agora imagine que este robô está sendo guiado remotamente por um operador humano para mapear lugares inacessíveis, tais como minas de extração de carvão, ambientes submarinos ou até mesmo outros planetas do nosso sistema solar. Os atuadores presentes neste robô permitem que ele se desloque pelo ambiente e as câmeras fornecem informação visual suficiente para que o operador identifique os objetos em torno do robô e também saiba como estes objetos estão orientados com relação ao próprio robô. O que este operador humano está fazendo é um exemplo de SLAM. A determinação da posição de objetos no ambiente é um exemplo de mapeamento e estabelecer a posição e orientação do robô com relação a estes objetos é um exemplo de localização. Grandes esforços têm sido feitos no campo da robótica no sentido de encontrar alternativas para que o próprio robô solucione o problema de SLAM de forma autônoma.

Uma aplicação bem conhecida e de bastante sucesso consiste na exploração do planeta Marte pelo robô *Spirit* ilustrado na Figura 1.1. Neste exemplo, o robô *Spirit* encontra-se em um ambiente completamente desconhecido e hostil, onde a teleoperação humana é difícil e lenta devida à longa distância. O robô age da seguinte forma: ele observa o caminho à sua frente e classifica qual a região mais segura para se trafegar.

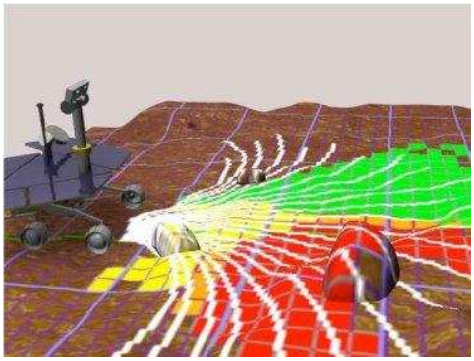


Figura 1.1: Navegação do robô *Spirit* [1]

Outra aplicação bem interessante é em robótica subaquática. A Figura 1.2 mostra a plataforma TRIMARES, projeto proposto por uma colabaração entre dois grupos de robótica portugueses do Instituto de Engenharia de Sistemas e Computadores (INESC Porto) e Ins-

tituto Superior de Engenharia do Porto (ISEP) , é um submarino robô capaz inspecionar estruturas de barragens e o assoreamento de bacias hidrográficas com grau de precisão na ordem de centímetros e em tempo real. Outras aplicações nas quais a utilização de robôs móveis podem ser úteis para substituir operadores humanos em situações perigosas ou ambientes hostis é na inspeção robotizada de dutos de petróleo. Nesta tarefa, o robô mapeia a espessura das paredes dos dutos, que sofrem problemas de corrosão com o tempo.

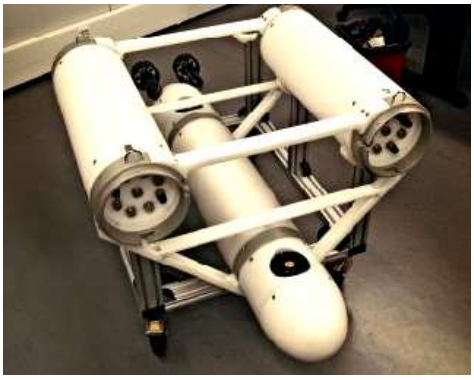


Figura 1.2: Robô submarino TRIMARES

Pode-se ainda citar o desenvolvimento de robôs autônomos para tarefas como segurança, limpeza e entretenimento. Por exemplo, o robô *Roomba* é uma plataforma comercial autônoma bastante conhecida que realiza tarefas de limpeza doméstica tais como aspiração de pó em pisos e carpetes, conforme mostrado na Figura 1.3.

Figura 1.3: Robô Roomba para limpeza em ambientes domésticos.

Entretanto, para que os robôs cheguem a este nível de aplicabilidade, questões mais básicas envolvendo o problema de SLAM devem ser solucionadas. Diante disto, nesta dissertação propõe-se uma nova solução para o problema de SLAM que é fundamental para navegação autônoma em robótica móvel.

1.2 PRINCIPAIS DIFICULDADES

Os problemas de localização e mapeamento em robótica móvel podem ser formulados e solucionados individualmente, entretanto, nos últimos anos, tem-se abordados estes dois problemas de forma simultânea. A necessidade de encarar estes dois problemas de forma simultânea surge quando o robô não tem acesso a uma mapa do ambiente e também não tem conhecimento da sua posição, portanto, durante a navegação, o robô deve realizar um mapeamento do ambiente e simultaneamente utilizar este mapa para calcular a sua posição [2]. Este processo envolve a combinação, por meio de algoritmos de filtragem, de dados de sensores proprio [1] e exteroceptivos [2]. A Figura 1.4 ilustra as principais etapas envolvidas neste processo as quais podem ser resumidas em quatro funcionalidades básicas:

- **Seleção de *landmarks*[3] no ambiente.** Esta etapa consiste em detectar no conjunto de dados provenientes dos sensores marcações notáveis no ambiente, cuja posição relativa ao robô possa ser medida.

- **Estimação de medidas relativas.** Dois processos estão envolvidos nesta etapa:

    1. Estimativas da localização dos (*landmarks*) com relação à postura do robô da qual eles são observados, estas estimativas são chamadas de ***observações***.
    2. Estimação do deslocamento do robô entre duas observações consecutivas, este processo é chamado de ***predição***. Esta estimava pode ser fornecida por sensores proprioceptivos, por um modelo cinemático de evolução ou até mesmo pela imposição de hipóteses sobre o deslocamento do robô, como um modelo de velociade constante, por exemplo.

- **Associação de dados.** As observações dos *landmarks* são úteis para calcular a estimativa de postura do robô, somente se elas são realizadas de diferentes posições. Deve-se então procurar uma correspondência entre os *landmarks* observados em diferentes instantes de tempo, caso contrário o processo de resolução do problema de SLAM torna-se totalmente inconsistente.

- **Estimação.** Consiste em integrar as várias predições e observações para estimar a postura do robô e as posições dos *landmarks* em um mesmo sistema de referência (Sistema de referência global).

O problema de SLAM é um tema de pesquisa em robótica móvel bastante ativo na comunidade. Apesar de haver um grande número de trabalhos neste tema nessas duas últimas

[1] Medem grandezas próprias do sistema, como aceleração, velocidade, rotação, inclinação, dentre outras.

[2] Medem grandezas externas ao sistema, como imagens, campo magnético local, velocidade do vento, etc.

[3] Estruturas ou características do ambiente que são facilmente observáveis, por exemplo, bordas, cantos, planos, dentre outras.

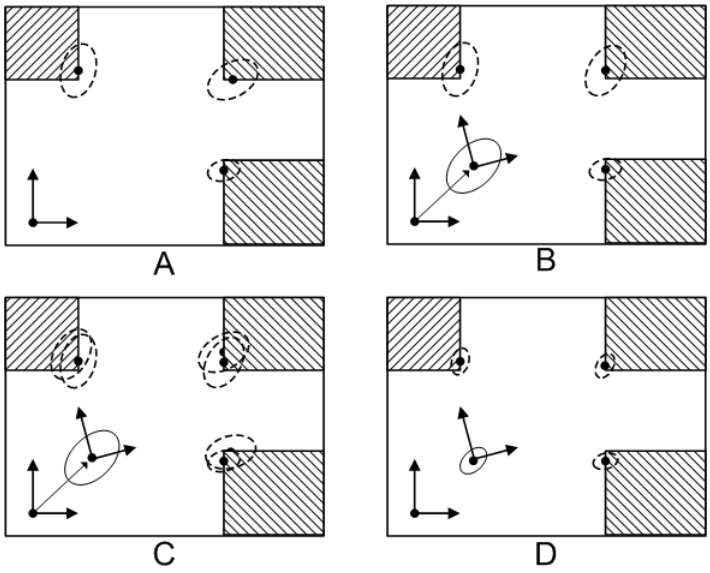


Figura 1.4: Etapas principais do processo de SLAM. Por meio das primeiras observações, o robô constrói um mapa formado por três pontos (A), então o robô desloca-se pelo ambiente e uma estimativa de sua localização é calculada (B), três novas observações são realizadas (C) e fusionadas para atualizar o mapa e consequentemente a postura do robô (D).

décadas, o problema de pesquisa ainda não está totalmente fechado, havendo ainda muito espaço para investigação. Fundamentalmente, o SLAM incorpora as etapas de localização e mapeamento que são fortemente dependentes uma da outra. A etapa de localização é concebida de modo a prover uma estimativa da postura do robô referenciada a um mapa do ambiente. Por outro lado, as informações do ambiente, capturadas por sensores associados ao sistema de coordenadas do robô, juntamente com a estimativa atual de localização podem ser utilizadas para atualizar os elementos do mapa. Uma vez atualizado, o mapa pode ser usado para resolver o problema de localização do robô levando-se em conta uma representação mais atual do ambiente. Em SLAM observa-se um forte acoplamento entre as etapas de localização e mapeamento. Em um contexto de estimação estocástica, incertezas associadas a uma etapa são propagadas para a outra.

Dentre as abordagens propostas na literatura para solucionar o problema de SLAM, os métodos baseados em estimação Bayesiana obtiveram maior sucesso [3]. Neste contexto, uma das primeiras propostas de SLAM foi feita por Smith, Self e Cheeseman [4]. Neste trabalho, um Filtro de Kalman Estendido (FKE) é usado para realizar a estimação conjunta da postura do robô e dos parâmetros geométricos dos elementos do mapa. A formulação com FKE teve muita receptividade na comunidade, pois permite incorporar de forma direta medições de sensores proprioceptivos para predição e de sensores exteroceptivos para correção da estimativa de postura e das estruturas do mapa. [5, 6]. Funcionalidades complementares tais como casamento probabilístico e mecanismos de manutenção do mapa compõem normalmente técnicas de SLAM com mapas estocásticos. [7, 8].

Um ponto chave na abordagem de SLAM com FKE é a complexidade computacional $\mathcal{O}(N^3)$ do algoritmo devido, em grande parte, à manutenção de correlações cruzadas entre elementos do mapa e da pose do robô. A solução utilizando FKE tem sido usada com sucesso em aplicações para ambientes de pequena escala, contudo a complexidade computacional desta solução limita bastante seu uso em grandes ambientes. [9]. Buscando redução de complexidade computacional e usando filtro de partículas para lidar com o processo de estimação não-linear multi-modal intrinseco ao problema do SLAM, Montemerlo *et al.* apresentaram os algoritmos FastSLAM [10] e FastSLAM2.0 [11]. Ambos algoritmos são avaliados com mapas estocáticos compostos por pontos (*landmarks*) e a estrutura de dados organizada em árvore permite uma grande redução da complexidade computacional. Várias pesquisas recentes seguiram nesta direção, conforme os trabalhos apresentados em [12, 13].

Outra grande limitação dos algoritmos de SLAM baseados em FKE está relacionada ao processo de linearização que resulta em aproximações muito otimistas para as incertezas das variáveis estimadas e isso faz com que o estimador comece a desconsiderar novas medições dos sensores exteroceptivos, levando o filtro à divergência [14, 15]. De fato, matematicamente, a maior falha no SLAM com mapas estocásticos e FKE é na dificuldade em propagar corretamente as correlações cruzadas [16]. Com o objetivo de superar estes problemas, surgiram abordagens que se preocupavam com a consistência do mapa e usavam a transformada *Unscented* para melhorar a propagação das correlações cruzadas [17, 18]. Atualmente, ainda existem trabalhos que mantêm esta preocupação e propõem variações para aplicação da transformada *Unscented* [19, 20, 21].

Um processo crítico em SLAM consiste num procedimento denominado associação de dados, ou seja, estabelecer correspondências entre as observações atuais e todos os dados previamente armazenados [22, 23, 24]. Este é um processo que exige uma série de verificações e operações de inversão matricial, contribuindo bastante para a complexidade computacional do algoritmo de filtragem, principalmente quando o robô é levado a navegar em ambientes de larga escala.

Diante disso, o que se observa na literatura é que ainda existem problemas abertos que são relacionados principalmente à convergência e ao custo computacional da solução. Uma abordagem eficaz e elegante para atacar ambos os problemas, convergência e tempo de computação, tem sido o desacoplamento do sistema em sub-mapas conforme proposto em [25, 26, 27]. Apesar de não haver nenhuma prova formal, existe uma forte evidencia empírica de que a abordagem por sub-mapas locais contribui para melhorar a consistência do processo de estimação em SLAM [14]. Em casos de desacoplamento, o mapa global pode ser obtido, pela combinação dos sub-mapas locais usando vários algoritmos tais como demonstrados em [28, 29, 30]. Outros trabalhos que propõem esta mesma abordagem são *Constant Time SLAM* [31] e o sistema ATLAS [32]. Duas técnicas que alcançaram resultados importantes no que diz respeito à redução do custo computacional são: SLSJF SLAM [33] e *Divide and Conquer* (D&C)SLAM [9].

Uma outra tendência consiste em manter armazenados os dados passados e usá-los para atualizar uma estimativa da trajetória do robô. Neste contexto, além de se estimar a postura atual do robô, estima-se também a sua postura quando da aquisição de dados anteriores. Assim se recompõe a trajetória realizada pelo robô e o mapa é montado a posteriori usando as estimativas de postura do robô sobre a trajetória e os dados armazenados. Esta abordagem é bastante conveniente quando uma localização bastante precisa do robô é mais importante do que a manutenção de uma mapa detalhado do ambiente. Desta forma, o mapa se torna consistente em cada região da trajetória e permite que, quando detectado que o robô se encontra em alguma região já visitada anteriormente, pode-se ajustar a trajetória como um todo. Alguns exemplos nessa direção são GraphSLAM [34], Graphical SLAM [35], P-SLAM [36], RBPF-SLAM [37], SAM [38], Square Root SAM [39] e Pose SLAM[40, 41, 42].

1.3 PROPOSTA DE SOLUÇÃO E PRINCIPAIS CONTRIBUIÇÕES

A principal inovação deste trabalho está no processo de filtragem que explora a formulação do problema de SLAM de acordo com um sistema dinâmico híbrido. Sistemas híbridos referem-se a uma ampla classe de sistemas dinâmicos cujo comportamento combina variáveis de estado contínuas e discretas [43, 44]. A variável discreta do vetor de estado denota um modo de operação do sistema, ou seja, trata-se de um sistema de múltiplos modelos. Em razão de sua versatilidade, o paradigma de modelagem híbrido tem sido aplicado a uma ampla variedade de situações e diversos trabalhos têm abordado o tema de estimação de estado para este tipo de sistema [45, 46]. Outros trabalhos também fizeram uso desta formulação híbrida em SLAM utilizando filtros apropriados tal como a técnica *Interacting Multiple Model* (IMM) [47, 48]. Contudo, os trabalhos anteriores, abordam temas que são primordialmente relacionados com aproximações do modelo não-linear e não-Gaussiano que pode surgir na formulação do problema de SLAM.

Neste trabalho, o problema de SLAM é tratado de forma desacoplada, ou seja, o que se estima é a pose atual do robô bem como um conjunto de poses anteriores que determinam a trajetória percorrida até aquele momento. À medida que o robô se desloca, o ambiente é particionado em várias subregiões e os dados provenientes dos sensores são armazenados formando uma sequência de sub-mapas locais associados à cada subregião. Esta divisão do espaço em subregiões não corresponde a uma divisão lógica ou semântica, como um corredor ou um cômodo, mas depende exclusivamente da natureza dos sensores utilizados no processo.

O problema de SLAM pode ser formulado como um sistema híbrido no qual a porção contínua do vetor de estados é formada pela pose atual do robô e as poses referentes a cada subregião da trajetória e a porção discreta do vetor de estados pode ser representada por uma variável que indica qual subregião o robô está tomando as medidas naquele instante.

Estimar o estado de tais sistemas normalmente requer a filtragem simultânea das porções discreta e contínua do vetor de estados e diversas técnicas podem ser encontradas em [49, 50, 51, 52] e suas referências. Dentre as técnicas mais importantes citadas, destaca-se o algoritmo Interacting Multiple Model (IMM) [50, 53], baseado em um conjunto de filtros de Kalman que serviu de base para realização deste trabalho.

Uma grande vantagem encontrada na aplicação do paradigma de modelagem de sistemas híbridos ao problema de SLAM está na possibilidade do algoritmo de filtragem prover uma estimativa de qual subregião da trajetória o robô está tomando as medidas. A idéia principal seria utilizar esta estimativa para eliminar do processo de associação de dados as subregiões com baixa probabilidade de tal forma que a verificação de correspondências seja feita apenas com um subconjunto de todos os dados armazenados até o momento.

Poranto, a contribuição deste trabalho está na proposta de uma nova formulação para o problema de SLAM que segue um paradigma de modelagem de sistemas dinâmicos híbridos. Do ponto de vista do processo de estimação, foi proposto uma modificação no algoritmo de filtragem IMM que colaborou bastante para redução da complexidade computacional da solução do problema de SLAM.

1.4 ORGANIZAÇÃO DA DISSERTAÇÃO

Essa dissertação está organizada em quatro capítulos, incluindo esta introdução geral. No Capítulo 2 é feita uma revisão teórica procurando mostrar os fundamentos matemáticos associados à formulação do problema de SLAM, bem como um levantamento das principais soluções que tratam do problema de localização e mapeamento em grandes ambientes. Neste mesmo capítulo ainda é tratado o tema de estimação em sistemas a múltiplos modelos que é um caso particular de sistemas dinâmicos híbridos e também é apresentado os fundamentos da técnica de filtragem IMM, disponível na literatura para estimação de estados deste tipo de sistema. Em seguida, o Capítulo 3 aborda as principais questões envolvendo a formulação do problema de SLAM, em que o paradigma de modelagem por sistemas híbridos é explorado com o objetivo de reduzir a complexidade computacional do problema de SLAM. No Capítulo 4 são apresentados alguns resultados numéricos que ajudam a entender as vantangens introduzidas pelo método híbrido. Por fim, no Capítulo 5 são feitas algumas últimas considerações acerca da abordagem proposta nesta dissertação, bem como algumas sugestões de trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 PRINCÍPIOS BÁSICOS EM SLAM

No Capítulo 1 foi colocada uma visão geral do problema de SLAM enfatizando um pouco da evolução histórica e listando algumas das principais limitações na resolução do problema. Nesta seção o tema de SLAM é tratado de maneira um pouco mais detalhada de forma que fique claro para o leitor os fundamentos matemáticos associados à formulação clássica do problema de SLAM enfatizando a complexidade computacional do algoritmo, bem como as principais soluções que buscam a redução do custo computacional sobretudo quando o robô se locomove em grandes ambientes.

2.1.1 Robôs móveis para SLAM

A maior parte dos procedimentos que compõem o problema de SLAM possuem alguma dependência com relação ao modelo cinemático do robô e aos sensores nele implantados. Em muitos trabalhos vistos na literatura, o SLAM é tratado sob plano bidimensional, ou seja, o robô se desloca sobre o solo plano. O modelo cinemático do robô depende basicamente da construção mecânica dos eixos responsáveis pela sua locomoção. A técnica apresentada em [54] é bastante usada para obtenção do modelo cinemático, dado sob a forma direta

$$\boldsymbol{\xi} = \mathbf{J}(\mathbf{q}, \boldsymbol{\lambda}, \theta) \cdot \dot{\mathbf{q}} \tag{2.1}$$

em que $\mathbf{J}$ é denominada matriz Jacobiana do modelo cinemático direto (MCD). $\mathbf{q}$ é o vetor de variáveis articulares responsáveis pela movimentação do robô e $\boldsymbol{\lambda}$ representa o conjunto de parâmetros geométricos do robô. A pose do robô é descrita no sistema de coordenadas global (ou absoluto) $\mathcal{X}^G \times \mathcal{Y}^G$ pelo vetor $\boldsymbol{\xi} = (x, y, \theta)^T$. A Figura 2.1 ilustra os parâmetros que descrevem a pose do robô.

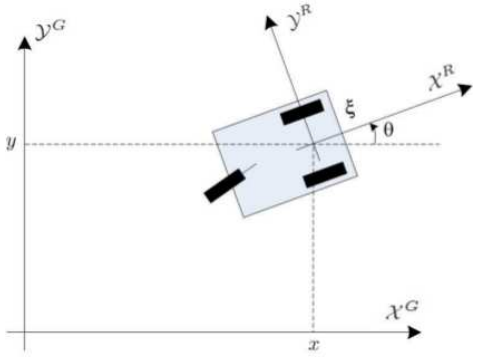


Figura 2.1: Parâmetros que descrevem a pose do robô.

No que diz respeito à parte sensorial, pode-se dispor dos seguintes instrumentos:

- Sensores proprioceptivos: são sensores utilizados em SLAM para predição da localização do robô por odometria, que é apresentada na Seção 2.1.2. Os principais sensores proprioceptivos são os codificadores ópticos das rodas (incrementais ou absolutos), girômetros e acelerômetros.

- Sensores exteroceptivos: são sensores importantes para aquisição de medidas com relação ao ambiente, o que permite a correção de erros na localização do robô e atualização do mapa do ambiente. Os mais utilizados são as câmeras de vídeo, em composição monocular ou binocular, e os radares a *laser*, popularmente conhecidos como Ladar.

2.1.2 Formulação clássica com mapas estocásticos

A solução do problema de SLAM mais tradicional considera a formulação do problema sob o ponto de vista de um processo de estimação estocástico em tempo discreto k. A localização do robô é definida pelo vetor $\boldsymbol{\xi} = (x_k, y_k, \theta_k)^T$ denominado de pose do robô no espaço de trabalho 2-D, em que (x_k, y_k) são as coordenadas de um ponto de referência no robô e θ_k é o ângulo de orientação tomado com relação ao eixo $\mathcal{X}^G$. O mapa estocástico é representado por um conjunto de estruturas do ambiente (pontos, arestas, semiplanos,etc). Os parâmetros do mapa estocástico são armazenados em um vetor $\mathbf{m}_k$.

Para explicar de forma mais genérica o processo de SLAM, considere que no instante $k-1$ o robô apresente uma pose $\boldsymbol{\xi}_{k-1}$ e o acionamento dos eixos de tração o faz realizar uma trajetória levando-o à pose $\boldsymbol{\xi}_k$. Durante este intervalo os sensores proprioceptivos capturam informações sobre o movimento do robô, isto é, os dados provenientes destes sensores estão relacionados com a trajetória realizada. No instante k, os sensores exteroceptivos capturam medidas sobre o ambiente resultando em um conjunto de observações locais com relação ao sistema de coordenadas do robô $\mathcal{X}^R \times \mathcal{Y}^R$. A relação existente entre as observações locais e as estruturas do mapa global é função da pose $\boldsymbol{\xi}$ do robô. Portanto esta relação pode ser explorada para fins de determinação da pose. Para tanto, é preciso determinar uma correspondência entre as observações locais e as estruturas armazenadas do mapa global. Uma vez realizado este processo, depara-se com dois cenários principais:

- As estruturas do mapa global que foram correspondidas com as observações locais podem ter seus parâmetros atualizados.

- Se alguma observação local não tiver correspondência no mapa global, isto pode sugerir o aparecimento de uma nova estrutura. Portanto, ela pode ser incluída no mapa global, levando em conta que seus parâmetros devem ser transformados para o sistema de coordenadas global considerando a pose $\boldsymbol{\xi}_k$ do robô.

A arquitetura de SLAM com mapas estocásticos basea-se em um esquema de predição-correção para estimação de $\boldsymbol{\xi}_k$ e $\mathbf{m}_k$, sendo que estas estimativas são denominadas de $\hat{\boldsymbol{\xi}}_k$ e

$\hat{\mathbf{m}}_k$, respectivamente. Na predição, os dados dos sensores proprioceptivos são usados para atualizar a estimativa da pose $\boldsymbol{\xi}_k$. Este processo é conhecido como odometria, que provê uma atualização da estimativa $\boldsymbol{\xi}_k$ usando medições de velocidades e orientação angular dos eixos de movimentação do robô. Em um contexto de estimação estocástica a estimativa dada pelo processo de predição é denominada de $\hat{\boldsymbol{\xi}}_{k|k-1}$. Contudo, o processo de estimação por predição apresenta um erro que aumenta com o passar do tempo. Incertezas associadas ao processo de aquisição dos dados dos sensores proprioceptivos, bem como aproximações usadas para o modelo de odometria acabam por gerar erros que são integrados no tempo. Portanto faz-se necessário sempre tomar alguma medida relativa ao ambiente para corrigir $\hat{\boldsymbol{\xi}}_{k|k-1}$, resultando na estimava $\hat{\boldsymbol{\xi}}_k$. Essas medidas são tomadas usando os sensores exteroceptivos. A partir dos dados provenientes dos sensores exteroceptivos, são extraídas características geométricas que formarão um conjunto de observações locais. Para que sejam determinadas as estruturas do mapa global que são observadas no instante atual é necessário um processo de casamento, conhecido na literatura por sua expressão em inglês *matching*. Comumente empregam-se técnicas métricas para o teste de correspondência considerando a estimativa por predição $\hat{\boldsymbol{\xi}}_{k|k-1}$. Uma vez determinadas as correspondências entre as estruturas do mapa global e as observações feitas localmente, as predições $\hat{\boldsymbol{\xi}}_{k|k-1}$ e $\hat{\mathbf{m}}_{k|k-1}$ são atualizadas, resultando em $\hat{\boldsymbol{\xi}}_k$ e $\hat{\mathbf{m}}_k$.

Conforme dito anteriormente, a modelagem mais tradicional de SLAM é abordada como um problema de filtragem estocástica, ou seja, estimação de estados de um sistema estocástico representado no espaço de estados. Assim, define-se $\mathbf{x}_k$ um vetor de estados na forma

$$\mathbf{x}_k = \begin{pmatrix} \boldsymbol{\xi}_k \\ \mathbf{m}_k \end{pmatrix} \tag{2.2}$$

com $\boldsymbol{\xi} = (x_k, y_k, \theta_k)^T$ sendo a pose do robô e $\mathbf{m}_k$ sendo o vetor de parâmetros do mapa global, formado por

$$\mathbf{m}_k = (\mathbf{s}_{1,k}^T \, \mathbf{s}_{2,k}^T \, \cdots \, \mathbf{s}_{n_k,k}^T) \tag{2.3}$$

em que $\mathbf{s}_{i,k},\ i = 1, 2, ..., n_k$, é o vetor de parâmetros da i-ésima estrutura do mapa, composto de n_k estruturas. Deve ser observado que a quantidade n_k de estruturas do mapa pode variar com o tempo. Define-se então um modelo estocástico na forma

$$\mathbf{x}_k = \mathbf{f}(\mathbf{x}_{k-1}, \mathbf{u}_k, \mathbf{w}_k) \tag{2.4}$$

$$\mathbf{y}_k = \mathbf{h}(\mathbf{x}_k) + \mathbf{v}_k \tag{2.5}$$

em que (2.4) é denominado modelo de processo e (2.5) é o modelo de medição. O modelo de processo revela como o vetor de estados evolui no tempo discreto em resposta a uma entrada $\mathbf{u}_k$, que frequentemente são medidas dos sensores proprioceptivos. O vetor $\mathbf{w}_k$ é uma variável aleatória que modela o ruído de processo e as fontes de incerteza do processo de odometria, geralmente suposto $\mathbf{w}_k \sim N(\mathbf{0}, \mathbf{Q}_k)$. Além disso, percebe-se que $\boldsymbol{\xi}_k$ e $\mathbf{m}_k$ se tornam correlacionados, pois para atualizar o mapa é necessário usar uma estimativa da pose

do robô. E assim, $\mathbf{w}_k$ também tem influência no aumtento de incerteza de $\mathbf{m}_k$. Em geral usa-se

$$\mathbf{f}(\mathbf{x}_{k-1}, \mathbf{u}_k, \mathbf{w}_k) = \begin{pmatrix} \mathbf{f_o}(\boldsymbol{\xi}_{k-1}, \mathbf{u}_k, \mathbf{w}_{o,k}) \\ \mathbf{m}_{k-1} + \mathbf{w}_{m,k} \end{pmatrix} \tag{2.6}$$

com $\mathbf{f_o}(\boldsymbol{\xi}_{k-1}, \mathbf{u}_k, \mathbf{w}_{o,k})$ representando o processo de odometria. O vetor de incertezas $\mathbf{w}_k = (\mathbf{w}_{o,k}^T, \mathbf{w}_{m,k}^T)^T$ é composto por $\mathbf{w}_{o,k} \sim N(\mathbf{0}, \mathbf{Q}_{o,k})$ que corresponde ao ruído atuando no processo de odometria e $\mathbf{w}_{m,k} \sim N(\mathbf{0}, \mathbf{Q}_{m,k})$ que é o ruído modelando o aumento de incerteza do mapa. Estes ruídos são considerados descorrelacionados, isto é, $E\{\mathbf{w}_{o,k}\mathbf{w}_{m,k}^T\} = \mathbf{0}$, uma vez que a movimentação do robô não provoca alterações na localização das estruturas de um ambiente. Portanto,

$$\mathbf{Q}_k = \begin{pmatrix} \mathbf{Q}_{o,k} & \mathbf{0} \\ \mathbf{0} & \mathbf{Q}_{m,k} \end{pmatrix} \tag{2.7}$$

Entretanto, isto não significa que o processo de estimação $\mathbf{Q}_{o,k}$ não interfere na incerteza dos elementos em $\mathbf{m}_k$. Esta interferência irá ocorrer e, nesse caso, é o modelo de medição que mostra uma corrrelação entre $\boldsymbol{\xi}_k$ e $\mathbf{m}_k$.

O modelo de medição (2.5) determina como medições $\mathbf{y}_k$ provenientes dos sensores exteroceptivos são relacionadas com os elementos do vetor de estados $\mathbf{x}_k$. Em geral, este modelo envolve uma procedimento de transformação de coordenadas, visto que as medidas extraídas dos sensores exteroceptivos são observações de estruturas do mapa global tomadas no espaço de coordenadas local do robô. $\mathbf{v}_k$ é uma variável aleatória que modela o ruído de medição, comumente representado por $\mathbf{v}_k \sim (\mathbf{0}, \mathbf{R}_k)$.

Um dos estimadores mais empregados na resolução do problema de SLAM foi o Filtro de Kalman Estendido (FKE) [55, 28, 56]. Segundo o FKE, a estimativa do vetor de estados $\mathbf{x}_k$ e de sua matriz de covariâncias $\mathbf{P}_k$ é obtida de forma recursiva partindo-se de condições iniciais $\hat{\mathbf{x}}_k$ e $\mathbf{P}_0$, tais que $\mathbf{x}_0 \sim N(\hat{\mathbf{x}}_0, \mathbf{P}_0)$. Pode-se ainda assumir que inicialmente o robô encontra-se na origem do sistema de coordenadas global e desta forma tem-se $\hat{\mathbf{x}}_0 = \hat{\boldsymbol{\xi}}_0 = (0, 0, 0)^T$ sendo que $n_0 = 0$ (mapa inicialmente vazio) e $\mathbf{P}_0 = diag(0, 0, 0)$. Desta forma, a cada instante de tempo discreto k o FKE calcula a estimativa $\hat{\mathbf{x}}_k$ em dois passos [57, 58]:

- Passo de predição: a partir de $\mathbf{u}_k$, faz-se uso dos seguintes procedimentos

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{f}(\hat{\mathbf{x}}_{k-1}, \mathbf{u}_k, \mathbf{0}) \tag{2.8}$$

$$\mathbf{P}_{k|k-1} = \nabla_{\mathbf{x}}\mathbf{f} \cdot \mathbf{P}_{k-1} \cdot \nabla_{\mathbf{x}}\mathbf{f}^T + \nabla_{\mathbf{w}}\mathbf{f} \cdot \mathbf{Q}_k \cdot \nabla_{\mathbf{w}}\mathbf{f}^T \tag{2.9}$$

em que

$$\nabla_{\mathbf{x}}\mathbf{f} = \left.\frac{\partial \mathbf{f}(\mathbf{x}_{k-1}, \mathbf{u}_k, \mathbf{0})}{\partial \mathbf{x}_{k-1}}\right|_{\mathbf{x}_{k-1}=\hat{\mathbf{x}}_{k-1}} \tag{2.10}$$

$$\nabla_{\mathbf{w}}\mathbf{f} = \left.\frac{\partial \mathbf{f}(\hat{\mathbf{x}}_{k-1}, \mathbf{u}_k, \mathbf{w}_k)}{\partial \mathbf{w}_k}\right|_{\mathbf{w}_k=\mathbf{0}} \tag{2.11}$$

Em robótica móvel, o método mais conhecido aplicado ao passo de predição é chamado de odometria. Este método faz uso de integração numérica do MCD do robô e de medições de movimento tomadas nos eixos, geralmente fornecidas por codificadores ópticos. Por aproximação de primeira ordem de (2.1) obtem-se a relação

$$\hat{\boldsymbol{\xi}}_k = \hat{\boldsymbol{\xi}}_{k-1} + \mathbf{J}(\mathbf{q}_k, \boldsymbol{\lambda}, \hat{\theta}_{k-1}) \cdot T\dot{\mathbf{q}}_k \tag{2.12}$$

para atualização da estimativa $\boldsymbol{\xi}$ de localização do robô a partir de $\dot{\mathbf{q}}_k$ medido com sensores ópticos entre os instantes discretos $k-1$ e k e T sendo o período de amostragem. Sendo assim o passo de predição com FKE se resume a

$$\hat{\mathbf{x}}_{k|k-1} = \begin{pmatrix} \hat{\boldsymbol{\xi}}_{k|k-1} \\ \hat{\mathbf{m}}_{k|k-1} \end{pmatrix} = \begin{pmatrix} \hat{\boldsymbol{\xi}}_{k-1} + \mathbf{J}(\mathbf{q}_k, \boldsymbol{\lambda}, \hat{\theta}_{k-1}) \cdot T\dot{\mathbf{q}}_k \\ \hat{\mathbf{m}}_{k-1} \end{pmatrix} \tag{2.13}$$

$$\mathbf{P}_{k|k-1} = \nabla_{\mathbf{x}}\mathbf{f} \cdot \mathbf{P}_{k-1} \cdot \nabla_{\mathbf{x}}\mathbf{f}^T + \nabla_{\mathbf{w}}\mathbf{f} \cdot \mathbf{Q}_k \cdot \nabla_{\mathbf{w}}\mathbf{f}^T \tag{2.14}$$

em que

$$\nabla_{\mathbf{x}}\mathbf{f} = \begin{pmatrix} \left.\frac{\partial \mathbf{f}_o(\xi_{k-1}, \mathbf{u}_k, \mathbf{w}_{o,k})}{\partial \xi_{k-1}}\right|_{\xi_{k-1}=\hat{\xi}_{k-1}} & \mathbf{0}_{3\times n_k} \\ \mathbf{0}_{n_k\times 3} & \mathbf{I}_{n_k} \end{pmatrix} \tag{2.15}$$

$$\nabla_{\mathbf{w}}\mathbf{f} = \begin{pmatrix} \left.\frac{\partial \mathbf{f}_o(\xi_{k-1}, \mathbf{u}_k, \mathbf{w}_{o,k})}{\partial \mathbf{w}_{o,k}}\right|_{\mathbf{w}_{o,k}=\mathbf{0}} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_{n_k} \end{pmatrix} \tag{2.16}$$

- Passo de correção: dispondo-se de uma medição $\mathbf{y}_k$ consiste nos seguintes procedimentos para corrigir a predição resultando na estimativa atual $\hat{\mathbf{x}}_k$

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_{k|k-1} + \mathbf{G}_k\nu_k \tag{2.17}$$

$$\mathbf{P}_k = (\mathbf{I} - \mathbf{G}_k\nabla_{\mathbf{x}}\mathbf{h})\,\mathbf{P}_{k|k-1} \tag{2.18}$$

$$= (\mathbf{I} - \mathbf{G}_k\nabla_{\mathbf{x}}\mathbf{h})\,\mathbf{P}_{k|k-1}\,(\mathbf{I} - \mathbf{G}_k\nabla_{\mathbf{x}}\mathbf{h})^T + \mathbf{R}_k \tag{2.19}$$

em que $\nu_k = \mathbf{y}_k - \mathbf{h}(\hat{\mathbf{x}}_{k|k-1})$ é chamado de termo de inovação, e

$$\mathbf{G}_k = \mathbf{P}_{k|k-1} \cdot \nabla_{\mathbf{x}}\mathbf{h}^T \cdot \left(\nabla_{\mathbf{x}}\mathbf{h} \cdot \mathbf{P}_{k|k-1} \cdot \nabla_{\mathbf{x}}\mathbf{h}^T + \mathbf{R}_k\right)^{-1} \tag{2.20}$$

é chamado de ganho de Kalman com

$$\nabla_{\mathbf{x}}\mathbf{h} = \left.\frac{\partial \mathbf{h}(\mathbf{x}_k)}{\partial \mathbf{x}_k}\right|_{\mathbf{x}_k=\hat{\mathbf{x}}_{k|k-1}} \tag{2.21}$$

Uma etapa importante que acontece entre entre os passos de predição e correção é o processo de casamento de dados. Por meio deste processo, são determinadas as correspondências entre as observações locais (extraídas dos sensores exteroceptivos) e as estruturas armazenadas no mapa global. As correspondências entre essas estruturas são determinadas usando um teste estatístico de hipótese conhecido como distância de Mahalanobis. Para

tanto, as observações locais são representadas no sistema de coordenadas global usando a estimativa $\hat{\boldsymbol{\xi}}_{k|k-1}$ através de um transformação de coordenadas dada por uma matriz de rotação e um vetor de translação, bem como a propagação das matrizes de covariâncias por essas transformações. Uma vez que todas as estruturas estejam representadas no mesmo sistema de coordenadas, a compatibilidade entre os dados observados no instante k e os dados armazenados no mapa global é verificada individualmente para cada par usando o termo de inovação dado em (2.17) e sua matriz de covariâncias dada por

$$\mathbf{S}_k = \nabla_{\mathbf{x}}\mathbf{h} \cdot \mathbf{P}_{k|k-1} \cdot \nabla_{\mathbf{x}}\mathbf{h}^T + \mathbf{R}_k \tag{2.22}$$

Por exemplo, se para um instante k, uma observação i corresponde com algum elemento do mapa global j, então a distância de Mahalanobis $D^2_{k,ij}$ satisfaz a seguinte relação [59]:

$$D^2_{k,ij} = \nu^T_{k,ij}\mathbf{S}^{-1}_{k,ij}\nu_{k,ij} < \mathcal{X}^2_{d,1-\alpha} \tag{2.23}$$

em que d é a dimensão de cada medida e 1-α representa o grau de confiança desejado.

O Algoritmo 1 resume todas as etapas envolvidas no processo de estimação em SLAM. A função *receber_medidas* diz respeito a aquisição de dados dos sensores exteroceptivos

Algoritmo 1: $(\hat{\mathbf{x}}_k, \mathbf{P}_k)$ = **fke_slam** (`num_passos`) [9]

$$\begin{aligned}
\mathbf{y}_0, \mathbf{R}_0 &= \textit{receber_medidas} \\
\hat{\mathbf{x}}_0, \mathbf{P}_0 &= \textit{init_mapa}(\mathbf{y}_0, \mathbf{R}_0)
\end{aligned}$$

for $k = 1$ to `num_passos` **do**

$$\begin{aligned}
\hat{\boldsymbol{\xi}}_k &= \textit{odometria}(\hat{\boldsymbol{\xi}}_{k-1}, \mathbf{u}_k) && (2.12) \\
\hat{\mathbf{x}}_{k|k-1}, \mathbf{P}_{k|k-1} &= \textit{predição}(\hat{\mathbf{x}}_{k-1}, \mathbf{P}_{k-1}, \hat{\boldsymbol{\xi}}_k, \mathbf{Q}_k) && (2.13) - (2.14)
\end{aligned}$$

$$\begin{aligned}
\mathbf{y}_k, \mathbf{R}_k &= \textit{receber_medidas} \\
\mathcal{H}_k &= \textit{associação_de_dados}(\hat{\mathbf{x}}_{k|k-1}, \mathbf{P}_{k|k-1}, \mathbf{y}_k, \mathbf{R}_k) && (2.22) - (2.23) \\
\hat{\mathbf{x}}_k, \mathbf{P}_k &= \textit{corrreção}(\hat{\mathbf{x}}_{k|k-1}, \mathbf{P}_{k|k-1}, \mathbf{y}_k, \mathbf{R}_k, \mathcal{H}_k) && (2.17) - (2.20) \\
\hat{\mathbf{x}}_k, \mathbf{P}_k &= \textit{aumentar_mapa}(\hat{\mathbf{x}}_k, \mathbf{P}_k, \mathbf{y}_k, \mathbf{R}_k, \mathcal{H}_k)
\end{aligned}$$

end for

`return` ($\hat{\mathbf{x}}_k$,$\mathbf{P}_k$)

que correspondem a um conjunto de estruturas do ambiente observadas localmente. A cada instante k, obtem-se dos sensores exteroceptivos um conjunto de medidas $\mathbf{y}_{k,i}(i = 1, ..., m)$. O processo de correspondência, dado no algoritmo pela função *associação_de_dados*, consiste em determinar uma identidade de cada medida em termos das estruturas do mapa global $\mathbf{s}_{k,j}(j = 1, ..., n)$. O resultado desta etapa é dado por $\mathcal{H}_k$ que é uma estrutura que indica as associações de cada medida $\mathbf{y}_{k,i}$ com as estruturas em $\mathbf{s}_{k,j}$.

2.1.3 Complexidade computacional em SLAM

Vários trabalhos [60, 61] já se preocuparam em descrever e analisar a complexidade computacional deste algoritmo, de forma que esta seção aborda a formulação matemática do FKE-SLAM levando em conta principalmente aspectos de tempo computacional da solução.

A abordagem clássica pelo FKE tem sido aplicada com sucesso em ambientes de pequena escala, tais como salas de escritório ou até mesmo cômodos em uma residência [9], todavia a complexidade computacional do algoritmo, $O(N^3)$ em que N representa o número de *landmarks* (estruturas representativas do ambiente) no mapa, limita bastante o uso desta técnica em ambientes maiores, nos quais o número de *landmarks* pode crescer significativamente. Nos últimos anos, vários pesquisadores tem se dedicado a encontrar uma solução computacionalmente eficiente para lidar com grandes ambientes [62].

Uma importante contribuição é a idéia de construir sub-mapas locais do ambiente e depois juntá-los de forma consistente a fim de obter o mapa global. DSM-SLAM (Decoupled Stochastic Mapping)[63], Constant Time SLAM [31] e o sistema ATLAS [32] são técnicas que empregam a construção de sub-mapas e que apresentam redução no custo computacional, contudo a precisão de suas estimativas são prejudicadas devido à várias aproximações feitas durante o processo de transição entre os sub-mapas. Outras técnicas que também utilizam a abordagem por sub-mapas mas que mostram-se como soluções exatas (exceto por um processo de linearização) são Map Joining SLAM [28] e CLSF-SLAM (Constrained Local Submap Filter) [29]. Um resultado interessante é encontrado em [9] onde os autores mostram que dado um certo ambiente e as características dos sensores existe um tamanho ótimo para os sub-mapas de forma que o tempo computacional da solução seja minimizado.

Outra solução em termos de redução de complexidade computacional está na exploração da característica esparsa da matriz de informação, que é a inversa da matriz de covariâncias [64, 40, 65, 66]. Esta abordagem sugere a utilização do filtro de informação estendido que é o dual do FKE na solução do problema de SLAM. Desta forma, o filtro de Kalman é expresso em função de medidas da informação sobre o vetor de estados ao invés dos estados diretamente [67]. Em [64] os autores propõem um filtro de informação estendido esparso (mais conhecido por sua sigla em inglês, SEIF) que resulta em uma complexidade $O(1)$ para o processo de atualização do vetor de informação. Todavia, esta abordagem no formato de informação do filtro de Kalman torna mais difícil o processo de correspondência de dados em SLAM visto que os estados e a matriz de covariâncias não estão diretamente disponíveis [68]. Esta limitação foi superada em [69] com a apresentação do filtro ESEIF (Exactly Sparse Extended Information Filter) que demonstra uma estratégia exata de esparsificação da matriz de informação. Neste sentido, algumas soluções têm se destacado, tais como Tectonic SAM [70] e SLSJF-SLAM (Sparse Local Submap Joining Filter) [71]. Outro algoritmo que apresentou resultados bastante interessantes foi o CF-SLAM (Combined Filter SLAM) [72] que também adota uma abordagem por sub-mapas e utiliza uma combinação do FKE com

o Filtro de Informação Estendido juntamente com uma estratégia de dividir e conquistar durante o mapeamento.

Ainda mantendo um paradigma de representação no formato de informação do filtro de Kalman, uma tendência um pouco mais recente tem tratado o problema de SLAM como um processo de estimação em que apenas um conjunto de poses é estimado ao longo da trajetória do robô, de tal forma que a informação sobre a localização dos *landmarks* no ambiente é útil apenas para fornecer um conjunto de restrições entre as poses. Na literatura, esta técnica é referida comumente como POSE-SLAM [73, 74] e se baseia na idéia de que o mapa global do ambiente pode ser recuperado uma vez que a trajetória do robô tenha sido devidamente estimada [75]. Uma grande limitação desta classe de algoritmos advem do fato de que incorporar todas as poses do robô ao longo da trajetória requer um custo computacional que cresce independentemente do tamanho da área a ser mapeada. Além disso considerar todas as possíveis relações entre as poses reduz a esparsividade da matriz de informação levando a uma queda de desempenho na execução do algoritmo [16, 76]. Buscando principalmente reduzir o custo computacional o trabalho apresentado em [74] propõe uma abordagem que leva em conta apenas um conjunto reduzido e não redundante de poses. Mas o que ainda continua sendo um empecilho à implementação em ambientes de grande escala é o processo de associção de dados, isto é, selecionar dentre todas as poses da trajetória aquelas que são mais prováveis de se obter uma correspondência entre as observações atuais e os landmarks previamente armazenados. Em [74] este problema é minimizado através da escolha de armazenamento da informação em uma estrutura de dados em árvore conforme mostrado em [77].

Uma outra categoria de algoritmos [34, 78, 79] baseia-se em uma representação gráfica do problema de SLAM em que os dados são organizados em uma estrutura de grafo cujos nós correspondem às poses do robô em diferentes instantes de tempo ou às posições de *landmarks* no ambiente e cujos arcos representam restrições entre as poses. Uma vez que o grafo é construído, um problema crucial é encontar uma configuração espacial dos nós que seja consistente com as observações feitas do ambiente. Neste sentido a solução envolve a aplicação de técnicas de otimização e minimização de erros visto que a soma de todas as restrições impostas pelos arcos no grafo recai em um problema de mínimos quadrados não linear [80]. Este tipo de abordagem segue uma tendência encontrada em trabalhos anteriores de aplicação de técnicas de otimização para solução offline em SLAM [81, 82, 83, 84, 85]. Estes trabalhos sustentam a idéia de recuperação da trajetória do robô como um todo e além disso os dados processados não são descartados permitindo que o processo de construção do mapa seja prorrogado. Adicionalmente, a possibilidade de redução de variáveis no grafo aliada às grandes ferramentas no campo da álgebra linear esparsa contribui bastante para reduzir a complexidade computacional da solução [80], haja visto o trabalho em [34] que se mostrou eficiente em resolver grandes ambientes.

Um trabalho também voltado para reconstrução da trajetória do robô e aplicação de

SLAM em grandes ambientes que alcançou resultados bastante significativos foi a técnica denominada HMT-SLAM (Hybrid Metric-Topological SLAM) [86]. Esta técnica segue uma abordagem já apresentada em trabalhos anteriores [87, 88] que considera a construção de mapas híbridos, ou seja, mapas topológicos que contêm informações méticas locais. Portanto, a formulação do problema em [86] faz uma representação topológica do ambiente por meio de grafos cujos nós demarcam uma sub-região (sub-mapas locais) e cujos arcos (conecções entre os sub-mapas) denotam uma transformação de coordenadas entre cada nó. Uma característica importante do HMT-SLAM é que o algoritmo naturalmente corrige a trajetória do robô como um todo quando da detecção de um laço fechado sem a necessidade de reconstrução de uma mapa métrico global do ambiente.

Um algoritmo bastante eficiente que também mantem a idéia de estimação da trajetória do robô e apresenta o problema de SLAM para grandes ambientes é o FAST-SLAM [10, 11]. Segundo esta técnica o problema de SLAM é decomposto em dois processos separados: o problema de estimação da localização do robô e o problema de estimação da posição de um conjunto de *landmarks* no ambiente. A Seção 2.1.4 trata com mais detalhes deste algoritmo mostrando sua formulação matemática, o processo de estimação e a justificativa em torno da fatoração do problema de SLAM em duas etapas distintas.

2.1.3.1 Complexidade computacional do FKE-SLAM

O custo computacional total na aplicação do FKE na resolução do problema de SLAM é da ordem de $O(N^3)$, em que N representa o número de elementos no mapa [9]. Grande parte das aplicações em SLAM exigem a navegação do robô em grandes ambientes o que resulta em uma quantidade significativamente grande de elementos presentes no mapa fazendo com que a utilização desta técnica seja muitas vezes impraticável, principalmente para aplicações em tempo real [61]. Nesta seção será feita uma explicação do algoritmo FKE-SLAM enfatizando os detalhes matemáticos associados à complexidade da solução.

Para simplificar a análise da complexidade computacional desta abordagem considera-se que os landmarks estão distribuídos de maneira aproximadamente uniforme pelo ambiente e que a quantidade de medições feitas a cada instante de tempo permanece mais ou menos constante. Desta forma, considere as seguintes hipóteses para algum instante de tempo k:

- o mapa contem n elementos,
- o sensor fornece m observações,
- r dessas observações correspondem a *landmarks* que já foram observados anteriormente,
- $s = m - r$, correspondem a novos *landmarks*.

A complexidade computacional em um passo do algoritmo envolve as seguintes etapas:

- Cálculo de um vetor de estados predito, $\hat{\mathbf{x}}_{k|k-1}$, $\hat{\mathbf{P}}_{k|k-1}$, cujas equações estão mostradas em (2.13)-(2.14). Conforme mostrado anteriormente esta etapa requer a obtenção das matrizes Jacobianas dadas em (2.15)-(2.16).

- Resolução do problema de associação de dados (a complexidade deste problema será melhor explicada mais a frente).

- Correção do vetor de estados predito. Esta etapa requer o cálculo da matriz Jacobiana dada em (2.21), da matriz de ganho de Kalman $\mathbf{G}_k$, bem como do termo de inovação ν_k e sua respectiva matriz de covariâncias $\mathbf{S}_k$.

Uma característica importante relacionada com a complexidade computacional do FKE-SLAM é que as matrizes Jacobianas são esparsas [60, 61, 39], ou seja, o cálculo destas matrizes tem complexidade $O(1)$. Portanto, isto faz com que o custo computacional das etapas de predição e correção seja reduzido. Por exemplo, considere o cálculo da matriz de covariâncias do termo de inovação $\mathbf{S}_k$ dado em (2.22). Sem levar em conta a esparsividade da matriz Jacobiana $\nabla_{\mathbf{x}}h$ o cálculo de $\mathbf{S}_k$ requereria rn^2+r^2n multiplicações e $rn^2+r^2n+r^2$ somas, isto é, $O(n^2)$ operações. Contudo, considerando que a matriz $\nabla_{\mathbf{x}}h$ é esparsa, com uma dimensão efetiva de $r \times c$, este mesmo cálculo exigiria $rcn + r^2c$ multiplicações e $rcn + r^2c + r^2$ somas, isto é, um custo computacional de $O(n)$. A Figura 2.2(a) ilustra esta análise, enfatizando a esparsividade da matriz Jacobiana $\nabla_{\mathbf{x}}h$. Uma análise semelhante pode mostrar que o cálculo da matriz de covariância predita $\mathbf{P}_{k|k-1}$ e da matriz de ganho de Kalman $\mathbf{G}_k$ tem complexidade computacional de $O(n)$. A maior parte do custo computacional da solução por FKE-SLAM está no cálculo da matriz de covariância corrigida $\mathbf{P}_k$ que é de $O(n^2)$ conforme mostrado na Figura 2.2(b). Portanto, o custo computacional por passo do FKE-SLAM é quadrático com relação à quantidade de elementos no mapa.

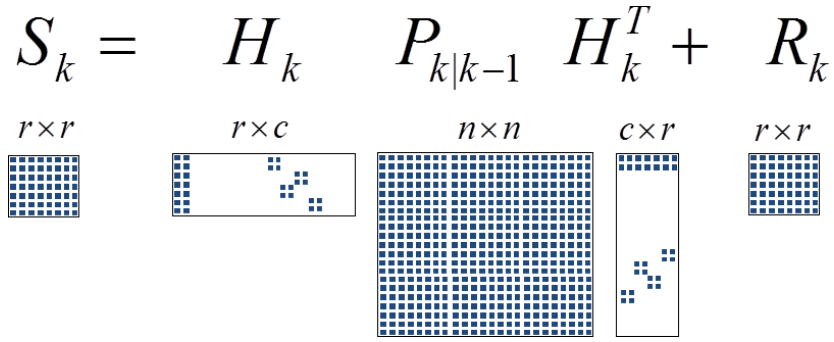
Considerando as hipóteses anunciadas anteriormente, um número constante s de novos landmarks são incorporados ao mapa em cada passo. Então, dado um ambiente com n landmarks, n/s passos são necessários para mapea-lo completamente, de forma que o custo total do FKE-SLAM seria:

$$C_{FKE\_SLAM} = \sum_{k=1}^{n/s} O((ks)^2) = \sum_{k=1}^{n/s} O(k^2) \tag{2.24}$$

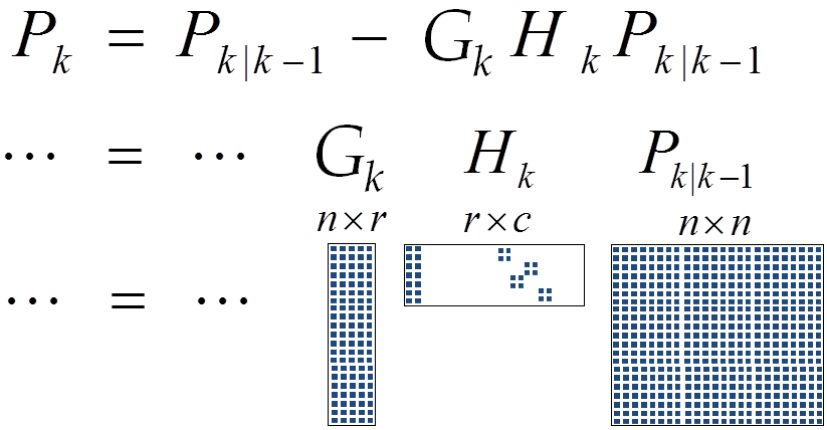
A expressão em (2.24) é comumente conhecida na literatura por soma de quadrados e pode ser escrita da seguinte forma:

$$\sum_{i=1}^{j} i^2 = \frac{j(j+1)(2j+1)}{6} \tag{2.25}$$

Então o custo computacional total da solução por FKE-SLAM é cúbico com relação ao



(a) Cálculo da matriz de covariâncias $\mathbf{S}_k$ do termo de inovação. Dado que a dimensão efetiva da matriz Jacobiana $\mathbf{H}_k = \nabla_{\mathbf{x}} h$ é $r \times c$, este cálculo exige um custo computacional de $O(n)$ operações.



(b) Matriz de covariâncias $\mathbf{P}_k$ requer $O(n^2)$ operações.

número de *landmarks* existentes no mapa, isto é,

$$\begin{aligned} C_{FKE\_SLAM} &= O\left(\frac{(n/s)(n/s+1)(2n/s+1)}{6}\right) \\ &= O\left(\frac{n^3}{s^3}\right) \\ &= O(n^3). \end{aligned} \tag{2.26}$$

Com relação ao custo computacional da etapa de associação de dados, considerando que o número de observações m permanece aproximadamente constante em cada passo, o custo computacional deste passo é $O(mn) = O(n)$, ou seja, linear com relação à quantidade de elementos do mapa.

2.1.4 FAST-SLAM: uma solução fatorada para o problema de SLAM

O FAST-SLAM é uma abordagem de SLAM para grandes ambientes que alcançou bastante sucesso na comunidade científica, visto que vários pesquisadores já se debruçaram sobre este tema e vários trabalhos já foram publicados seguindo esta abordagem [89]. Fundamentalmente, o FAST-SLAM baseia-se na idéia de que se a trajetória realizada pelo robô é conhecida então a localização de cada landmark no ambiente pode ser encarada como um

processo de estimação independente [10]. Portanto, fica evidente que o FAST-SLAM é uma solução que prevê a reconstrução da trajetória realizada pelo robô (estimação de um conjunto de poses ao longo do tempo) e desta forma o problema de determinar a localização de uma conjunto de N *landmarks* no ambiente pode ser transformado em N processos de estimação independentes, um para cada *landmark*. Esta observação foi feita inicialmente em [90].

Em um contexto de estimação estocástica, o FAST-SLAM utiliza um filtro de partículas [91] para estimação da trajetória do robô. Cada partícula possui N filtros de Kalman responsáveis por estimar as posições de N *landmarks* condicionadas à estimativa da trajetória. Uma primeira tentativa de implementação desta idéia resulta em um algoritmo que requer um custo computacional de ordem $O(MN)$, em que M é o número de partículas do filtro e N a quantidade de *landmarks* no mapa. Para o FAST-SLAM, um estratégia de redução da complexidade computacional é encontrada por meio da organização dos dados em uma estrutura em árvore. Esta estratégia ajuda bastante durante o processo de associação de dados reduzindo o custo de computação do algoritmo para $O(MlogN)$ [10].

2.1.4.1 Formulação do problema

O problema de SLAM é comumente definido na literatura como um processo de estimação estocástico, onde o que se estima é a pose atual do robô, $\boldsymbol{\xi}_k$, e a posição de um conjunto de *landmarks* espalhados pelo ambiente. Para robôs que se deslocam no plano, a pose é composta pelas coordenadas $x - y$ no plano e um ângulo de orientação com relação ao um referêncial absoluto. A pose do robô evolui ao longo do tempo de acordo com um modelo probabilístico, mais conhecido como modelo de odometria [2] dado pela seguinte f.d.p. (função densidade de probabilidade):

$$p(\boldsymbol{\xi}_k|\mathbf{u}_k, \boldsymbol{\xi}_{k-1}) \tag{2.27}$$

Portanto, $\boldsymbol{\xi}_k$ é função de um conjunto de entradas de controle $\mathbf{u}_k$ (por exemplo, a velocidade de cada roda do robô) e também da pose no instante de tempo anterior $\boldsymbol{\xi}_{k-1}$.

Cada *landmark* no ambiente, representado por $\mathbf{s}_{k,j}$ para $j = \{1, 2, ..., n_k\}$, é caracterizado pela sua posição 2-D no espaço. Para estimar a posição de todo conjunto de *landmarks* torna-se necessário observá-los e representá-los com relação a um sistema de coordenadas local fixo no robô. Desta forma, as observações extraídas através dos sensores também podem ser descritas por um modelo probabilístico, comumente referido por modelo de medição de acordo com a f.d.p. a seguir:

$$p(\mathbf{y}_k|\boldsymbol{\xi}_k, \mathbf{m}_k, \mathbf{c}_k) \tag{2.28}$$

em que $\mathbf{y}_k$ representa um conjunto de medições obtidas no instante k, $\mathbf{m}_k = \{\mathbf{s}_{1,k}, \mathbf{s}_{2,k}, ..., \mathbf{s}_{n_k,k}\}$ representa o conjunto de todos os *landmarks* e $\mathbf{c}_k \in \{1, ..., n_k\}$ é chamada de variável de correspondência e denota o índice de cada *landmark* observado no instante k. Por exemplo, na Figura 2.2 tem-se $c_1 = 1$, $c_2 = 2$ e $c_3 = 1$, visto que o robô primeiramente observa o *landmark* $\mathbf{s}_1$, depois o *landmark* $\mathbf{s}_2$ e por fim $\mathbf{s}_1$ novamente. Já foi comentado anteriormente que

o processo para determinar $\mathbf{c}_k$ é conhecido como problema de associação de dados e é parte fundamental durante a solução do problema de SLAM.

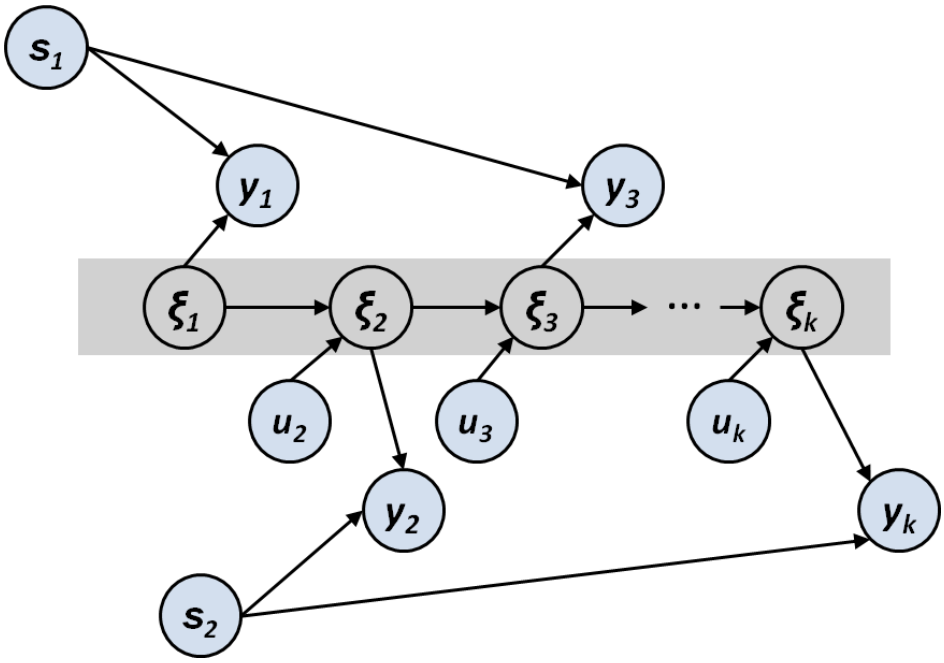


Figura 2.2: O robô se desloca a partir da posição $\boldsymbol{\xi}_1$ pela aplicação de uma sequência de entradas $\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_k$. À medida que ele se move alguns *landmarks* são observados. No instante $k = 1$ observa-se $\mathbf{s}_1$ e sua medida é representada por $\mathbf{y}_1$. Em $k = 2$ observa-se $\mathbf{s}_2$ e em $k = 3$ observa-se $\mathbf{s}_1$ novamente. A solução do problema de SLAM está relacionada com a estimação da posição dos *landmarks* no ambiente, bem como a reconstrução da trajetória realizada pelo robô a partir das entradas $\mathbf{u}_{1:k}$ e das observações $\mathbf{y}_{1:k}$.

Desta forma, pode-se formular o problema de SLAM como sendo um processo de determinar a posição de todos os *landmarks* $\mathbf{m}_k$, bem como o conjunto de todas as poses do robô ao longo do tempo $\boldsymbol{\xi}_{1:k}$, a partir de um conjunto de medições $\mathbf{y}_{1:k}$ e entradas de controle $\mathbf{u}_{1:k}$ conforme ilustrado na Figura 2.2. Em termos probabilísticos esta formulação pode ser expressa por $p(\boldsymbol{\xi}_{1:k}, \mathbf{m}_k | \mathbf{y}_{1:k}, \mathbf{u}_{1:k})$. Para facilitar a análise e o entendimento do algoritmo FAST-SLAM, assume-se que a variável de correspondência $\mathbf{c}_{1:k}$ é conhecida. Contudo, mais a frente será mostrado uma técnica de obtenção *on-the-fly* de uma estimativa destas correspondências com base nos dados extraídos dos sensores. Então, se as correspondências são conhecidas o problema pode ser reescrito da seguinte forma:

$$p(\boldsymbol{\xi}_{1:k}, \mathbf{m}_k | \mathbf{y}_{1:k}, \mathbf{u}_{1:k}, \mathbf{c}_{1:k}) \tag{2.29}$$

De acordo com o que foi discutido no começo desta seção, a função densidade de probabilidade (f.d.p.) *a posteriori* representada em (2.29) pode ser fatorada da seguinte maneira

[10, 11]:

$$p(\boldsymbol{\xi}_{1:k}, \mathbf{m}_k|\mathbf{y}_{1:k}, \mathbf{u}_{1:k}, \mathbf{c}_{1:k}) = \underbrace{p(\boldsymbol{\xi}_{1:k}|\mathbf{y}_{1:k}, \mathbf{u}_{1:k}, \mathbf{c}_{1:k})}_{\text{trajetória do robô } \textit{à posteriori}} \underbrace{\prod_{i=1}^{n_k} p(\mathbf{s}_{i,k}|\boldsymbol{\xi}_{1:k}, \mathbf{y}_{1:k}, \mathbf{u}_{1:k}, \mathbf{c}_{1:k})}_{\text{estimadores de cada } \textit{landmark}} \quad (2.30)$$

Em outras palavras, decompõe-se o problema em $n_k + 1$ processos de estimação, um problema de estimação *a posteriori* da trajetória do robô e n_k problemas de estimação da posição de n_k *landmarks* condicionados à estimativa da trajetória. Este tipo de fatoração é exata e sempre aplicável ao problema de SLAM conforme discutido em [89].

Para estimação da f.d.p. $p(\boldsymbol{\xi}_{1:k}|\mathbf{y}_{1:k}, \mathbf{u}_{1:k}, \mathbf{c}_{1:k})$ em (2.30), o algoritmo FAST-SLAM utiliza um procedimento bastante cohecido na literatura como filtro de partículas [92]. Esta técnica é bastante similar ao algoritmo de localização de Monte Carlo , da sigla em inglês MCL (*Monte Carlo Localization*) [93]. De acordo com esta técnica, a cada instante de tempo existe um conjunto de partículas, Ξ_k, que representa a f.d.p. *a posteriori* $p(\boldsymbol{\xi}_{1:k}|\mathbf{y}_{1:k}, \mathbf{u}_{1:k}, \mathbf{c}_{1:k})$. Cada partícula $\boldsymbol{\xi}_{1:k}^{[m]} \in \Xi_k$ guarda uma estimativa da trajetória do robô, isto é,

$$\Xi_k = \{\boldsymbol{\xi}_{1:k}^{[m]}\}_m = \{\boldsymbol{\xi}_1^{[m]}, \boldsymbol{\xi}_2^{[m]}, ..., \boldsymbol{\xi}_k^{[m]}\}_m \quad (2.31)$$

em que o superescrito $[m]$ é usado para se referir à *m-ésima* partícula do conjunto.

O conjunto de partículas Ξ_k é calculado de forma incremental a partir de Ξ_{k-1}, da entrada $\mathbf{u}_k$ e de um conjunto de observações $\mathbf{y}_k$. Primeiramente, cada partícula em Ξ_{k-1} é usada para gerar uma estimativa predita da pose do robô no instante k de acordo com o modelo dado em (2.27), ou seja,

$$\boldsymbol{\xi}_k^{[m]} \sim p(\boldsymbol{\xi}_k|\mathbf{u}_k, \boldsymbol{\xi}_{k-1}^{[m]}). \quad (2.32)$$

Armazenam-se estas estimativas em um conjunto de partículas temporário, juntamente com as estimativas anteriores da trajetória do robô $\boldsymbol{\xi}_{1:k-1}^{[m]}$. Partindo do pressuposto de que o conjunto de partículas Ξ_{k-1} é distribuído de acordo com $p(\boldsymbol{\xi}_{1:k-1}|\mathbf{y}_{1:k-1}, \mathbf{u}_{1:k-1}, \mathbf{c}_{1:k-1})$, cada nova partícula obtida a partir de (2.32) está distribuida de acordo com $p(\boldsymbol{\xi}_{1:k}|\mathbf{y}_{1:k-1}, \mathbf{u}_{1:k}, \mathbf{c}_{1:k-1})$. Esta distribuição de probabilidade é mais conhecida pelo seu termo em inglês *proposal distribution* do filtro de partículas.

Observa-se que durante a geração das novas partículas $\boldsymbol{\xi}_k^{[m]}$ não foram consideradas o conjunto de medições mais recente $\mathbf{y}_k$, mas somente as entradas $\mathbf{u}_k$ no instante atual. Portanto um passo adicional de reamostragem torna-se necessário para corrigir este fato [94]. Logo, o novo conjunto de partículas Ξ_k é obtido por meio de reamostragem do conjunto de partículas geradas a partir de (2.32). Neste processo, cada partícula é acompanhada de um peso, mais conhecido como fator de importância $w_k^{[m]}$ que denota a probabilidade desta partícula ser amostrada:

$$w_k^{[m]} = \frac{p(\boldsymbol{\xi}_{1:k}^{[m]} \,|\, \mathbf{y}_{1:k}, \mathbf{u}_{1:k}, \mathbf{c}_{1:k})}{p(\boldsymbol{\xi}_{1:k}^{[m]} \,|\, \mathbf{y}_{1:k-1}, \mathbf{u}_{1:k}, \mathbf{c}_{1:k-1})}. \quad (2.33)$$

O trabalho em [10] mostra com mais detalhes os cálculos do peso de cada partícula.

Por outro lado, a f.d.p. $p(\mathbf{m}_{i,k}|\boldsymbol{\xi}_{1:k}, \mathbf{y}_{1:k}, \mathbf{u}_{1:k}, \mathbf{c}_{1:k})$ em (2.30), é estimada com o auxílio do já conhecido filtro de Kalman. Visto que estas estimativas são condicianadas à pose do robô, cada partícula do conjunto Ξ_k implementa um filtro de Kalman individual. Mais especificamente, o problema completo de SLAM dado em (2.30) é representado pelo seguinte conjunto de partículas:

$$\Xi_k = \{\boldsymbol{\xi}_{1:k}^{[m]}, \boldsymbol{\mu}_1^{[m]}, \boldsymbol{\Sigma}_1^{[m]}, ..., \boldsymbol{\mu}_{n_k}^{[m]}, \boldsymbol{\Sigma}_{n_k}^{[m]}\}_m, \tag{2.34}$$

em que $\boldsymbol{\mu}_i^{[m]}, \boldsymbol{\Sigma}_i^{[m]}$ para $i = \{1, 2, ..., n_k\}$, são a média e matriz de covariâncias da f.d.p. Gaussiana que representa o *i-ésimo landmark* relacionado com a *m-ésima* partícula.

A estimativa da posição do *i-ésimo landmark* é obtida levando em consideração se $\mathbf{c}_k = i$ ou $\mathbf{c}_k \neq i$, ou seja, se $\mathbf{s}_i$ foi observado no instante de tempo k. Para $\mathbf{c}_k = i$, obtem-se:

$$\begin{aligned} p(\mathbf{m}_k|\boldsymbol{\xi}_{1:k}, \mathbf{y}_{1:k}, \mathbf{u}_{1:k}, \mathbf{c}_{1:k}) &\overset{Bayes}{\propto} \eta\ p(\mathbf{y}_k|\mathbf{m}_k, \boldsymbol{\xi}_{1:k}, \mathbf{y}_{1:k-1}, \mathbf{u}_{1:k}, \mathbf{c}_{1:k}) p(\mathbf{m}_k|\boldsymbol{\xi}_{1:k}, \mathbf{y}_{1:k-1}, \mathbf{u}_{1:k}, \mathbf{c}_{1:k}) \\ &\overset{Markov}{=} p(\mathbf{y}_k|\mathbf{m}_k, \boldsymbol{\xi}_k, \mathbf{c}_k) p(\mathbf{m}_k|\boldsymbol{\xi}_{1:k-1}, \mathbf{y}_{1:k-1}, \mathbf{u}_{1:k-1}, \mathbf{c}_{1:k-1}) \end{aligned} \tag{2.35}$$

Para $\mathbf{c}_k \neq i$, a f.d.p. simplesmente permanece inalterada:

$$p(\mathbf{m}_k|\boldsymbol{\xi}_{1:k}, \mathbf{y}_{1:k}, \mathbf{u}_{1:k}, \mathbf{c}_{1:k}) = p(\mathbf{m}_k|\boldsymbol{\xi}_{1:k-1}, \mathbf{y}_{1:k-1}, \mathbf{u}_{1:k-1}, \mathbf{c}_{1:k-1}). \tag{2.36}$$

No algoritmo FastSLAM a equação (2.35) é implementada por um filtro de Kalman estendido (FKE). Logo a solução do FastSLAM é similar àquela dada tradicionalmente pelo FKE-SLAM [95], no sentido em que também aproxima o modelo de medição dado em (2.28) por uma função de probabilidade Gaussiana linear. Porém uma diferença siginificativa entre estes dois algoritmos é que o processo de atualização das partículas no FastSLAM envolve uma função Gaussiana de duas dimensões (dois parâmetros para a posição de cada *landmark*), enquanto que no FKE-SLAM este passo envolve uma f.d.p. de dimensão $2N + 3$ (N *landmarks* e três parâmetros para a pose do robô). Conclui-se que esta etapa tem complexidade computacional constante para o FastSLAM [10] ao passo que este mesmo procedimento requer um tempo de computação de ordem quadrática com relação a N quando se trata do FKE-SLAM.

2.1.4.2 Uma implementação eficiente para o FastSLAM

Conforme descrito até aqui, o algoritmo FastSLAM requer uma complexidade linear com relação ao número de *landmarks* para cada passo de atualização das partículas. Isto se deve principalmente à etapa de reamostragem, ou seja, cada vez que uma partícula é adicionada no conjunto Ξ_k todas as partículas devem ser copiadas. Se cada partícula contem N *landmarks* este procedimento requer um tempo de computação da ordem de $O(MN)$. Contudo, muitas destas operações podem ser evitadas se o conjunto de funções de probabilidade em cada partícula for representado por uma estrutura em árvore. A Figura 2.3 mostra esta representação

para uma única partícula no caso de 8 *landmarks*. Os parâmetros de cada função $\boldsymbol{\mu}_i^{[m]}$ e $\boldsymbol{\Sigma}_i^{[m]}$ são colocados como folhas da árvore (nós que não possuem filhos). Observa-se que o acesso a qualquer um destes nós requer um tempo de ordem logaritmica com relação a N, isto é, $O(logN)$.

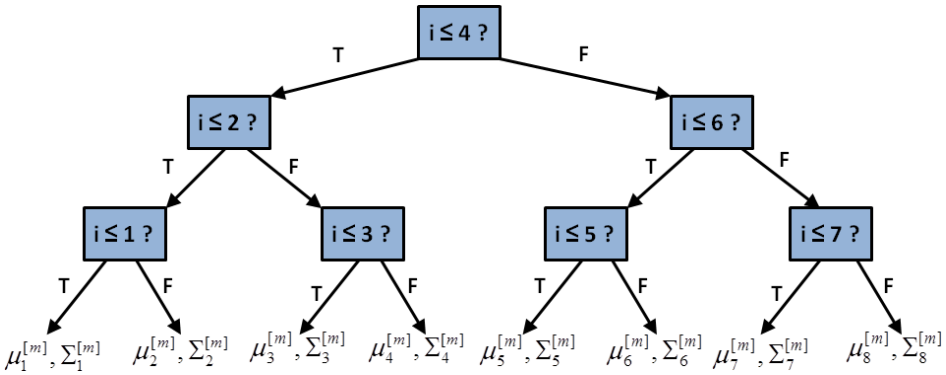


Figura 2.3: Estrutura em árvore representando $N = 8$ *landmarks* para uma única partícula.

Desta forma, no momento da geração de uma nova partícula, apenas um caminho tem que ser modificado na estrutura em árvore. Considere o exemplo mostrado na Figura 2.4. Assume-se que $\mathbf{c}_k = 3$, ou seja, apenas os parâmetros $\boldsymbol{\mu}_3^{[m]}$ e $\boldsymbol{\Sigma}_3^{[m]}$ serão atualizados. Ao invés de gerar uma nova árvore completa, apenas um novo caminho é criado, que leva aos parâmetros da função de probabilidade para a qual $\mathbf{c}_k = 3$. Assim, a criação deste novo ramo da árvore tem complexidade computacional de ordem logarítmica com relação a N. Portanto, tanto a geração de uma nova partícula quanto o acesso a qualquer ramo da árvore de dados tem um tempo de computação $O(logN)$. Dado que em cada passo de atualização M novas partículas são criadas, a solução por FastSLAM tem uma complexidade computacional de $O(MlogN)$.

2.1.4.3 Associação de dados no FastSLAM

Grande parte das situações em SLAM requer a solução do problema de associação de dados, ou seja, em muitas situações do mundo real os *landmarks* não são previamente identificados e o número total deles também não pode ser obtido. Portanto, durante o processo de estimação em SLAM faz-se necessário obter um conjunto de correspondências entre as observações atuais e os *landmarks* previamente adicionados no mapa. Tradicionalmente, este problema é resolvido por meio de uma medida probabilística que representa o grau de similaridade entre duas estruturas do ambiente. Esta medida é geralmente conhecida como distância de Mahalanobis.

No FastSLAM o processo de associação de dados é realizado tomando-se cada partícula individualmente, isto é,

$$\mathbf{c}_k^{[m]} = \text{argmax}_{\mathbf{c}_k} p(\mathbf{y}_k | \boldsymbol{\xi}_k^{[m]}, \mathbf{c}_k). \tag{2.37}$$

Como consequência, cada partícula pode ter um valor diferente para a variável de correspon-

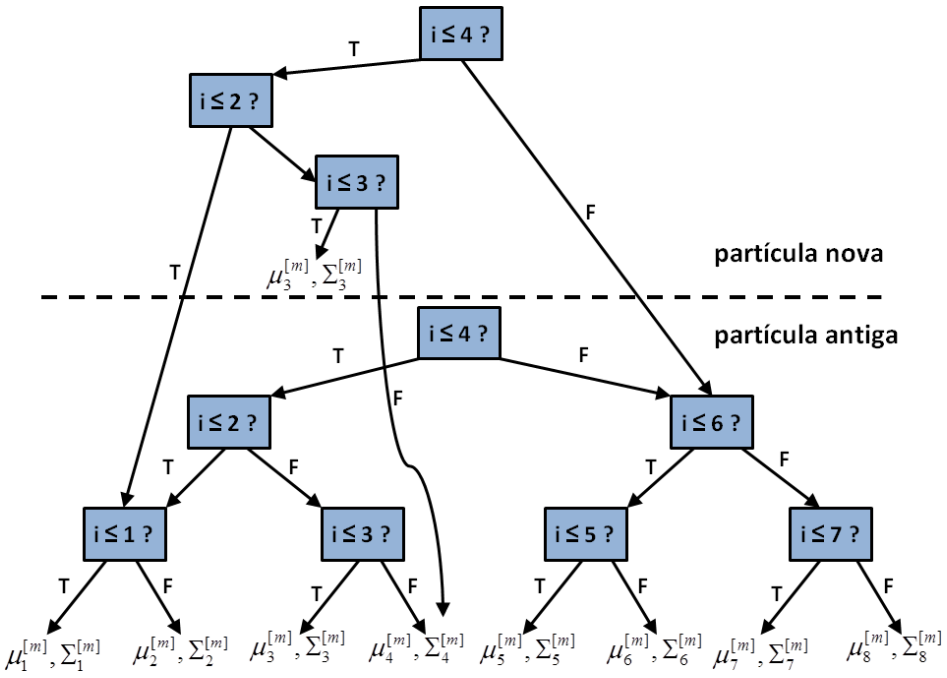


Figura 2.4: Geração de uma nova partícula a partir de uma antiga modificando apenas um caminho na estrutura em árvore. A nova particula recebe apenas uma parte da árvore.

dência $\mathbf{c}_k^{[m]}$, mais ainda, cada partícula pode possuir uma quantidade diferente de *landmarks* em seus respectivos mapas. Este fato constitui a principal diferença para a abordagem do FKE-SLAM para o qual o processo de associação de dados é determinado apenas uma vez para cada observação. Devido à sua capacidade de buscar múltiplas correspondências simultâneamente, o FastSLAM é menos sensível ao problema de falsas associações que comprometem de forma catastrófica a abordagem por FKE-SLAM [96]. Partículas com correspondências incorretas são mais prováveis de serem eliminadas no processo de reamostragem, o que faz do FastSLAM um algoritmo bem mais robusto com relação a falsas correspondências.

2.2 FILTRAGEM DE SISTEMAS A MÚLTIPLOS MODELOS

No Capítulo 1 define-se sistemas híbridos como sistemas dinâmicos que combinam variáveis de estado contínuas e discretas. Esta é uma definição bastante abrangente visto que sua aplicação pode ser útil em um incontável número de situações distintas.

Um caso particular de sistemas híbridos em que uma das possíveis variáveis de estado discreta denota o modo de operação do sistema e define o modelo matemático que descreve a forma como a porção contínua do vetor de estados evolui é comumente referido na literatura como sistemas a múltiplos modelos (MM). Seguindo este paradigma, o trabalho em [97] apresenta uma formulação em que vários modelos são projetados para representar os di-

ferentes padrões de comportamento do sistema (também denominados modos de operação). Desta forma, o vetor de estados do sistema é estimado com base na combinação de estimativas de vários filtros calculados simultaneamente correspondendo aos diferentes modos do sistema. Uma característica importante desta formulação é a existência de saltos entre esses modos de operação significando mudanças de modelos. Os primeiros trabalhos não consideravam transições entre os modos, contudo os trabalhos mais recentes iniciados em [98] propõem algoritmos que levam em consideração essas transições. Uma revisão bastante atual a respeito de sistemas híbridos é feita em [99].

Muitas aplicações práticas, tais como circuitos eletrônicos chaveados ou sistemas espaciais, possuem um comportamento dinâmico bastante complexo, sendo que sua formulação matemática mostra-se muito difícil levando a resultados ruins [100]. A principal preocupação deste capítulo está na apresentação do filtro IMM, um algoritmo tradicionalmente empregado em aplicações envolvendo sistemas dinâmicos híbridos conforme ressaltado por Boers e Driessen em [101]. Esta técnica de filtragem foi introduzida por Henk Blom em [50, 53] e, na época, trouxe notáveis inovações principalmente em seu passo de combinação de estimativas.

2.2.1 O filtro *Interacting Multiple Models* (IMM)

O filtro IMM é um estimador subótimo que possui uma boa relação custo-benefício em termos de complexidade computacional e desempenho na estimação de estados em sistemas híbridos [99]. A proposição do IMM feita em [50, 53] foi motivada por um problema de rastreamento de aeronaves em um Sistema de Controle de Tráfego Aéreo (SCTA). O algoritmo IMM mostra-se bastante eficaz em estimar o estado de sistemas a múltiplos modelos por meio da "mistura"das estimativas de seu conjunto de estimadores, o que será melhor explicado adiante.

A importância do filtro IMM para a comunidade científica fica evidente através da quantidade de trabalhos que este filtro influenciou e que se dedicaram sobre o desenvolvimento de métodos de avaliação e melhoria deste filtro [102, 103, 104]. Uma aplicação bastante comum do filtro IMM é no rastreamento de alvos. [105, 49, 106, 107, 108]. Entretanto, filtragem de sistemas híbridos possui um contexto mais abrangente. Por exemplo, o trabalho feito em [109] mostra que o filtro IMM pode ser usado em aplicações para detecção e correção de falhas em um sistema de localização de um VANT. Também esta dissertação propõe o uso de uma variação do IMM na resolução do problema de SLAM em robótica móvel, o que reafirma a abrangência da sua aplicabilidade.

2.2.1.1 Formulação matemática

O filtro IMM foi desenvolvido para ser uma nova estratégia de filtragem de Sistemas Lineares com Saltos Markovianos (MJLS, do inglês *Markov Jump Linear Systems*) [50, 53]. Considera-se o seguinte modelo em espaço de estados:

$$x_k = A_{m_{k-1}} x_{k-1} + B_{m_{k-1}} w_{k-1} \tag{2.38}$$

$$y_k = H_{m_k} x_k + G_{m_k} v_{k-1} \tag{2.39}$$

em que $k \in \mathbb{N}$ é o número da amostra obtida no instante $k\tau$ onde τ é o período de amostragem; x_k é o vetor de estados contínuos do sistema; $m_k \in \mathbb{M} \triangleq \{1, 2, \cdots, M\}$ é a porção discreta do estado do sistema (modo) e evolui segundo uma Cadeia de Markov (CM) com Matriz de Probabilidade de Transição (MPT) $\Pi : \mathbb{R}^M \times \mathbb{R}^M \to \mathbb{R}^M$; $A_{m_{k-1}}$, $B_{m_{k-1}}$, H_{m_k}, G_{m_k} são matrizes conhecidas dependentes do modo; y_k é o vetor de medições dependentes do modo; e $w_{k-1} \sim N(0, Q_{k-1})$ e $v_k \sim N(0, R_k)$ são processos de ruído branco Gaussiano descorrelacionados, com média nula e matrizes de covariâncias Q_{k-1} e R_k, respectivamente.

De acordo com as equações apresentadas em (2.38)-(2.39) um conjunto de M modelos em espaço de estados são usados para descrever a dinâmica do sistema. A patir desta formulação, almeja-se obter estimativas do vetor de estado contínuo do sistema $\hat{x}_k$, de sua matriz de covariâncias associada $\hat{P}_k$ e do vetor de probabilidades dos modos $\hat{p}(m_k)$ por meio de um algoritmo bem definido e partindo de um conjunto de medições corrompidas por ruído $y_{1:k} = \{y_1, y_2, ..., y_k\}$. Assume-se que a MPT

$$\Pi = \{\pi_{i,j}\}, \;\; \pi_{i,j} = \Pr\{m_k = j | m_{k-1} = i\}, \;\; i, j \in \mathbb{M}, \forall k \in \mathbb{N} \tag{2.40}$$

seja conhecida, contudo as transições entre os modos da CM $m_{k-1} = i \longrightarrow m_k = j$ podem ser desconhecidas. O algoritmo de filtragem proposto por [50] contem cinco etapas fundamentais:

i Estimação das probabilidades dos modos por predição de acordo com a cadeia de Markov;

$$\Pr(m_{k-1}|y_{1:k-1}) \xrightarrow{\text{modelo da CM}} \Pr(m_k|y_{1:k-1})$$

ii Ajustar a distribuição de x_{k-1} a partir da atualização de m_k;

$$p(x_{k-1}|m_{k-1}, y_{1:k-1}) \xrightarrow{\text{"mistura"de estimativas}} p(x_{k-1}|m_k, y_{1:k-1})$$

iii Propagar as estimativas x_{k-1} por predição;

$$p(x_{k-1}|m_k, y_{1:k-1}) \xrightarrow{\text{Predição}} p(x_k|m_k, y_{1:k-1})$$

iv Corrigir a estimativa de x_k por meio da nova medição;

$$p(x_k|m_k, y_{1:k-1}) \xrightarrow{\text{Correção}} p(x_k|m_k, y_{1:k})$$

v Corrigir a probabilidade dos modos;

$$\Pr(m_k|y_{1:k-1}) \xrightarrow{\text{teorema de Bayes}} \Pr(m_k|y_{1:k})$$

Conforme proposto em [50] as equações correspondentes a cada etapa podem ser resumidas no algoritmo descrito a seguir.

IMM (*Interacting Multiple Models*)[50]

Sejam $\hat{x}_k^{(i)}$ e $\hat{P}_k^{(i)}$, i ∈ {1,2,...,M}, o vetor de estados e sua matriz de covariâncias associada que correspondem ao filtro seguindo o modo $m_k = i$ no k-ésimo instante amostral.

Seja também y_k o vetor atual de saída do sistema. Definindo $\hat{p}^{(i)}(m_k) = \Pr(m_k = i|y_{1:k})$ e assumindo condições iniciais

$$\hat{p}^{(i)}(m_0), \hat{x}_0^{(i)}, \hat{P}_0^{(i)}, \ i \in \{1, 2, ..., M\},$$

um conjunto de M filtros é usado para estimar o vetor de estados x_k dado em (2.38)-(2.39), cada um deles seguindo um modo diferente do sistema, de acordo com os seguintes passos:

i Predição das probabilidades dos modos pela cadeia de Markov

$$\overline{p}^{(i)}(m_k) = \sum_{j=1}^{M} \pi_{j,i} \hat{p}^{(j)}(m_{k-1}) \tag{2.46}$$

ii Mistura das estimativas

$$\underline{x}_{k-1}^{(i)} = \sum_{j=1}^{M} \frac{\pi_{j,i} \hat{p}^{(j)}(m_{k-1}) \hat{x}_{k-1}^{(j)}}{\overline{p}^{(i)}(m_k)}, \tag{2.47}$$

$$\underline{P}_{k-1}^{(i)} = \sum_{j=1}^{M} \frac{\pi_{j,i} \hat{p}^{(j)}(m_{k-1}) \left[\hat{P}_{k-1}^{(j)} + \left(\hat{x}_{k-1}^{(j)} - \underline{x}_{k-1}^{(i)} \right) \left(\hat{x}_{k-1}^{(j)} - \underline{x}_{k-1}^{(i)} \right)^T \right]}{\overline{p}^{(i)}(m_k)} \tag{2.48}$$

iii Predição das estimativas

$$\overline{x}_k^{(i)} = A_{m_{k-1}} \underline{x}_{k-1}^{(i)}, \tag{2.49}$$

$$\overline{P}_k^{(i)} = A_{m_{k-1}} \underline{P}_{k-1}^{(i)} A_{m_{k-1}}^T + B_{m_{k-1}} Q_{m_{k-1}} A_{m_{k-1}}^T. \tag{2.50}$$

iv Correção das estimativas

$$
\begin{aligned}
K_k^{(i)} &= \overline{P}_k^{(i)} H_{m_k}^T \left(H_{m_k} \overline{P}_k^{(i)} H_{m_k}^T + G_{m_k} R_{m_k} G_{m_k}^T \right), \\
\hat{x}_k^{(i)} &= \overline{x}_k^{(i)} + K_k^{(i)} \left(y_k - H_{m_k} \overline{x}_k^{(i)} \right), \\
\hat{P}_k^{(i)} &= \left(\mathbb{I} - K_k^{(i)} H_{m_k} \right) \overline{P}_k^{(i)} \left(\mathbb{I} - K_k^{(i)} H_{m_k} \right)^T + \left(K_k^{(i)} G_{m_k} \right) R_{m_k} \left(K_k^{(i)} G_{m_k} \right)^T,
\end{aligned}
$$

em que $\mathbb{I}$ é a matriz identidade de dimensões apropriadas.

v Correção da probabilidade de cada modo

`For i=1,2,...,M`

$$
\begin{aligned}
\vartheta_k^{(i)} &= y_k - H_{m_k} \overline{x}_k^{(i)}, \\
\Sigma_{\vartheta_k^{(i)}} &= H_{m_k} \overline{P}_k^{(i)} H_{m_k}^T + G_{m_k} R_{m_k} G_{m_k}^T, \\
\hat{p}^{(i)}(m_k) &= \frac{\overline{p}^{(i)}(m_k)}{c_i \det\left(\Sigma_{\vartheta_k^{(i)}} \right)^{\frac{1}{2}}} \exp\left\{ -\frac{1}{2} \left(\vartheta_k^{(i)} \right)^T \left(\Sigma_{\vartheta_k^{(i)}} \right) \vartheta_k^{(i)} \right\},
\end{aligned}
$$

`end For`

$$
\begin{aligned}
\gamma_p &= \sum_{j=1}^{M} \hat{p}^{(j)}(m_k), \\
\hat{p}(m_k) &= [\hat{p}^1(m_k), \cdots, \hat{p}^M(m_k)]^T \left(\frac{1}{\gamma_p} \right),
\end{aligned}
$$

em que c_i é uma constante de normalização.

vi Geração das saídas

$$
\begin{aligned}
\hat{x}_k &= \sum_{i=1}^{M} \hat{p}^{(i)}(m_k) \hat{x}_k^{(i)}, \\
\hat{P}_k &= \sum_{i=1}^{M} \hat{p}^{(i)}(m_k) \left[\hat{P}_k^{(i)} + \left(\hat{x}_k^{(i)} - \hat{x}_k \right) \left(\hat{x}_k^{(i)} - \hat{x}_k \right)^T \right].
\end{aligned}
$$

Uma interpretação gráfica do algoritmo de estimação do filtro IMM está mostrada na Figura 2.5.

2.2.2 Estimação da matriz de probabilidade de transição

Muitos trabalhos, entre eles relativos à estimação de estados no contexto de sistemas com saltos Markovianos [105, 50, 53, 101, 110] pressupõem que a probabilidade de transição entre os modos de operação é conhecida a priori, ou seja, a matriz de probabilidades de transição Π é um parâmetro dado. Entretanto, em grande parte dos casos esta hipótese é irreal [111]. Fixar um valor prévio para Π pode degradar o desempenho do filtro, portanto a

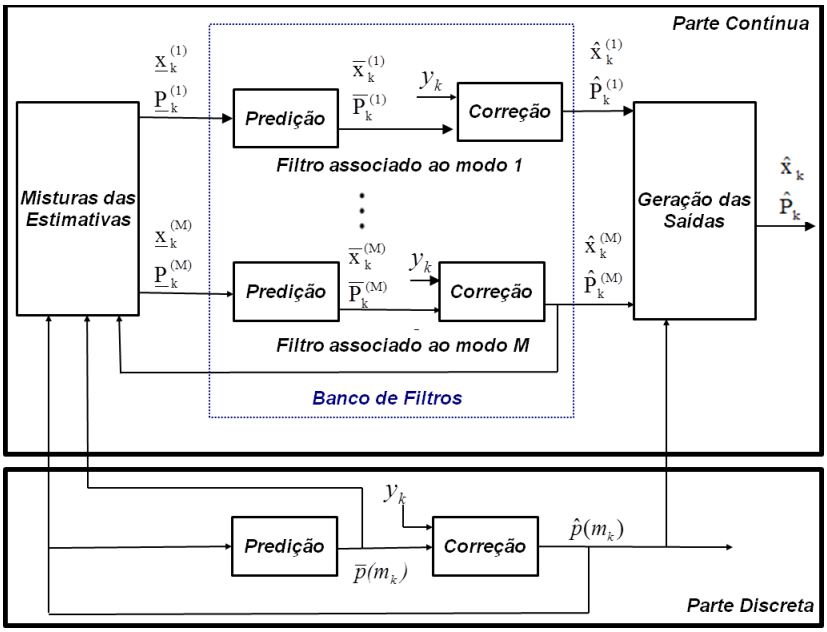


Figura 2.5: Diagrama esquemático das principais etapas envolvidas no algoritmo IMM.

estimação *online* da MPT baseada em um conjunto de observações $y_{1:k}$ torna-se um recurso bastante interessante neste caso.

O trabalho apresentado em [112] realiza estimação *online* de MPTs desconhecidas e não-estacionárias. Conforme o algoritmo descrito neste trabalho, cada linha de Π é modelada seguindo uma distribuição *a priori* de Dirichlet e deriva um estimador Bayesiano baseado no fato de que a distribuição de Dirichlet é conjugada à distribuição multinomial. Contudo, nesta dissertação, utilizou-se o algoritmo *Quasi-Bayesiano* descrito em [111] pois o estimador apresentado em [112] assume que os modos do sistema podem ser perfeitamente observados, o que pode não ser verdade.

Algoritmo Quasi-Bayesiano[111]
Seja π_i, $i \in \{1, 2, ..., M\}$, *a i-ésima linha de* Π, *i.e.,* $\Pi = [\pi_1^T \ \pi_2^T \ \cdots \ \pi_M^T]^T$. *Defina*

$$\begin{aligned}
\mu_i(k) &= \Pr(m_k = i|\hat{\Pi}_{k-1}, y_{1:k-1}), \\
\hat{p}(m_k) &= [\mu_1(k) \ \mu_2(k) \ \cdots \ \mu_M(k)]^T, \\
\lambda_i(k) &= p(y_k|m_k = i, \hat{\Pi}_{k-1}, y_{1:k-1}), \\
\Lambda(k) &= [\lambda_1(k) \ \lambda_2(k) \ \cdots \ \lambda_M(k)]^T, \\
\eta_i(k) &= \frac{\mu_i(k-1)}{\hat{p}(m_{k-1})^T \hat{\Pi}_{k-1} \Lambda(k)}.
\end{aligned}$$

Assumindo que $\pi_i(k)$ *segue uma distribuição de Dirichlet com parâmetros*

$$\alpha_i(k) = [\alpha_{i,1}(k) \ \alpha_{i,2}(k) \ \cdots \ \alpha_{i,M}(k)]$$

no k-ésimo instante amostral e tendo como parâmetros a priori

$$\begin{aligned}
\alpha_i(0) &= [\alpha_{i,1}(0)\ \alpha_{i,2}(0)\ \cdots\ \alpha_{i,M}(0)], \\
\gamma_i(0) &= \sum_{j=1}^{M} \alpha_{i,j}(0), \quad \alpha_{i,j}(0) \geq 0, \\
\hat{\pi}_i(0) &= \frac{1}{\gamma_i(0)} \alpha_i(0),
\end{aligned}$$

a matriz $\hat{\Pi}(k)$ *pode ser estimada recursivamente conforme a seguir:*

```
For i=1,2,...,M
   For j=1,2,...,M
```

$$\begin{aligned}
g_{i,j}(k) &= 1 + \eta_i(k) \left[\lambda_j(k) - \hat{\pi}_i(k-1)^T \Lambda(k)\right] \\
\alpha_{i,j}(k) &= \alpha_{i,j}(k-1) + \frac{\alpha_{i,j}(k-1) g_{i,j}(k)}{\sum_{l=1}^{M} \alpha_{i,l}(k-1) g_{i,l}(k)} \\
\hat{\pi}_{i,j}(k) &= \frac{1}{k + \gamma_i(0)} \alpha_{i,j}(k)
\end{aligned}$$

```
   end For
```

$$\hat{\pi}_i(k) = [\hat{\pi}_{i,1}(k)\ \cdots\ \hat{\pi}_{i,M}(k)]$$

```
end For
```

$$\hat{\Pi}_i(k) = \left[\hat{\pi}_1^T\ \hat{\pi}_2^T\ \cdots\ \hat{\pi}_M^T\right]$$

Caso não haja informação prévia acerca de Π, pode-se escolher $\alpha_i(0) = [1\ 1\ \cdots\ 1]$, resultando em $\hat{\pi}_{i,j}(0) = 1/M$.

3 ABORDAGEM HÍBRIDA PARA O PROBLEMA DE SLAM

3.1 INTRODUÇÃO

Tradicionalmente, em SLAM, tanto a pose do robô quanto os elementos do mapa (estruturas do ambiente) são estimados à medida que o robô explora o seu espaço de trabalho. Neste sentido, novas estruturas são incorporadas no mapa, bem como estruturas adicionadas previamente são atualizadas por meio de observações feitas localmente [113, 114, 115, 116].

O principal problema com este tipo de abordagem é que erros cometidos durante o processo de estimação são propagados para a estimativa da pose do robô bem como para as estimativas das estruturas do mapa levando o algoritmo à divergência. Além disso, os dados dos sensores usados previamente para atualização do mapa são descartados, prejudicando a detecção de laços fechados [16, 117].

A formulação do problema de SLAM adotada neste trabalho segue uma linha mais recente de pesquisa nesta área no sentido de reconstruir a trajetória realizada pelo robô [34, 36, 38]. Desta forma, são estimados tanto a pose atual do robô, bem como as poses no momento da aquisição de dados anteriores. Para tanto, faz-se necessário manter armazenados dados dos sensores associados aos instantes de cada pose. Entende-se, portanto, que não são mantidas covariâncias entre os elementos do mapa e a pose atual do robô, o que denota uma visão desacoplada do problema de SLAM.

Neste capítulo, descreve-se o problema de SLAM através do paradigma de modelagem de sistemas dinâmicos híbridos. Neste contexto, à medida que o robô navega pelo ambiente os dados dos sensores são armazenados formando um conjunto de submapas locais e o que se estima são as poses do robô ao longo da trajetória bem como de qual subregião do ambiente (submapa local) o robô está tomando as medidas. Segundo esta formulação, em um contexto de estimação estocástica, o processo de filtragem leva em conta um vetor de estados composto tanto por variáveis contínuas (poses do robô) quanto por uma variável discreta (índice da subregião da qual as medidas são feitas).

Este capítulo representa o cerne desta dissertação em que é apresentado uma nova proposta de solução do problema de SLAM, denominada Hybrid-SLAM. A fundamentação teórica apresentada no Capítulo 2.2 serve como base a este capítulo e apresenta uma possível aplicação prática para seus resultados. Por fim, propõe-se uma nova versão do algoritmo de filtragem IMM aplicado ao problema de SLAM. No Capítulo 4 será mostrado através de algumas simulações que esta modificação levou a um significativo ganho de desempenho.

3.2 ESTIMAÇÃO DE ESTADOS EM UM SISTEMA DINÂMICO HÍBRIDO

Antes da apresentação do modelo híbrido específico para o problema de SLAM, esta seção retoma a modelagem descrita na Seção 2.2.1.1 afim de dar um aspecto mais detalhado do que o do Capítulo 2.2 para alguns aspectos envolvidos na filtragem estocástica de sistemas híbridos.

O cojunto de equações a seguir pode ser usado para descrever um sistema híbrido a tempo discreto que combina variáveis de estado contínuas e discretas:

$$x_k = f_{m_{k-1}}(x_{k-1}, u_{k-1}, w_{k-1}), \tag{3.1}$$

$$y_k = h_{m_k}(x_k, v_k), \tag{3.2}$$

em que x_k é o vetor de estados contínuos; $m_k \in \mathbb{M} \triangleq \{1, 2, ..., M\}$ é o estado discreto do sistema (modo), podendo assumir M diferentes valores; $f_{m_{k-1}}$ é uma função de evolução do processo possivelmente não-linear e dependente do modo; h_{m_k} e y_k são, respectivamente, a função de medição e o vetor de medições; u_{k-1} é o vetor de entradas; e w_{k-1} e v_k representam ruídos associados a cada um dos processos. As equações em (3.1)-(3.2) definem um conjunto de M diferentes funções $f_{m_{k-1}}$ e h_{m_k} em espaços de estados para descrever a dinâmica do sistema.

Através das equações descritas em (3.1)-(3.2) define-se um modelo dinâmico para cada modo de operação do sistema. Além disso, faz-se necessário descrever a maneira pela qual ocorrem as transições entre estes diferentes modos de operação. Uma possibilidade de formular estas transições é pela utilização do modelo Markoviano que tem sido adotado em vários trabalhos como [118, 49, 50]. Segundo este modelo, as transições entre os modos dependem apenas do modo atual do sistema e ocorrem conforme uma Matriz de Probabilidades de Transição (MPT) que pode ser variante no tempo. Desta forma, assume-se então que a variável m_k segue uma cadeia de Markov com vetor inicial de probabilidades possivelmente desconhecido $p(m_0)$ e MPT também possivelmente desconhecida

$$\Pi_k = \{\pi_{i,j}\}, \;\; \pi_{i,j} = \Pr\{m_k = j | m_{k-1} = i\}, \;\; i, j \in \mathbb{M}, \;\; \forall k \in \mathbb{N}. \tag{3.3}$$

Portanto, uma vez que (3.1)-(3.2) tenham sido definidas e as transições entre os modos tenham sido modeladas, o principal objetivo no processo de estimação de sistemas híbridos é obter a função densidade de probabilidade (f.d.p.) *a posteriori* conjunta de x_k e m_k baseado em um sequência $y_{1:k} = \{y_1, y_2, ..., y_k\}$ de medidas ruidosas geradas de acordo com (3.2). Esta f.d.p. pode ser decomposta em duas partes conforme a seguir:

$$p(x_k, m_k | y_{1:k}) = p(x_k | m_k, y_{1:k}) \Pr(m_k | y_{1:k}), \tag{3.4}$$

em que ambos x_k e m_k podem não ser diretamente mensuráveis. Desta forma, pode-se estimar a f.d.p. em (3.4) como dois problemas distintos: estimar $p(x_k | m_k, y_{1:k})$ que é a f.d.p.

a posteriori de x_k condicionada ao modo m_k do sistema; e estimar $\Pr(m_k|y_{1:k})$ que é a probabilidade condicional modal discreta e não depende de x_k.

O termo representado por $\Pr(m_k|y_{1:k})$ pode ser desenvolvido usando o simples resultado do Teorema de Bayes, visto que

$$\Pr(m_k|y_{1:k}) = \frac{p(y_k|m_k, y_{1:k-1}) \Pr(m_k|y_{1:k-1})}{p(y_k|y_{1:k-1})} \tag{3.5}$$

Analisando melhor o resultado mostrado em (3.5) pode-se chegar a um entendimento bastante intuitivo de cada uma das suas partes. Primeiramente, nota-se que para o termo $p(y_k|m_k, y_{1:k-1})$ o modo m_k é dado, isto é, pressupõe-se que ele seja perfeitamente conhecido. Isto implica que esta f.d.p. é simplesmente a verossimilhança da medida mais atual y_k assumindo que um determinado modo de operação do sistema esteja em vigor. A complexidade desta expressão depende exclusivamente da função de medição h_{m_k}. Ainda mais, se v_k for suposto Gaussiano em (3.2) o termo $p(y_k|m_k, y_{1:k-1})$ torna-se a famosa expressão da f.d.p. exponencial Normal.

Usando a informação do modelo matemático descrito em (3.1)-(3.2) pode-se calcular um valor predito da saída do sistema. A verossimilhança de y_k pode ser entendida como uma medida de quão próxima está esta observação experimental deste valor predito. Além disso, percebe-se que na f.d.p. $p(y_k|m_k, y_{1:k-1})$ são dadas apenas as medições $y_{1:k-1}$, excluindo-se a medição mais atual da saída do sistema. Tal fato implica que para o cálculo da verossimilhança de y_k a melhor estimativa que pode-se ter do estado do sistema é aquela predita pelo modelo de evolução em (3.1), denotada por $\overline{x}_k$. Em um contexto de filtragem estocástica, um método bastante difundido para cálculo desta estimativa predita $\overline{x}_k = E\{x_k|y_{1:k-1}\}$ é o algoritmo de estimação dado pelo filtro de Kalman ou alguma das suas variações. Portanto, de posse do valor de $\overline{x}_k$ e sabendo o modo do sistema m_k calcula-se a saída predita do sistema pela aplicação da função de medição (3.2) conforme a seguir

$$\overline{y}_k = h_{m_k}(\overline{x}_k, E\{v_k\}). \tag{3.6}$$

Se o ruído de medição v_k é suposto aditivo e Gaussiano de média nula, $v_k \sim \mathcal{N}(0, R_{v_k})$, (3.6) simplifica-se para

$$\overline{y}_k = h_{m_k}(\overline{x}_k) + 0 = h_{m_k}(\overline{x}_k). \tag{3.7}$$

A adequação do modo m_k em predizer a saída y_k do sistema pode ser avaliada com base na seguinte distância:

$$\nu = y_k - \overline{y}_k \tag{3.8}$$

entre a saída atual e a saída predita do sistema que é chamada termo de inovação. Esta diferença praticamente nunca é zero, devido ao efeito conjunto de erros de modelagem e ruídos dos sensores.

Além disso, também é necessário que se saiba a incerteza Σ_ν associada à diferença em (3.8). Por exemplo, se Σ_ν for "pequena"(o que indica pouca dispersão em torno dos valores

médios), um termo de inovação de grande magnitude pode indicar que o modelo m_k não é capaz de predizer corretamente $\overline{y}_k$, levando a crer que talvez outro modo do sistema esteja em vigor. Geralmente, estimar Σ_ν é uma tarefa complicada devido a fortes não-linearidades presentes na função de medição h_{m_k} o que dificulta a propagação de incertezas dos argumentos para a saída da função. No desenvolvimento deste trabalho utilizou-se um filtro estocástico baseado em linearização, para o qual a propagação de incertezas para estimação de Σ_ν foi feita por meio da seguinte equação

$$\Sigma_\nu = \left(\frac{\partial h_{m_k}(x_k)}{\partial x_k}\bigg|_{\overline{x}_k} \right) \overline{P}_k \left(\frac{\partial h_{m_k}(x_k)}{\partial x_k}\bigg|_{\overline{x}_k} \right)^T + R_{v_k} \tag{3.9}$$

em que $\partial h_{m_k}(x_k)/\partial x_k$ é a matriz Jacobiana da função de medição avaliada na estimativa predita $\overline{x}_k$ e $\overline{P}_k$ é a matriz de covariâncias associada a esta estimativa.

Portanto, se a dimensão do vetor de saída associado a $h_{m_k}(x_k, v_k)$ por denotado por n_y, tem-se que a expressão fechada para o cálculo da verossimilhança de y_k para o caso em que v_k é Gaussiano

$$p(y_k|m_k, y_{1:k-1}) = \frac{1}{(2\pi)^{\frac{n_y}{2}} \det(\Sigma_\nu)^{\frac{1}{2}}} \exp\left\{ -\frac{1}{2} \nu^T \Sigma_\nu^{-1} \nu \right\}. \tag{3.10}$$

Mais adiante, será mostrado que a determinação da verossimilhança de y_k pode ser usada para auxiliar uma etapa fundamental do problema de SLAM que é o passo de associação de dados. Tradicionalmente em SLAM, esta etapa requer um mecanismo de verificação externo ao filtro o que não ocorre na abordagem por sistemas híbridos que já incorpora estes testes de verificação no cálculo da verossimilhança e naturalmente determina o conjunto de equações de medição mais adequado sem a necessidade de verificações adicionais.

O próximo termo a ser analisado é a probabilidade modal discreta $\Pr(m_k|y_{1:k-1})$. Este termo representa uma estimativa da probabilidade modal do sistema m_k no instante atual e uma vez que as probabilidades de transição dadas pelo modelo em (3.3) dependem apenas do modo atual do sistema, o cálculo de $\Pr(m_k|y_{1:k-1})$ necessita apenas da matriz Π_{k-1}, a MPT do instante anterior, e do vetor de probabilidade dos modos no instante anterior $p(m_{k-1})$. Então escrevendo-se

$$\overline{p}(m_{k-1}) = [\Pr(m_k = 1|y_{1:k-1}), \Pr(m_k = 2|y_{1:k-1}), ..., \Pr(m_k = M|y_{1:k-1})]^T \tag{3.11}$$

como sendo o vetor de probabilidades preditas dos modos no instante atual utilizando apenas dados do instante anterior, tem-se a seguinte igualdade

$$\overline{p}(m_{k-1}) = (\Pi_{k-1})^T p(m_{k-1}) \tag{3.12}$$

Para terminar esta etapa, nota-se que o valor $p(y_k|y_{1:k-1})$ não depende de x_k e tão pouco de m_k, por isso pode ser excluído da análise. De fato, este termo representa apenas um fator constante que multiplica todas as probabilidades dos modos e pode ser cancelado por um

simples processo de normalização das probabilidades levando em conta a seguinte restrição $\sum_{i=1}^{M} \Pr(m_k = i|y_{1:k}) = 1$.

O último termo que resta para estimação da f.d.p. em (3.4) é $p(x_k|m_k, y_{1:k})$. Para o caso mais simples em que o sistema possui apenas um modo, ou seja, $M = 1$, as funções em (3.1)-(3.2) são lineares e os ruídos são brancos, Gaussianos e descorrelacionados, mostra-se que o Filtro de Kalman é o algoritmo ótimo para estimação da f.d.p. *a posteriori* $p(x_k|m_k, y_{1:k})$. Entretanto, para $M > 1$ é fácil ver que o termo $p(x_k|m_k, y_{1:k})$ torna-se uma complexa soma de Gaussianas acabando com a otimalidade do FK. Para contornar esta dificuldade, a estratégia do IMM descrita no Capítulo 2.2 aproxima as densidades de probabilidades por Gaussianas permitindo que a f.d.p $p(x_k|m_k, y_{1:k})$ possa ser determinada por meio de estimadores tradicionais (algoritmos baseados em filtragem de Kalman) sabendo-se que o modo atual m_k, e consequentemente o modelo atual em espaço de estados, são dados.

Neste sentido, um passo importante do filtro IMM é a combinação de estimativas do passo anterior de forma a gerar novas estimativas iniciais para cada FK associado a um modo específico do sistema. Para ilustrar melhor esta etapa, considere as seguinte equações para o passo de mistura de estimativas do IMM:

$$\begin{aligned} \Pr(m_k = i|y_{1:k-1}) &= \sum_{j=1}^{M} \pi_{j,i} \Pr(m_{k-1} = j|y_{1:k-1}), \\ p(x_{k-1}|m_k = i, y_{1:k-1}) &= \frac{\sum_{j=1}^{M} \pi_{j,i} \Pr(m_{k-1} = j|y_{1:k-1}) g_{k-1}(j)}{\Pr(m_k = i|y_{1:k-1})}, \qquad (3.13) \\ g_{k-1}(j) &= p(x_{k-1}|m_{k-1} = j, y_{1:k-1}). \qquad (3.14) \end{aligned}$$

Portanto, conforme discutido anteriormente, a expressão em (3.14) é, em geral, uma soma de M^{k-1} Gaussianas ponderadas e a abordagem de combinação de estimativas do IMM pressupõe que

$$p(x_{k-1}|m_{k-1} = i, y_{1:k-1}) \sim N(\hat{x}_{k-1}^{(i)}, \hat{P}_{k-1}^{(i)}), \qquad (3.15)$$

em que $\hat{x}_{k-1}^{(i)}$ e $\hat{P}_{k-1}^{(i)}$ denotam a estimativa de estado e sua matriz de covariâncias associada fornecidas por um FK seguindo o modo $m_{k-1} = i$.

Tendo em vista o acima descrito e considerando um sistema híbrido com MPT desconhecida conforme mostrado em (3.1)-(3.2), almeja-se mostrar durante o restante deste capítulo a aplicabilidade desta formulação ao problema de SLAM. Desta forma, procura-se descrever mais especificamente o vetor de estados do sistema, as funções f_{m_k} e h_{m_k} bem como o processo de estimação com relação ao problema de localização e mapeamento simultâneos em robótica móvel.

3.3 VETOR DE ESTADOS DO SISTEMA

Em robótica móvel é bastante comum representar a pose do robô da seguinte forma:

$$\boldsymbol{\xi}_k = \begin{pmatrix} x_k \\ y_k \\ \theta_k \end{pmatrix},$$

em que (x_k, y_k) denota a posição 2D no plano cartesiano e θ_k representa um ângulo de orientação do robô.

Seguindo a formulação descrita na introdução deste capítulo, o vetor de estados do sistema é composto pela pose atual do robô, bem como por um conjunto de poses anteriores que representam a trajetória percorrida. Desta forma, o vetor de estados pode ser escrito da seguinte maneira:

$$\mathbf{x}_k = \left(\boldsymbol{\xi}_k^T, \boldsymbol{\xi}_{1,k}^T, \boldsymbol{\xi}_{2,k}^T, ..., \boldsymbol{\xi}_{n_k,k}^T\right)^T \tag{3.16}$$

em que $\boldsymbol{\xi}_{i,k}^T$ representa a pose do robô no i-ésimo ponto da trajetória, sendo que $i = 1, ..., n_k$. $\boldsymbol{\xi}_{i,k}^T$ também pode ser pensado como sendo a pose que o robô tinha quanto ele visitou a i-ésima região do ambiente, denotada aqui por Ω_i.

As regiões do ambiente são definidas a medida que o robô de desloca pela aplicação de regras simples tais como, distância percorrida desde a última região visitada ou então quando o sistema não consegue mais identificar a região atual. Em cada caso, assim que uma nova região é iniciada, o vetor de estados $\mathbf{x}_k$ é aumentado pela pose do robô no instante atual. Portanto, esta pose determina a origem do sistema de coordenadas local de cada região da trajetória. Então, o número de regiões n_k aumenta com o tempo.

3.4 MODELOS NO ESPAÇO DE ESTADOS

Deseja-se obter uma formulação em espaço de estados para o problema de SLAM utilizando a abordagem de sistemas dinâmicos híbridos descrita conforme o modelo dado em (3.1)-(3.2). Seguindo este paradigma, define-se como modo de operação do sistema a variável s_k que corresponde a qual subregião do ambiente o robô está tomando as medidas. Desta forma, deseja-se obter uma estimativa de um conjunto de poses do robô ao longo da trajetória, bem como da subregião na qual o robô faz suas observações sendo que o vetor de estados aumentado do sistema é composto tanto por variáveis contínuas (poses do robô) quanto por uma variável discreta (índice da subregião), ou seja,

$$\mathbf{X}_k = \begin{pmatrix} \mathbf{x}_k \\ s_k \end{pmatrix}$$

em que $s_k \in [1, ..., n_k]$.

Em SLAM, os dados adquiridos através dos sensores são essenciais para a estimação do vetor de estados completo do sistema. Os dados proprioceptivos, por exemplo, podem ser usados para se obter uma estimativa da pose atual do robô bem como estabelecer uma relação entre as poses de cada região da trajetória. Desta forma, à medida que o robô se move pelo ambiente e uma nova região Ω_i é criada, armazena-se o conjunto de dados proprioceptivos adquiridos desde a última região. Representa-se este conjunto de dados por $P_{i,i-1}$. O conjunto de dados proprioceptivos adquiridos no instante atual é denotado por U_k.

Seguindo a formulação em espaço de estados mostrada em (3.1)-(3.2) em um contexto de estimação estocástica, os dados proprioceptivos podem ser usados para calcular uma estimativa predita do estado do sistema de acordo com o seguinte modelo:

$$\mathbf{x}_k = f\left(\mathbf{x}_{k-1}, \mathbf{u}_k, \mathbf{w}_k\right) \tag{3.17}$$

O modelo descrito pela Equação (3.17), tenta capturar a relação que existe entre o estado passado do robô, $\mathbf{x}_{k-1}$ e o estado atual do robô, $\mathbf{x}_k$ dado um conjunto de dados, $\mathbf{u}_k$, provenientes de sensores proprioceptivos. Além disso, $\mathbf{w}_k$ representa um vetor de variáveis aleatórias que descreve os ruídos associados às medidas dos sensores proprioceptivos, bem como as incertezas inerentes ao próprio modelo.

Dependendo da confiabilidade com a qual o deslocamento do robô pode ser modelado e dos sensores disponíveis para capturar as mudanças no estado do robô, o modelo representado em (3.17) pode ser bastante preciso. Idealmente, este modelo é capaz de calcular o deslocamento do robô sem qualquer incerteza associada, ou seja, para cada transição de estado, a pose do robô seria precisamente conhecida. Na prática não é isto que acontece. O fato de que os sinais providos pelos sensores e os sinais aplicados aos atuadores sempre são contaminados por algum tipo de ruído impossibilita que uma estimativa tão precisa do estado do robô seja alcançada.

Geralmente a função $f(\mathbf{x}_{k-1}, \mathbf{u}_k, \mathbf{w}_k)$ assume o seguinte formato:

$$f\left(\mathbf{x}_{k-1}, \mathbf{u}_k, \mathbf{w}_k\right) = \begin{pmatrix} \boldsymbol{\xi}_{k-1} + f_o(\boldsymbol{\xi}_{k-1}, \mathbf{u}_k) \\ \boldsymbol{\xi}_{1,k-1} \\ \vdots \\ \boldsymbol{\xi}_{n_k,k-1} \end{pmatrix} + \mathbf{w}_k \tag{3.18}$$

em que $f_o(\boldsymbol{\xi}_{k-1}, \mathbf{u}_k)$ é comumente conhecida por função de odometria obtida do modelo cinemático do robô e $\mathbf{w}_k \sim N(0, \mathbf{Q}_k)$ representa a incerteza associada a este modelo.

O robô também é equipado com sensores do tipo exteroceptivos. Os dados provenientes destes sensores representam informações sobre o ambiente tomadas a partir da pose atual e das poses passadas do robô que formam a trajetória percorrida até o momento. Estes dados podem ser provenientes dos mais variados tipos de sensores, tais como câmeras de video ou radar a laser. Representa-se por E_i o conjunto de dados exteroceptivos associados à pose

$\boldsymbol{\xi}_{i,k}$. Por outro lado, representa-se por D_k o conjunto de dados exteroceptivos obtidos a partir da pose atual $\boldsymbol{\xi}_k$.

Mais uma vez, em um contexto de estimação estocástica, defini-se um modelo de medição do sistema conforme (3.2) que leva em conta a relação entre as informações extraídas dos conjuntos de dados D_k e E_i, a pose atual do robô e as poses relativas a cada região da trajetória. De fato, este modelo depende explicitamente de qual região da trajetória o robô está tomando as medidas, portanto o modelo de medição tem a seguinte forma:

$$\mathbf{y}_k = h_{s_k}(\mathbf{x}_k) + v_k \tag{3.19}$$

em que $v_k \sim N(0, \mathbf{R}_k)$ representa o ruído de medição. Geralmente, $\mathbf{y}_k$ são parâmetros geométricos de estruturas do ambiente extraídos a partir de D_k.

Uma etapa importante do processo de estimação em SLAM é a verificação de correspondências entre as informações extraídas do conjunto D_k e os dados exteroceptivos armazenados em cada região da trajetória. Este processo consiste em procurar em cada região $\Omega_1, ..., \Omega_{n_k}$ estruturas similares aos dados provenientes de D_k. Geralmente este procedimento de busca envolve um processo de transformação de coordenadas cujos parâmetros dependem explicitamente das poses do robô em cada região da trajetória. Então, se existe uma correspondência entre $\mathbf{y}_k$ e a i-ésima pose $\boldsymbol{\xi}_{i,k}$, a função $h_{s_k}(\mathbf{x}_k)$ pode ser escrita da seguinte maneira:

$$h_{s_k}(\mathbf{x}_k) = h(\boldsymbol{\xi}_k, \boldsymbol{\xi}_{i,k}, E_i) \tag{3.20}$$

De acordo com o modelo proposto nesta seção, os dados obtidos através dos sensores são armazenados à medida que o robô percorre sua trajetória. Resumidamente, estes dados podem ser representados pelos seguintes conjuntos:

- $P_{i,i-1}$: conjunto de dados proprioceptivos obtidos entre as regiões Ω_{i-1} e Ω_i;
- E_i: conjunto de dados extraceptivos relacionados à pose do robô na região Ω_i;
- U_k: dados proprioceptivos obtidos no instante k;
- D_k: dados extraceptivos obtidos no instante k.

A Figura 3.1 mostra um esquemático do sistema enfatizando as poses indicativas de cada região e os conjuntos de dados armazenados.

3.5 O PROCESSO DE ESTIMAÇÃO

Dados os modelos descritos nas seções anteriores, o procedimento de localização e mapeamento consiste em obter estimativas da pose atual do robô ,$\boldsymbol{\xi}_k$, bem como das poses

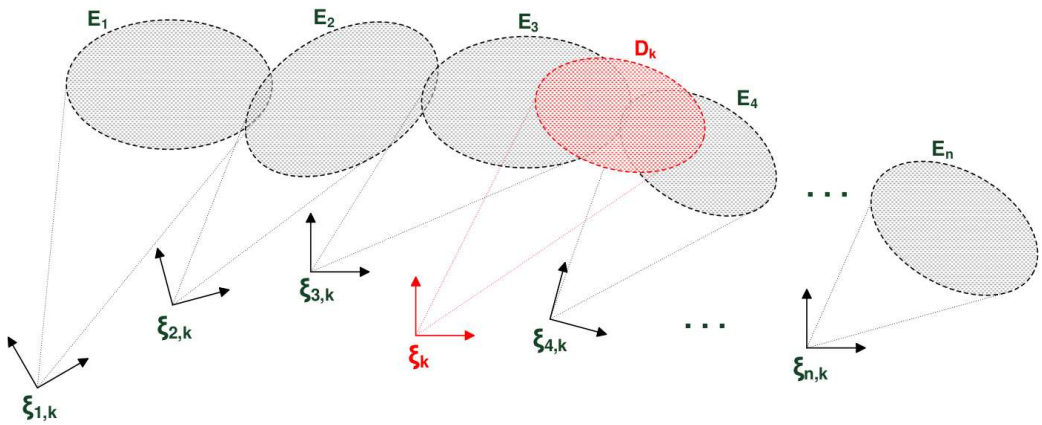


Figura 3.1: Trajetória percorrida pelo robô e conjunto de dados observados em cada pose

indicativas de cada região, $\boldsymbol{\xi}_{i,k}$, utilizando os conjuntos de dados providos pelos sensores. Ademais, em um contexto de estimação em sistemas híbridos, é também necessário estimar o vetor de probabilidades dos modos do sistema que são representados pelos índices de cada subregião do ambiente, isto é,

$$p(s_k) = [\Pr(s_k = 1), \Pr(s_k = 2), ..., \Pr(s_k = n_k)]^T. \tag{3.21}$$

Portanto, o processo de estimação consiste em obter uma estimativa para a f.d.p. *a posteriori* $p(\mathbf{x}_k, s_k|y_{1:k})$. Sabendo que a variável s_k está restrita a um conjunto finito de valores discretos, o Teorema da Probabilidade Total permite que se escreva

$$p(\mathbf{x}_k, s_k|y_{1:k}) = \sum_{i=1}^{n_k} p(\mathbf{x}_k|s_k = i, y_{1:k}) \Pr(s_k = i|y_{1:k}), \tag{3.22}$$

ou seja, estimar (3.22) consiste em combinar a estimativa de vários filtros, cada um assumindo que um modo específico do sistema esteja em vigor no instante atual. Desta forma, a estimativa final é calculada por uma ponderação da estimativa de cada filtro por meio do vetor $p(s_k)$ de acordo com a probabilidade de suas saídas estarem corretas.

Reforçando aquilo que foi exposto no Capítulo 2.2 e na Seção 3.2, o algoritmo IMM mostra-se como uma boa opção para estimação de estados em sistemas a múltiplos modelos [49]. Por este motivo, este algoritmo foi escolhido como base para realizar a estimação de estados descrito por (3.17) e (3.19). Além disso, optou-se por incorporar às equações do IMM o algoritmo de estimação da MPT (matriz que rege as transições entre os modos do sistema) mostrado na Seção 2.2.2, visto que a hipótese de que essa matriz seja previamente conhecida raramente é verdade podendo levar a uma significativa perda de desempenho do filtro [111]. Portanto, além de prover as estimativas $\hat{\mathbf{x}}_k$ e $\hat{p}(s_k)$, o algoritmo é capaz também de estimar recursivamente $\hat{\Pi}_k$ baseado em informações ruidosas extraídas das saídas dos sensores. A seguir, descreve-se com mais detalhes cada passo envolvido no processo de estimação enfatizando as etapas de predição, observação e correção que são típicas em um contexto de estimação estocástica.

3.5.1 Predição das probabilidades dos modos

De acordo com o que foi exposto anteriormente, o passo de predição das probabilidades dos modos segue o modelo de evolução dado por uma Cadeia de Markov na qual a transição entre os modos é determinada por uma matriz de probabilidades denominada MPT e representada por Π_k, tal que

$$\Pi_k = \{\pi_{i,j}(k)\}, \;\; \pi_{i,j}(k) = \Pr\{s_k = j | s_{k-1} = i\}, \;\; i,j \in [1,2,...,n_k], \forall k \in \mathbb{N} \tag{3.23}$$

Então, definindo

$$\hat{p}^{(i)}(s_k) = \Pr(s_k = i | y_{1:k}) \tag{3.24}$$

e assumindo as seguintes condições iniciais

$$\hat{p}^{(i)}(s_0), \; \hat{\Pi}_0, \; i \in \{1,2,...,n_k\}, \tag{3.25}$$

calcula-se o vetor de probabilidades predito $\overline{p}^{(i)}(s_k)$ através da seguinte equação

$$\overline{p}^{(i)}(s_k) = \sum_{j=1}^{n_k} \hat{\pi}_{j,i}(k-1)\hat{p}^{(j)}(s_{k-1}). \tag{3.26}$$

3.5.2 Mistura das estimativas

O passo de mistura das estimativas é uma etapa típica do filtro IMM e que o distingue da maioria dos outros algoritmos de filtragem de sistemas a múltiplos modelos. Nesta etapa, as estimativas do passo anterior são combinadas de forma a gerar novas estimativas iniciais para o seu banco de filtros de Kalman a cada instante de tempo. Por causa deste passo de mistura de estimativas, o IMM apresenta requisitos computacionais que são lineares com relação ao número de modos do sistema. Uma ótima descrição da vantagem deste passo de mistura do filtro IMM é dada em [99] em que o autor enfatiza matematicamente a importancia desta etapa no desempenho do algoritmo.

Portanto, sejam $\hat{x}_k^{(i)}$ e $\hat{P}_k^{(i)}$, tal que $i \in \{1,2,...,n_k\}$, o vetor de estados e sua matriz de covariâncias associadas correspondendo ao filtro ajustado à região $s_k = i$ no *k-ésimo* instante amostral. As equações para o passo de mistura são dadas por

$$\underline{x}_{k-1}^{(i)} = \frac{\sum_{j=1}^{n_k} \hat{\pi}_{j,i}(k-1)\hat{p}^{(j)}(s_{k-1})\hat{x}_{k-1}^{(j)}}{\overline{p}^{(i)}(s_k)}, \tag{3.27}$$

$$\underline{P}_{k-1}^{(i)} = \frac{\sum_{j=1}^{n_k} \hat{\pi}_{j,i}(k-1)\hat{p}^{(j)}(s_{k-1}) \left[\hat{P}_{k-1}^{(j)} + \delta(i,j)\right]}{\overline{p}^{(i)}(s_k)}, \tag{3.28}$$

$$\delta(i,j) = \left(\hat{x}_{k-1}^{(j)} - \underline{x}_{k-1}^{(i)}\right)\left(\hat{x}_{k-1}^{(j)} - \underline{x}_{k-1}^{(i)}\right)^T. \tag{3.29}$$

3.5.3 Predição das estimativas

Já foi mencionado nesta dissertação que o algoritmo de filtragem IMM aplicado ao problema de SLAM em robótica móvel faz uso de um banco de estimadores relacionados a cada subregião da trajetória realizada pelo robô (modos do sistema). Cada estimador em particular utiliza uma técnica bastante conhecida na literatura denominada Filtro de Kalman Estendido (FKE). Portanto, a etapa de predição das estimativas de cada filtro segue basicamente as equações do passo de predição do FKE.

A etapa de predição do FKE utiliza o modelo descrito na Equação (3.18) para gerar uma estimativa predita para cada filtro correspondendo à subregião $s_k = i$ que é dada por $\overline{\mathbf{x}}_k^{(i)}$, considerando os dados disponíveis até o instante $k-1$, isto é,

$$\overline{\mathbf{x}}_k^{(i)} = f(\underline{\mathbf{x}}_{k-1}^{(i)}, \mathbf{u}_k) \tag{3.30}$$

em que $\mathbf{u}_k$ é o conjunto de dados proprioceptivos extraídos de U_k.

Nota-se claramente neste caso, que a função de processo em (3.30) não depende de qual subregião o robô está tomando as medidas de forma que os filtros de cada modo do sistema compartilham o mesmo modelo de predição.

A matriz de covariâncias de cada filtro também deve ser propagada através deste modelo como parte da etapa de predição. O FKE utiliza um processo de linearização em torno da última estimativa do vetor de estados $\underline{\mathbf{x}}_{k-1}^{(i)}$ usando a matriz Jacobiana $\nabla_{\mathbf{x}} f$ da função f avaliada em $\underline{\mathbf{x}}_{k-1}^{(i)}$. Portanto a matriz de covariâncias deve ser propagada da seguinte maneira:

$$\overline{P}_k^{(i)} = (\nabla_{\mathbf{x}} f)\underline{P}_{k-1}^{(i)}(\nabla_{\mathbf{x}} f)^T + \mathbf{Q}_k \tag{3.31}$$

em que a matriz Jacobiana $\nabla_{\mathbf{x}} f$ é dada por

$$\nabla_{\mathbf{x}} f = \frac{\partial f(\mathbf{x}_{k-1}^{(i)}, \mathbf{u}_k)}{\partial \mathbf{x}_{k-1}^{(i)}} \tag{3.32}$$

Para considerar também os ruídos associados aos dados extraídos dos sensores proprioceptivos deve-se calcular também a Jacobiana $\nabla_{\mathbf{u}} f$ da função f com relação a $\mathbf{u}_k$, resultando na seguinte matriz de covariâncias:

$$\overline{P}_k^{(i)} = (\nabla_{\mathbf{x}} f)\underline{P}_{k-1}^{(i)}(\nabla_{\mathbf{x}} f)^T + (\nabla_{\mathbf{u}} f)P_{\mathbf{u}}(\nabla_{\mathbf{u}} f)^T + \mathbf{Q}_k, \tag{3.33}$$

em que a matriz $P_{\mathbf{u}}$ representa as variâncias associadas às medidas dos sensores proprioceptivos e a matriz Jacobiana $\nabla_{\mathbf{u}} f$ é dada por

$$\nabla_{\mathbf{u}} f = \frac{\partial f(\mathbf{x}_{k-1}^{(i)}, \mathbf{u}_k)}{\partial \mathbf{u}_k}. \tag{3.34}$$

3.5.4 Correção das estimativas

De acordo com o algoritmo do FKE, as estimativas preditas de cada filtro, $\overline{\mathbf{x}}_k^{(i)}$, podem ser corrigidas com base em um conjunto de medições ruidosas y_k observadas no instante atual. Em SLAM, o desempenho desta etapa está intimamente relacionado com um processo de casamento de dados pois as observações atuais são úteis para calcular uma estimativa da pose do robô somente se elas são realizadas de diferentes posições. Portanto, quando as observações são recebidas por meio dos sensores exteroceptivos, elas devem ser associadas àquelas previamente armazenadas. Este processo consiste em procurar entre todos os conjuntos de dados E_i por estruturas similares aos dados provenientes de D_k.

Este processo de casamento de dados é efetuado, primeiramente, calculando-se um conjunto de observações preditas $\overline{y}_k$ a partir das funções de medição h_{s_k} associadas a cada subregião da trajetória conforme descrito na Equação (3.20), ou seja

$$\overline{y}_k = h_{s_k}(\overline{\mathbf{x}}_k). \tag{3.35}$$

Uma boa medida de proximidade pode ser calculada a partir do termo de inovação ν_k dado pela seguinte expressão:

$$\nu_k = y_k - \overline{y}_k. \tag{3.36}$$

Contudo, a incerteza associada a esta medida também deve ser determinada afim de que a similaridade entre a observação predita $\overline{y}_k$ e a saída do sistema y_k seja verificada de forma consistente.

Os métodos mais comuns utilizados para resolver o problema de associação de dados baseiam-se em técnicas conhecidas como vizinho mais próximo [96, 119]. Uma medida estatística bastante comum na determinação de correspondência entre duas estimativas é o teste de Mahalanobis. Este teste fornece uma distância estatística entre duas estimativas de modo que o valor desta distância pode ser comparado com um limiar para validar a correspondência entre elas, isto é

$$d = \nu_k^T S_k^{-1} \nu_k < d_{min}, \tag{3.37}$$

em que S_k é a incerteza associada ao termo de inovação ν_k. Considerando que as estimativas podem ser modeladas como variáveis aleatórias Gaussianas, a distância de Mahalanobis segue uma distribuição de probabilidade $\mathcal{X}^2$ e pode ser usado então para aceitar ou rejeitar uma certa correspondência com um grau de confiança.

Conforme descrito na Seção 3.2 a determinação da verossimilhança de y_k é uma etapa básica na estimação da f.d.p. *a posteriori* $p(\mathbf{x}_k, s_k|y_{1:k})$ e o cálculo da expressão mostrada em (3.10) pode ser usado na verificação do teste de Mahalanobis formulado em (3.37). Isto mostra que o processo de casamento de dados em SLAM já está, de certa forma, incorporado às equações do algoritmo de filtragem IMM.

Uma vez concluído o processo de associação de dados, a estimativa predita de cada filtro é corrigida de acordo com a correspondência verificada pelo teste de Mahalanobis.

Então, se existe uma correspondência entre y_k e algum dado proveniente de E_i, tal que $i = [1, 2, ..., n_k]$, a função de medição h_{s_k} pode ser escrita conforme em (3.20) e a estimativa predita $\overline{\mathbf{x}}_k^{(i)}$ do filtro correspondente à subregião $s_k = i$ pode ser corrigida utilizando as equações do FKE.

No algoritmo do FKE, o procedimento usual de correção consiste em agrupar em um único vetor de saída todos os dados correspondidos e processar todas as informações ao mesmo tempo. Contudo, este método pode mostrar-se computacionalmente ineficiente caso a dimensão do vetor de saída fique demasiadamente grande [120]. Uma alternativa a este método foi proposta em [121] em que os autores demonstram que os dados dos sensores podem ser processados sequencialmente caso suas medidas sejam descorrelacionadas. Portanto, considerando esta a situação, o processo de correção das estimativas preditas de cada filtro pode ser feito de forma progressiva à medida em que o processo de associação de dados determina as correspondências entre o conjunto de observações y_k e os dados previamente armazenados E_i. Este procedimento está melhor ilustrado na Figura 3.2.

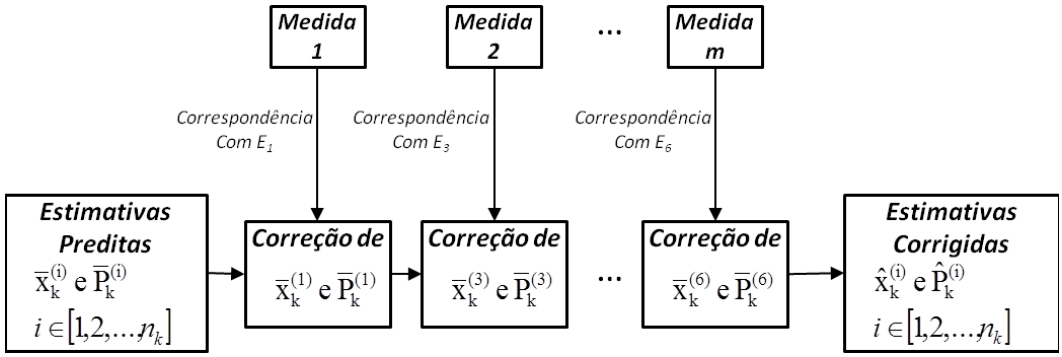


Figura 3.2: Processo de associção de dados e correção sequencial das estimativas.

Ainda com relação à Figura 3.2 nota-se que quando não há correspondência com nenhum dos dados dos conjuntos E_i, ou seja, quando uma nova medida é tomada, ela não é incorporada à etapa de correção e não contribui para o processo de atualização das estimativas.

Para concluir esta etapa, mostra-se uma breve explicação das equações utilizadas no processo de correção das estimativas. Esta etapa é realizada por uma matriz de correção comumente referida na literatura por matriz de ganho de Kalman G_k. Esta matriz de ganho realiza uma soma ponderada entre as estimativas preditas e as observações y_k e é calculada a partir da covariância do termo de inovação S_k e da matriz de covariâncias da estimativa predita $\overline{P}_k$. Desta forma, o processo de correção para as estimativas preditas do filtro correspondente à subregião $s_k = i$ é dado pelas seguintes equações

$$\hat{\mathbf{x}}_k^{(i)} = \overline{\mathbf{x}}_k^{(i)} + G_k \nu_k, \tag{3.38}$$

$$\hat{\mathbf{P}}_k^{(i)} = \overline{\mathbf{P}}_k^{(i)} - G_k S_k G_k^T, \tag{3.39}$$

em que a matriz de ganho G_k é dada por

$$G_k = \overline{\mathbf{P}}_k^{(i)} \left(\frac{\partial h_{s_k}(\mathbf{x}_k)}{\partial \mathbf{x}_k} \right)^T S_k^{-1}. \tag{3.40}$$

3.5.5 Correção das probabilidades dos modos

Terminado o processo de correção das estimativas, uma etapa importante do algoritmo IMM é a atualização do vetor de probabilidade dos modos de operação do sistema a partir de y_k. De acordo com a formulação do problema de SLAM proposta nesta dissertação, os modos do sistema são representados por cada uma das subregiões da trajetória realizada pelo robô de forma que o vetor $p(s_k)$ denota as probabilidades dos índices associados à cada subregião. Portanto, por meio do vetor de probabilidades corrigido $\hat{p}(s_k)$ é possível determinar probabilisticamente de qual subregião o robô está tomando as medidas no instante atual.

O conhecimento acerca do vetor de probabilidades $p(s_k)$ a cada passo de estimação do filtro levou à proposta de uma modificação no algoritmo convencional do estimador IMM que resultou em significativa melhora no desempenho para a solução do problema de SLAM. Os fundamentos desta pequena alteração serão melhor explicados na Seção 3.6.

Assim, com base no algoritmo do filtro IMM mostrado no Capítulo 2.2 as equações para a correção das probabilidades dos modos podem ser escritas da seguinte forma

$$\hat{p}^{(i)}(s_k) = \frac{p(y_k|s_k = i, \hat{\Pi}_{k-1}, y_{1:k-1})\overline{p}^{(i)}(s_k)}{c_i}, \tag{3.41}$$

$$\gamma_p = \sum_{j=1}^{n_k} \hat{p}^{(j)}(s_k), \tag{3.42}$$

$$\hat{p}(s_k) = [\hat{p}^{(1)}(s_k), ..., \hat{p}^{(n_k)}(s_k)]^T \left(\frac{1}{\gamma_p} \right), \tag{3.43}$$

em que o termo $p(y_k|s_k = i, \hat{\Pi}_{k-1}, y_{1:k-1})$ é a verossimilhança de y_k que já foi calculada durante o processo de associação de dados e c_i é uma constante que não precisa ser determinada.

3.5.6 Geração das saídas

Um vez calculadas as estimativas $\hat{\mathbf{x}}_k^{(i)}$ e $\hat{P}_k^{(i)}$ para cada filtro correspondendo à subregião $s_k = i$, tal que $i \in [1, 2, ..., n_k]$, bem como o vetor de probabilidades $\hat{p}(s_k)$, a estimativa final dada pelo algoritmo IMM é determinada por uma soma ponderada de acordo com as

seguintes equações

$$\hat{\mathbf{x}}_k = \sum_{i=1}^{n_k} \hat{p}^{(i)}(s_k), \tag{3.44}$$

$$\hat{P}_k = \sum_{i=1}^{n_k} \left[\hat{P}_k^{(i)} + \left(\hat{\mathbf{x}}_k^{(i)} - \hat{\mathbf{x}}_k \right) \left(\hat{\mathbf{x}}_k^{(i)} - \hat{\mathbf{x}}_k \right)^T \right]. \tag{3.45}$$

3.5.7 Atualização da MPT

Conforme já foi mencionado anteriormente, as transições entre os modos do sistema são governadas por uma Cadeia de Markov, cuja representação matemática é dada pela Matriz de Probabilidades de Transição (MPT). Esta matriz também é estimada $online$ de acordo com o algoritmo Quasi-Bayesiano mostrado na Seção 2.2.2.

3.5.8 Incorporação de novos elementos ao vetor de estados

De acordo com o modelo descrito neste trabalho o vetor de estados contínuos de cada filtro $\mathbf{x}_k^{(i)}$ correspondente à subregião $s_k = i$ é composto pela pose atual do robô, bem como as poses quando da aquisição da dados anteriores. Cada pose é associada a uma região da trajetória e portanto, à medida que o robô se desloca novas regiões são definidas e a pose do robô neste instante deve ser incorporada ao vetor de estados de cada filtro.

A matriz de covariâncias $P_k^{(i)}$ desta nova pose deve também ser devidamente inicializada, uma vez que ela é correlacionada com a estimativa da pose atual do robô e com as estimativas das outras poses que também fazem parte do vetor de estado do sistema. Ignorar o efeito desta correlação poderia levar a uma inconsistência no processo de estimação. Portanto, quando uma nova região é criada seguem-se as seguintes modificações para a matriz de covariâncias $P_k^{(i)}$. O procedimento descrito a seguir é realizado para cada filtro seguindo uma determinada subregião da trajetória.

Primeiramente, a matriz de covariâncias é aumentada com a incerteza associada à pose atual do robô. Depois, preenche-se o restante da matriz com as correlações cruzadas entre o restante dos elementos do vetor de estados. Assume-se que a matriz de covariâncias inicial é dada por P_k^- e representada pela seguinte equação:

$$P_k^- = \begin{pmatrix} P_k^{rr} & P_k^{r1} & \cdots & P_k^{rn_k} \\ P_k^{r1} & P_k^{11} & \cdots & P_k^{1n_k} \\ \vdots & \vdots & \ddots & \vdots \\ P_k^{rn_k} & P_k^{1n_k} & \cdots & P_k^{n_k n_k} \end{pmatrix} \tag{3.46}$$

em que P_k^{rr} representa as covariâncias relacionadas com a pose atual do robô, P_k^{jj}, tal que $j \in [1, 2, ..., n_k]$, representa as covariâncias associadas às poses de cada região da trajetória, P_k^{rj}, tal que $j \in [1, 2, ..., n_k]$, representa as covariâncias cruzadas entre a pose atual do robô

e as poses anteriores de cada região da trajetória e por último P_k^{jk}, tal que $j, k \in [1, 2, ..., n_k]$, representa as covariâncias cruzadas entre as estimativas das poses indicativas das regiões do ambiente.

Então, aumenta-se a matriz de covariâncias P_k^- com a incerteza associada à pose atual do robô, P_k^{rr}

$$P_k^+ = \begin{pmatrix} P_k^{rr} & P_k^{r1} & \cdots & P_k^{rn_k} & 0 \\ P_k^{r1} & P_k^{11} & \cdots & P_k^{1n_k} & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ P_k^{rn_k} & P_k^{1n_k} & \cdots & P_k^{n_k n_k} & 0 \\ 0 & 0 & \cdots & 0 & P_k^{rr} \end{pmatrix} \tag{3.47}$$

A matriz de covariâncias final é calculada então da seguinte forma:

$$P_k^+ = \begin{pmatrix} P_k^{rr} & P_k^{r1} & \cdots & P_k^{rn_k} & P_k^{rr} \\ P_k^{r1} & P_k^{11} & \cdots & P_k^{1n_k} & P_k^{r1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ P_k^{rn_k} & P_k^{1n_k} & \cdots & P_k^{n_k n_k} & P_k^{rn_k} \\ P_k^{rr} & P_k^{r1} & \cdots & P_k^{rn_k} & P_k^{rr} \end{pmatrix} \tag{3.48}$$

3.6 UMA NOVA VERSÃO PARA O FILTRO IMM APLICADO AO PROBLEMA DE SLAM

Na formulação matemática apresentada nas Seções 3.3 e 3.4 o problema de SLAM é abordado de forma desacoplada em que o que se estima é a pose atual do robô, bem como um conjunto de poses ao longo da trajetória. Então, ao invés de um mapa global, são mantidos apenas um conjunto de submapas locais associados às poses indicativas de cada subregião da trajetória. O procedimento de estimação proposto na Seção 3.5 leva em conta esta formulação sob um ponto de vista de sistemas híbridos. Segundo esta perspectiva, o processo de filtragem provê estimativas de um conjunto de poses do robô ao longo da trajetória, bem como uma estimativa de qual subregião o robô está tomando as medidas no instante atual.

À primeira vista, a escolha deste tipo de formulação pode levar a alguns inconvenientes. Primeiramente, nota-se que à medida que o robô navega pelo ambiente novas subregiões são criadas e os dados dos sensores (proprioceptivos e exteroceptivos) são armazenados nesse instante. Portanto, o número total de subregiões no instante atual, n_k, é variável no tempo. Isto implica em um aumento significativo no tempo computacional do algoritmo principalmente durante a etapa de mistura de estimativas e durante o processo de associação de dados, ainda mais quando o robô é levado a percorrer grandes ambientes por um longo período de operação.

A etapa de mistura de estimativas realiza uma combinação do vetor de estados e matrizes de covariâncias de cada filtro associado a uma determinada subregião do ambiente e gera

novas condições iniciais para o passo atual de filtragem. De fato, conforme mostrado nas equações da Seção 3.5.2, o tempo computacional relativo à esta etapa é fortemente dependente do número de subregiões no instante atual. À medida que n_k cresce esta etapa pode se tornar impraticável se considerarmos que os recursos computacionais não são ilimitados.

Outro processo crítico em SLAM no sentido de tempo computacional é o procedimento de verificação de correspondências entre os dados armazenados em cada subregião da trajetória e as observações feitas no instante atual. Este procedimento envolve muitas verificações e operações de inversão matricial, por isso à medida que o número de dados armazenados aumenta, esta etapa requer um tempo de computação cada vez maior.

Uma alternativa a este problema seria explorar a informação contida no vetor de probabilidades dos modos para encontrar alguma vantagem no sentido de tempo computacional. Uma solução bastante simples, mas que mostrou-se bastante eficaz foi eliminar do processo de estimação todas as subregiões que possuíssem baixa probabilidade no vetor $p(s_k)$. Então, a idéia seria introduzir uma certa heurística no processo de filtragem.

Desta forma, sejam E_i e E_j os conjuntos de dados exteroceptivos associados às poses $\boldsymbol{\xi}_{i,k}$ e $\boldsymbol{\xi}_{j,k}$, respectivamente. Considere também que estas regiões podem ser representadas pelos seus respectivos centróides de área C_i e C_j. As subregiões definidas por estes conjuntos de dados são vizinhas, se somente se, a distância euclidiana entre os seus centróides não ultrapassar um certo valor limite. Este limiar pode ser definido com base no alcance máximo dos sensores exteroceptivos.

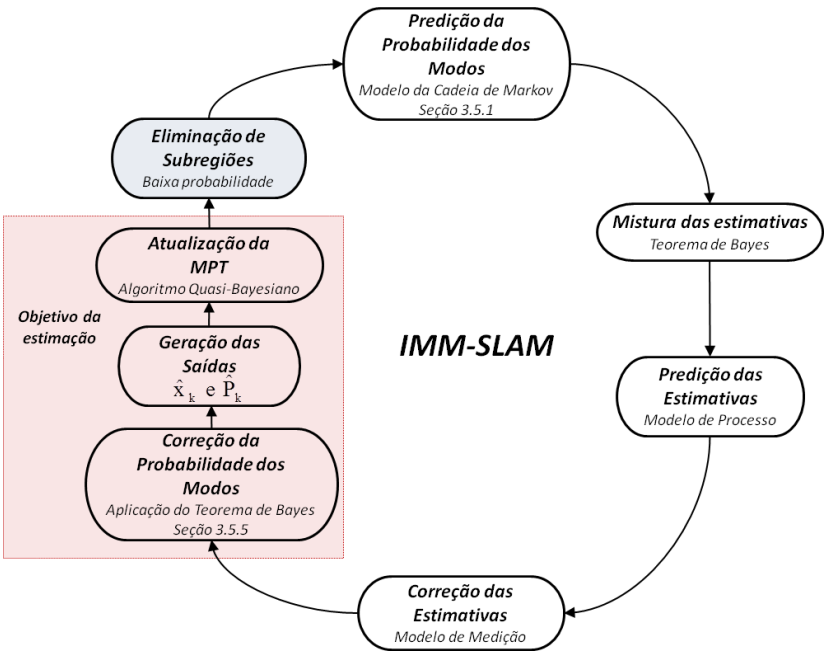


Figura 3.3: Diagrama esquemático das principais etapas envolvidas no algoritmo IMM aplicado ao problema de SLAM.

Então, definido este conceito de subregiões vizinhas, para cada passo de estimação usa-

se o vetor de probabilidades corrigido $\hat{p}(s_k)$ para determinar a região mais provável da qual o robô está tomando as medidas naquele instante e elimina-se do próximo passo de filtragem todas as subregiões que não são vizinhas. Portanto, as etapas do processo de estimação são realizadas apenas com um subconjunto de todos os dados armazenados até o momento. Através deste procedimento, verificou-se um significativo ganho no desempenho do filtro principalmente nas etapas de mistura das estimativas e no processo de associação de dados.

A Figura 3.3 ilustra os passos do algoritmo IMM mostrados na Seção 3.5 e além disso indica em que momento é realizada a eliminação de subregiões. Logo a seguir, o Algoritmo 2 resume todas as etapas envolvidas no processo de estimação desenvolvido nesta dissertação. No Capítulo 4 é apresentada uma simulação com o objetivo de validar o resultado teórico apresentado neste capítulo.

Algoritmo 2: $(\hat{\mathbf{x}}_k, \mathbf{P}_k)$ = **imm_slam** (`num_passos`)

$$\begin{aligned}
\mathbf{y}_0, \mathbf{R}_0 &= \textit{receber_medidas} \\
\hat{\mathbf{x}}_0, \mathbf{P}_0, \hat{p}(s_0), \hat{\mathbf{\Pi}}_0 &= \textit{init_subregiões}(\mathbf{y}_0, \mathbf{R}_0)
\end{aligned}$$

for $k = 1$ to `num_passos` **do**

$$\begin{aligned}
\mathcal{S}_k &= \textit{elimina_subregiões}\,(\hat{p}(s_{k-1})) \\
\overline{p}^{(i)}(s_k) &= \textit{pred_modos}\left(\hat{\pi}_{k-1}^{(j,i)}, \hat{p}^{(j)}(s_{k-1})\right) && (3.26) \\
\underline{\mathbf{x}}_{k-1}^{(i)}, \underline{\mathbf{P}}_{k-1}^{(i)} &= \textit{mistura}\left(\hat{\pi}_{k-1}^{(j,i)}, \hat{p}^{(j)}(s_{k-1}), \overline{p}^{(i)}(s_k), \hat{\mathbf{x}}_{k-1}^{(j)}, \hat{\mathbf{P}}_{k-1}^{(j)}\right) && (3.27) - (3.29) \\
\hat{\mathbf{x}}_{k|k-1}^{(i)}, \mathbf{P}_{k|k-1}^{(i)} &= \textit{predição}\left(\underline{\mathbf{x}}_{k-1}^{(i)}, \underline{\mathbf{P}}_{k-1}^{(i)}, \mathbf{u}_k, \mathbf{Q}_k\right) && (3.30) - (3.31) \\
\mathbf{y}_k, \mathbf{R}_k &= \textit{receber_medidas} \\
\mathcal{H}_k &= \textit{associação_de_dados}(\hat{\mathbf{x}}_{k|k-1}^{(i)}, \mathbf{P}_{k|k-1}^{(i)}, \mathbf{y}_k, \mathbf{R}_k, \mathcal{S}_k) && \text{Figura } 3.2 \\
\hat{\mathbf{x}}_k^{(i)}, \mathbf{P}_k^{(i)} &= \textit{correção}(\hat{\mathbf{x}}_{k|k-1}^{(i)}, \mathbf{P}_{k|k-1}^{(i)}, \mathbf{y}_k, \mathbf{R}_k, \mathcal{H}_k) && (3.38) - (3.40) \\
\hat{p}(s_k) &= \textit{correção_modos}\left(\hat{p}(s_{k-1}), \hat{\mathbf{\Pi}}_{k-1}, \mathbf{y}_k\right) && (3.41) - (3.43) \\
\hat{\mathbf{x}}_k, \mathbf{P}_k &= \textit{gerar_saída}\left(\hat{\mathbf{x}}_k^{(i)}, \mathbf{P}_k^{(i)}, \hat{p}_{s_k}\right) && (3.44) - (3.45) \\
\hat{\mathbf{x}}_k, \mathbf{P}_k &= \textit{aumentar_subregiões}(\hat{\mathbf{x}}_k, \mathbf{P}_k, \mathbf{y}_k, \mathbf{R}_k, \mathcal{H}_k)
\end{aligned}$$

end for `return` ($\hat{\mathbf{x}}_k$,$\mathbf{P}_k$)

A função *receber_medidas* diz respeito a aquisição de dados dos sensores exteroceptivos. Estes dados correspondem a um conjunto de estruturas do ambiente observadas localmente. A cada instante k, obtem-se dos sensores exteroceptivos um conjunto de medidas $\mathbf{y}_{k,i}(i = 1, ..., m)$. Os dados dos sensores exteroceptivos, juntamente com a pose do robô no instante de sua aquisição, são armazenados e formam as subregiões ao longo da trajetória realizada pelo robô. A função *elimina_subregiões* é responsável por eliminar as todas as subregiões

que possuem baixa probabilidade no vetor $p(s_k)$. Assim, várias etapas do processo de estimação, principalmente o processo de associação de dados, são feitos apenas com um subconjunto de todos os dados armazenados até o momento atual. A função *pred_modos* implementa o passo de predição das probabilidades dos modos seguindo um modelo dado por uma cadeia de Markov. O vetor de probabilidade dos modos predito é determinado pela MPT do passo anterior $\hat{\mathbf{\Pi}}_{k-1}$ e também pelo vetor de probabilidade dos modos no instante anterior $\hat{\mathbf{p}}(s_{k-1})$. Os índices $i, j \in [1, 2, ..., n_k]$ em que n_k denota a quantidade de modos existente no instante k.

4 RESULTADOS NUMÉRICOS

4.1 INTRODUÇÃO

O método lançado neste trabalho para solução do problema de SLAM foi avaliado por meio de uma simulação computacional desenvolvida em *Matlab*^{TM} com o objetivo de verificar o desempenho e algumas propriedades da proposta híbrida ao problema de SLAM. Os resultados obtidos, juntamente com as simulações computacionais, comprovam a viabilidade e a aplicabilidade da formulação do problema de localização e mapeamento simultâneos em robótica móvel em um contexto de sistemas dinâmicos híbridos.

4.2 MODELOS ENVOLVIDOS DURANTE O PROCESSO DE SLAM

Assume-se durante a simulação que o robô encontra-se em um ambiente 2D de modo que o seu deslocamento esteja limitado a um plano. Desta forma, a pose do robô é definida por suas coordenadas cartesianas (x, y) e por sua orientação θ com relação a um referencial absoluto. Os *landmarks* espalhados pelo ambiente são modelados por pontos e representados por $\mathbf{p}_i = (x_i, y_i)^T$, tal que $i = 1...N$. O modelo de locomoção considerado nos experimentos emprega uma configuração do tipo triciclo conforme mostrado na Figura 4.1.

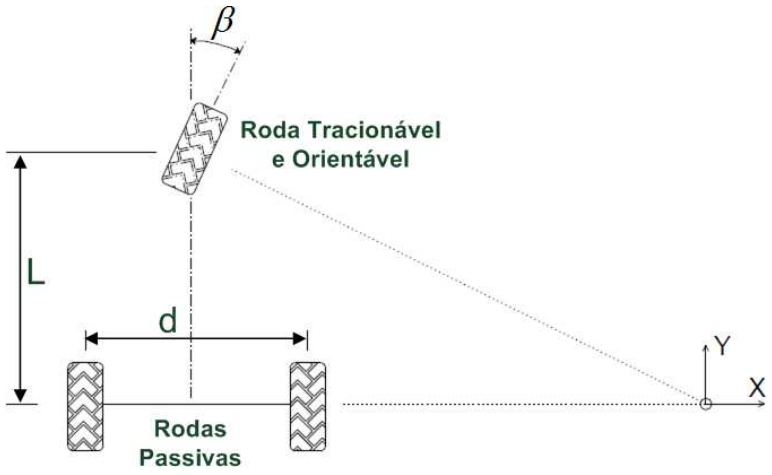


Figura 4.1: Configuração do tipo triciclo que utiliza uma única roda frontal tracionável e orientável e duas rodas traseiras passivas ligadas por um mesmo eixo.

A configuração mostrada na Figura 4.1 é uma plataforma composta por três rodas, sendo que a roda frontal é tracionável e orientável de forma independente enquanto que as duas rodas traseiras são fixas (não orientáveis) e livres para girar. As relações existentes entre as variáveis de configuração $\mathbf{q} = (\varphi, \beta)^T$, em que φ e β são respectivamente o deslocamento angular e o ângulo de direção da roda frontal, no espaço articular do robô e a variação de

sua pose $\xi = (x, y, \theta)^T$ dentro do espaço cartesiano de trabalho são obtidas por meio de um modelo cinemático. Para modelar as equações cinemáticas que descrevem o robô utilizado nesta simulação considera-se que há somente rolagem das rodas, ou seja, desconsidera-se que haja deslizamento na região de contato entre a roda e o solo. Os parâmetros geométricos envolvidos no modelo cinemático são mostrados na Figura 4.2.

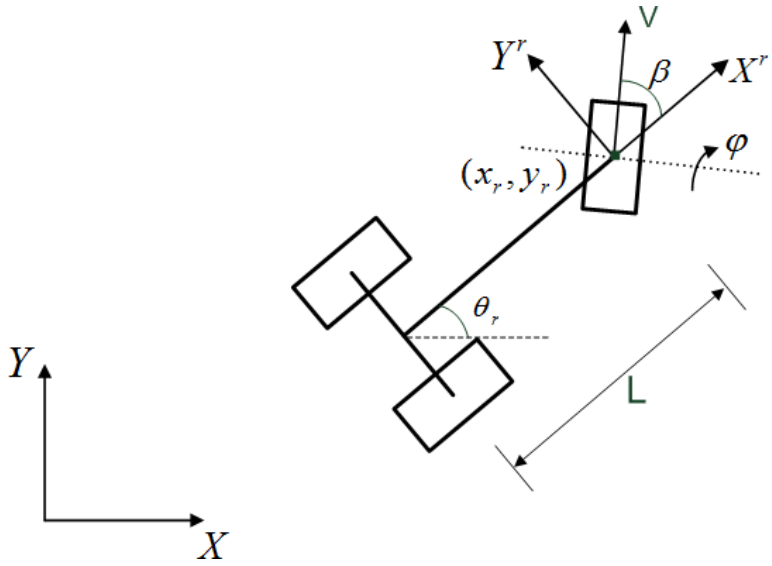


Figura 4.2: Esquema para obtenção do modelo cinemático enfatizando as variáveis de configuração φ e β no espaço articular do robô, bem como sua pose $\xi = (x, y, \theta)^T$ com relação a um referencial absoluto.

Para o levantamento do modelo cinemático é comum partir de relações no sistema de coordenadas do robô X^RxY^R e levar essas relações para o sistema de coordenadas absoluto XxY. Portanto, se $\dot{\xi}^R$ contem as velocidades tangencial e de rotação no espaço X^RxY^R, pode-se escrever:

$$v = R(\theta)v^R \tag{4.1}$$

$$\omega = \omega^R \tag{4.2}$$

desta forma tem-se

$$\dot{\xi} = \begin{pmatrix} R(\theta) & 0 \\ 0 & 1 \end{pmatrix} \dot{\xi}^R \tag{4.3}$$

em que a matriz $R(\theta)$ é dada por:

$$R(\theta) = \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix} \tag{4.4}$$

Uma característica deste tipo de configuração cinemática é que a roda frontal não permanece fixa, portanto o vetor v^R depende do ângulo de orientação β, ou seja,

$$v^R = \begin{pmatrix} V\cos(\beta) \\ V\sin(\beta) \end{pmatrix} = \begin{pmatrix} \dot{\varphi}.r.\cos(\beta) \\ \dot{\varphi}.r.\sin(\beta) \end{pmatrix} \tag{4.5}$$

em que $\dot{\varphi}$ é a velocidade de rotação da roda frontal e $V = \dot{\varphi}.r$ é a velocidade de deslocamento linear provocado pela roda frontal, sendo r o raio da roda.

A velocidade de rotação ω^R pode ser calculada a partir do centro instantâneo de rotação (CIR) que está relacionado ao raio de curvatura realizado pelo robô conforme mostra a Figura 4.3.

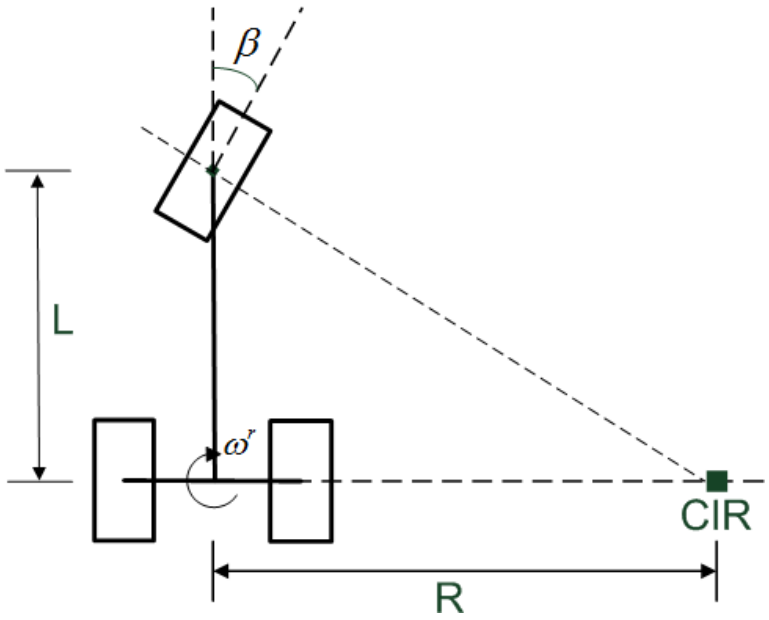


Figura 4.3: Ilustração do lugar geométrico do centro instântaneo de rotação (CIR).

No caso do robô mostrado na Figura 4.1, o CIR encontra-se sempre a uma distância R ao longo de uma reta perpendicular passando pelo eixo das rodas traseiras [122]. Pela Figura 4.3 e utilizando algumas relações trigonométricas, o raio de curvatura R pode ser calculado da seguinte maneira:

$$R = L \tan\left(\frac{\pi}{2} - \beta\right) \tag{4.6}$$

Além disso, a velocidade angular ω^R pode ser calculada da seguinte forma [122]:

$$\omega^R = \frac{V}{\sqrt{L^2 + R^2}} = \frac{\dot{\varphi}.r}{L} \sin\beta \tag{4.7}$$

Finalmente no sistema XxY tem-se:

$$\dot{\xi} = \begin{pmatrix} R(\theta) & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \dot{\varphi}.r.\cos(\beta) \\ \dot{\varphi}.r.\sin(\beta) \\ \frac{\dot{\varphi}.r}{L}\sin(\beta) \end{pmatrix} = \begin{pmatrix} \dot{\varphi}.r.\cos(\theta)\cos(\beta) - \dot{\varphi}.r.\sin(\theta)\sin(\beta) \\ \dot{\varphi}.r.\sin(\theta)\cos(\beta) - \dot{\varphi}.r.\cos(\theta)\sin(\beta) \\ \frac{\dot{\varphi}.r}{L}\sin(\beta) \end{pmatrix} \tag{4.8}$$

Assim pode-se escrever o modelo cinemático da seguinte forma:

$$\dot{x} = \dot{\varphi}.r.\cos(\theta + \beta) \tag{4.9}$$

$$\dot{y} = \dot{\varphi}.r.\sin(\theta + \beta) \tag{4.10}$$

$$\dot{\theta} = \frac{\dot{\varphi}.r}{L}\sin(\beta) \tag{4.11}$$

Geralmente em SLAM, existe um modelo que estabelece uma relação entre a pose anterior e a pose atual do robô dado um conjunto de dados provenientes de sensores proprioceptivos. Esta técnica de estimação de posicionamento relativo de robôs móveis baseado apenas em medidas da mudança de configuração no espaço articular do robô é comumente chamada de função de odometria. Grande parte destes modelos baseiam-se na integração discreta do modelo cinemático direto [123], mas existem outros modelos mais complexos que aproximam o movimento do robô por um conjunto de arcos de circunferência [124].

Portanto, para obter uma estimativa da pose do robô no instante atual $\hat{\xi}_k$ baseado na sua pose no instante anterior $\hat{\xi}_{k-1}$ e na configuração $\mathbf{q}_k = (\varphi_k, \beta_k)^T$ do espaço articular do robô no instante atual, usa-se a integração de Euler do modelo cinemático direto dado nas Equações (4.9)-(4.11):

$$\hat{x}_k = \hat{x}_{k-1} + \Delta T V_k \cos(\theta_{k-1} + \beta_k) \tag{4.12}$$

$$\hat{y}_k = \hat{y}_{k-1} + \Delta T V_k \sin(\theta_{k-1} + \beta_k) \tag{4.13}$$

$$\hat{\theta}_k = \hat{\theta}_{k-1} + \frac{\Delta T V_k \sin(\beta_k)}{L} \tag{4.14}$$

em que ΔT denota a diferença entre os instantes de tempo $k-1$ e k, sendo que durante este período a velocidade de deslocamento linear V_k e o ângulo de orientação β_k da roda frontal são considerados constantes. Comumente estas variáveis são denominadas entradas de controle $\mathbf{u}_k = (V_k, \beta_k)^T$.

O modelo de odometria supõe que estão satisfeitas várias hipóteses:

- A discretização considera nulas as componentes de ordem superior do modelo cinemático;
- As rodas não derrapam durante o deslocamento do robô, sendo seu contato com o solo considerado pontual;
- O solo no qual o robô navega é perfeitamente plano;
- Os parâmetros geométricos do robô, como o raio da roda r e a distância entre eixos L, são perfeitamente conhecidos.

Em sistemas reais estas hipóteses não são satisfeitas e além disso, as medidas de velocidade V e orientação β da roda frontal são inevitavelmente contaminadas por ruídos. Isto implica em um aumento significativo do erro de posicionamento por odometria. Durante a simulação, para contar com essas imperfeições, as entradas de controle do modelo de odometria são alteradas por ruído aditivo gaussiano de média zero e matriz de covariâncias P_u.

Além do modelo de processo que faz uso do modelo cinemático do robô para obtenção de uma estimativa do vetor de estados proposto neste trabalho sobre SLAM, requere-se também um modelo de observação dos *landmarks*. Este modelo utiliza informações extraídas dos

landmarks e estabelece uma relação entre a pose do robô no instante atual e um conjunto de poses indicativas da trajetória realizada pelo robô. Visto que este conjunto de poses forma o vetor de estado completo do sistema, o modelo de observação pode ser usado para prover uma estimativa corrigida deste vetor de estados. Um situação em que o robô observa um *landmark* é colocada na Figura 4.4.

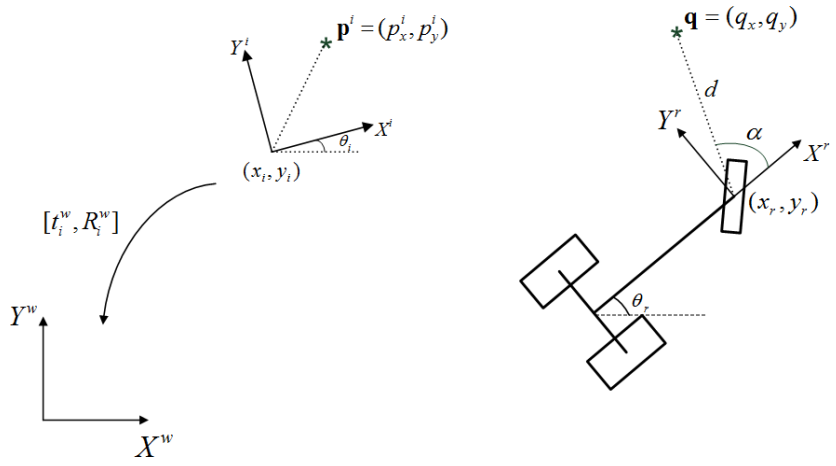


Figura 4.4: Diagrama esquemático do robô durante o processo de observação de um *landmark*.

O robô é equipado com um sensor capaz de detectar um *landmark* no ambiente e extrair uma medida de distância d e ângulo de visada α tomadas com relação a um sistema de coordenadas fixo no robô. O modelo de observação deve relacionar matematicamente as medições feitas no instante atual com os *landmarks* previamente armazenados em cada subregião da trajetória. Os *landmarks* são armazenados levando em conta suas coordenadas cartesianas 2D referenciadas a um sistema de coordenadas local. A transformação entre a representação retangular dada pelas coordenadas (x, y) de cada *landmark* e a representação através de (d, α) é dada por:

$$d = \sqrt{q_x^2 + q_y^2} \tag{4.15}$$

$$\alpha = \tan^{-1}\left(\frac{q_y}{q_x}\right) \tag{4.16}$$

O modelo de observação depende explicitamente das coordenadas locais de cada *landmark*, da pose do robô quando da observação destes *landmarks* e da pose atual do robô. Geralmente, este modelo envolve um processo de transformação de coordenadas cujos parâmetros dependem das poses do robô em cada subregião da trajetória. A Figura 4.4 ilustra o procedimento envolvido na obtenção deste modelo. O sistema de coordenadas local $X^i \mathrm{x} Y^i$ é obtido a partir da pose do robô $\xi_i = (x_i, y_i, \theta_i)^T$ para a i-ésima subregião da trajetória e o ponto $\mathbf{p}^i = (p_x^i, p_y^i)^T$ representa algum *landmark* com relação ao sistema de coordenadas i. Representando o sistema de coordenadas fixo no robô por sua pose $\xi_r = (x_r, y_r, \theta_r)^T$, o modelo de observação dos *landmarks* pode ser obtido com o auxílio de uma transformação

de coordenadas dada por um vetor de translação t_i^w e uma matriz de rotação R_i^w, isto é,

$$p^w = R_i^w p^i + t_i^w \tag{4.17}$$

em que $\mathbf{p}^w = (p_x^w, p_y^w)^T$ representa as coordenadas do ponto $\mathbf{p}$ com relação ao sistema global e além disso,

$$R_i^w = \begin{pmatrix} \cos(\theta_i) & -\sin(\theta_i) \\ \sin(\theta_i) & \cos(\theta_i) \end{pmatrix} \tag{4.18}$$

$$t_i^w = \begin{pmatrix} x_i \\ y_i \end{pmatrix} \tag{4.19}$$

Logo escreve-se,

$$p_x^w = p_x^i \cos(\theta_i) - p_y^i \sin(\theta_i) + x_i \tag{4.20}$$

$$p_y^w = p_x^i \sin(\theta_i) + p_y^i \cos(\theta_i) + y_i \tag{4.21}$$

Portanto, se $\hat{\mathbf{y}} = (\hat{d}, \hat{\alpha})^T$ é a saída do modelo de observação pode-se escrever:

$$\hat{d} = \sqrt{(p_x^w - x_r)^2 + (p_y^w - y_r)^2} \tag{4.22}$$

$$\hat{\alpha} = \tan^{-1}\left(\frac{p_y^w + y_r}{p_x^w + x_r}\right) - \theta_r \tag{4.23}$$

4.3 SIMULAÇÃO

Uma vez definidos os modelos a serem utilizados durante o processo de estimação, uma simulação do ambiente de navegação foi construída em *Matlab™* para realização de testes e avaliação da metodologia de SLAM proposta nesta dissertação. O ambiente de simulação foi desenvolvido em um plano 2D, de modo que a configuração deste ambiente é caracterizada por uma trajetória a ser seguida pelo robô e por um conjunto de pontos que denotam os *landmarks* espalhados em torno do robô. A Figura 4.5 ilustra o ambiente de simulação utilizado durante os testes.

Na Figura 4.5 a marca triangular de cor vermelha representa o corpo do robô, denotando sua posição e orientação iniciais. O caminho tracejado em vermelho é composto por segmentos de retas que interligam os pontos de passagem a serem alcançados pelo robô durante o deslocamento ao longo da sua trajetória. Por último, as marcações em azul indicam os *landmarks* espalhados pelo ambiente.

Durante a simulação, o robô é levado a atingir todos os pontos de passagem da trajetória (pontos marcados em preto) de forma que ele descreva um caminho similar ao traçado dado pela linha pontilhada.

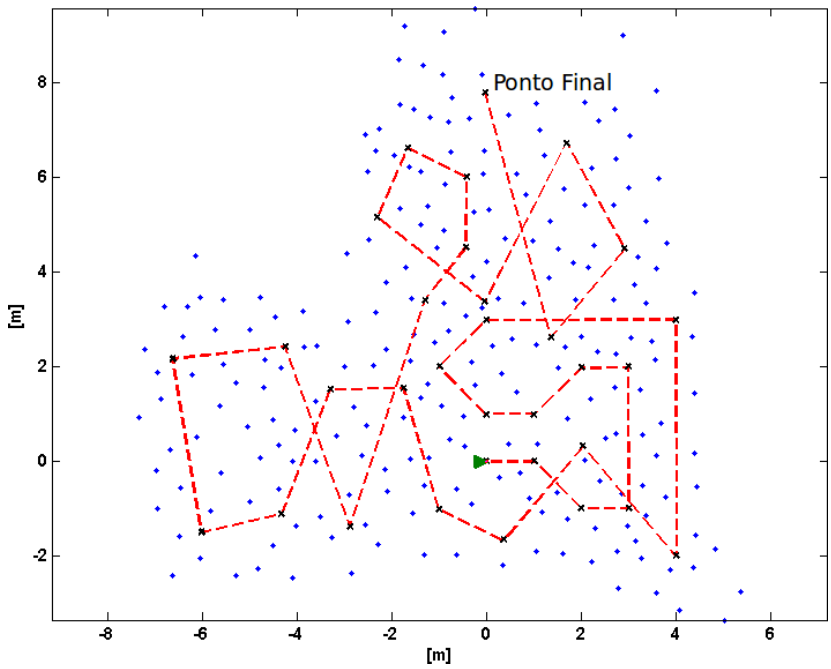


Figura 4.5: Ambiente de simulação em *Matlab*^{TM}

Uma vez alcançado o último ponto de passagem, o robô retorna ao estado inicial de forma que seja possível repetir toda a trajetória novamente. À medida que o robô navega pelo ambiente de simulação os *landmarks* vão sendo identificados e armazenados em cada subregião da trajetória. Além disso, as informações armazenadas sobre os *landmarks* formam um conjunto de submapas referenciados a sistemas de coordenadas locais que são representados pelas poses do robô ao longo da trajetória.

Com a finalidade de demonstrar a eficiência do método proposto aqui nesta dissertação, os resultados obtidos a partir da formulação híbrida do problema de SLAM, bem como a utilização do filtro IMM modificado conforme descrito na Seção 3.6 foi comparado com os resultados obtidos por meio da abordagem clássica, comumente denominada FKE-SLAM. Tradicionalmente em SLAM, utiliza-se como estimador o famoso Filtro de Kalman Estendido (FKE) e o sistema é formulado levando em consideração um único modelo em espaço de estados. Também é feita uma comparação do algoritmo híbrido de SLAM com uma técnica própria para grandes ambientes, FastSLAM, que foi detalhadamente descrita na Seção 2.1.4. A seguir serão apresentados alguns detalhes dos modelos matemáticos utilizados durante a simulação, bem como os principais resultados obtidos com os algoritmos IMM-SLAM, FastSLAM e EKF-SLAM.

4.3.1 Configurações básicas do simulador

O simulador desenvolvido neste trabalho é configurável de forma que a trajetória descrita pelo robô e a quantidade e posição de *landmarks* no ambiente podem ser alterados de acordo com o que for desejado. Além destas características, outras configurações relacionadas prin-

cipalmente com o algoritmo de estimação podem ser alteradas. Desta forma, o simulador conta com um arquivo que permite vários ajustes com relação ao algoritmo de SLAM utilizado, tais como parâmetros de controle (velocidade de translação, taxa máxima de rotação e dimensões físicas do robô), ruídos dos sensores, incertezas associadas aos modelos de processo e de medição, etc.

4.3.2 Modelo de medição

Além disso, o modelo de medição do sistema relaciona matematicamente as observações feitas no instante atual com as informações previamente armazenadas em cada uma das subregiões da trajetória. Desta maneira, fica evidente que a função de medição depende de qual subconjunto de dados E_i o robô está tomando as medidas. Então, para cada subregião da trajetória a função de medição $h_{s_k}(\mathbf{x}_k)$ assume um determinado formato e pode ser escrita conforme mostrado na Equação (3.20).

Fundamentalmente, a aplicação da função de medição envolve uma transformação de coordenadas entre o sistema de referência fixo no robô e os sistemas de referência locais representativos de cada subregião da trajetória. Matematicamente, esta transformação pode ser realizada com auxílio de matrizes que simbolizam operações de rotação e translação cujos parâmetros dependem explicitamente de qual subregião o robô está tomando as medidas.

4.3.3 Resultados da simulação

A nível de comparação, foram tomadas algumas medidas com relação a três algoritmos que apresentam a formulação do problema de SLAM e que distinguem-se principalmente na técnica de estimação utilizada. O algoritmo tradicional, FKE-SLAM, bem como a abordagem FastSLAM colocada na Seção 2.1.4 foram avaliadas e comparadas com o método proposto nesta dissertação, denominado IMM-SLAM, de forma que fosse possível demonstrar a eficiência do paradigma de modelagem de sistemas híbridos. Os resultados mostrados nos gráficos abaixo nos ajudam a entender melhor as diferenças entre estas técnicas e as vantagens incorporadas pelas abordagem híbrida.

Para mostrar uma indicação da consistência das estimativas do filtro IMM-SLAM e desta forma mostrar que não há divergência no mapa, calcula-se uma medida relativa de dispersão dos *landmarks*. Uma característica importante que deve ser mantida é que após um certo tempo de operação as distâncias entre os *landmarks* devem permanecer constantes sugerindo desta forma uma convergência para o mapa global. Isto mostra que a atualização de uma das poses indicativas da trajetória implica na atualização da trajetória como um todo, fazendo com que os *landmarks* associados a cada subregião sejam também atualizados adequadamente. A Figura 4.6 coloca uma trajetória de teste na qual o robô é acionado exaustivamente e os *landmarks* são observados por um longo período de tempo sob vários pontos de vista di-

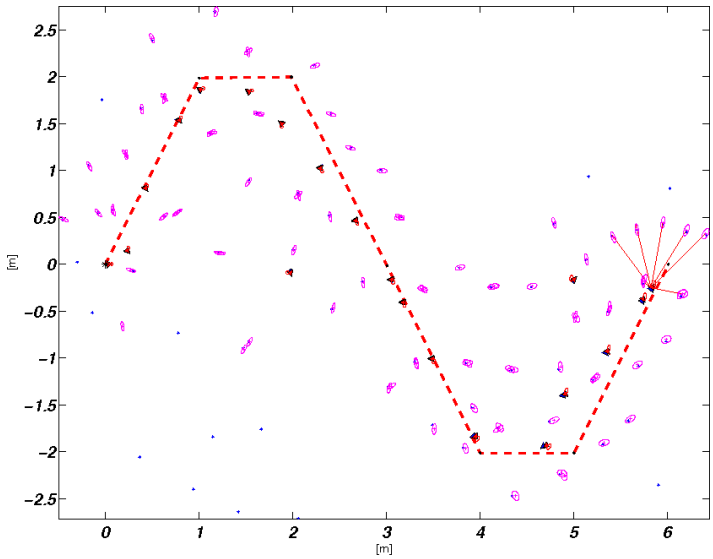


Figura 4.6: Aplicação do IMM-SLAM em um ambiente de simulação.

ferentes. Fica evidente ao longo da trajetória as poses e os *landmarks* estimados que formam um conjunto de submapas locais associados a cada subregião da trajetória.

Durante todo o tempo de operação do robô ao longo da trajetória mostrada na Figura 4.6 foram calculadas as distâncias relativas entre os *landmarks* do mapa e estes resultados estão apresentados logo a seguir na Figura 4.7. Os dados da Figura 4.7 foram obtidos durante várias voltas ao longo da trajetória da Figura 4.6 e mostram que mesmo durante um elevado tempo de operação e diferentes pontos de observação dos *landmarks* as distâncias entre os *landmarks* permanecem fixas. Este resultado é bastante interessante pois indica uma convergência para o mapa global.

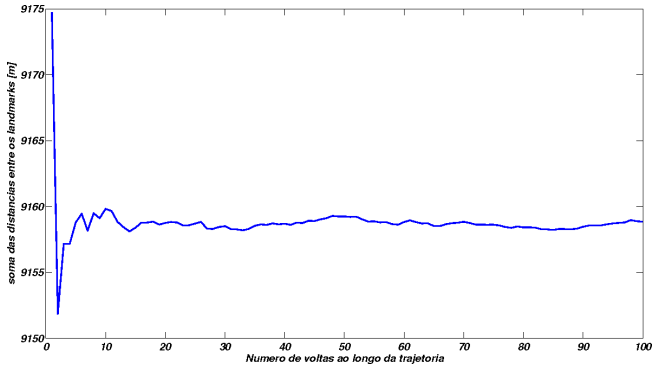


Figura 4.7: Soma das distâncias entre os *landmarks* do mapa.

Resultados semelhantes foram obtidos com o algoritmo FastSLAM para o mesmo trajeto mostrado na Figura 4.6. A Figura 4.8 a seguir mostra a aplicação do algoritmo FastSLAM no ambiente de simulação. Nesta figura, evidencia-se um conjunto de partículas que guardam uma estimativa da trajetória realizada pelo robô, bem como uma representação da localização dos *landmarks* no ambiente. Sob as mesmas condições de operação gerou-se uma medida de

dispersão dos *landmarks* considerando a aplicação do FastSLAM. O gráfico da Figura 4.9 mostra a soma das distâncias entre os *landmarks* do mapa.

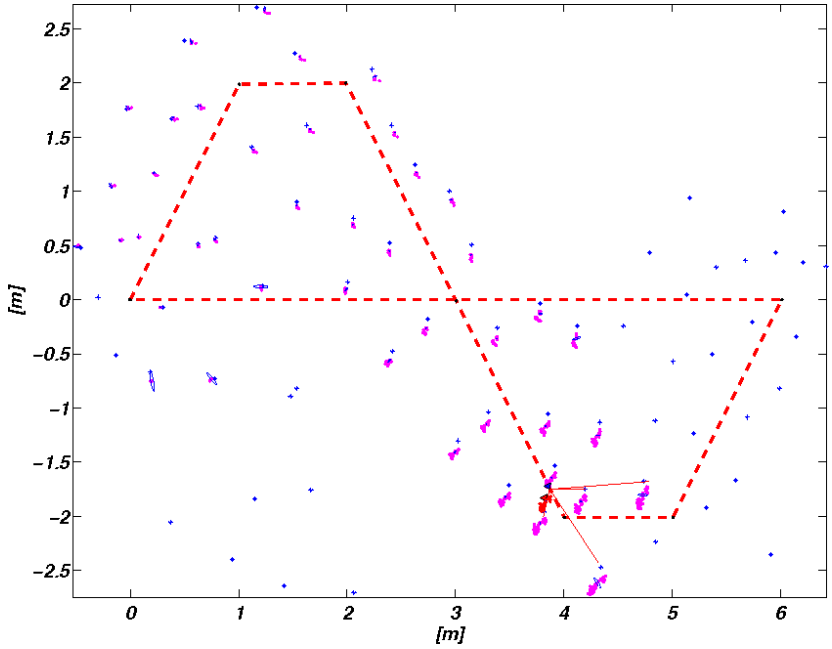


Figura 4.8: Aplicação do algoritmo FastSLAM à trajetória mostrada na Figura 4.6.

Comparando os resultados das Figuras 4.7 e 4.9 nota-se que em ambos os casos o mapa global convergiu e as distâncias entre os *landmarks* permaneceram constantes. Contudo existe uma diferença entre a soma das distâncias para o algoritmo FastSLAM e a abordagem híbrida IMM-SLAM. Esta disparidade pode ser explicada pelo fato de que no algoritmo IMM-SLAM existe uma sobreposição entre subregiões da trajetória fazendo com que alguns *landmarks* possuam mais de uma representação dependendo da sua pertinência com relação às subregiões. Por exemplo, de acordo com a abordagem proposta neste trabalho uma mesma estrutura física real no ambiente pode pertencer a duas subregiões distintas e isto faz com que este *landmark* tenha duas identidades, ou seja, uma representação para cada subregião. Portando, é de se esperar que a soma total das distâncias entre os *landmarks* seja maior para o algoritmo híbrido IMM-SLAM.

Uma grande vantagem introduzida pela abordagem híbrida ao problema de SLAM é a redução do custo computacional da solução possibilitando a aplicação desta solução em grandes ambientes com um elevado número de *landmarks*. Fundamentalmente, a formulação híbrida desenvolvida para o problema de SLAM sugere que tanto a etapa de associação de dados quanto as outras etapas do processo de estimação podem ser realizadas com apenas um subconjunto de todos os dados armazenados até o momento, ou seja, pode-se excluir do processo de filtragem as subregiões que possuem baixa probabilidade de serem visitadas, diminuindo o tempo de computação e consequentemente reduzindo a complexidade computacional do algoritmo. Para verificar esta característica, elaborou-se uma trajetória em que o robô fosse levado a explorar uma grande área de forma que o número de *landmarks* no am-

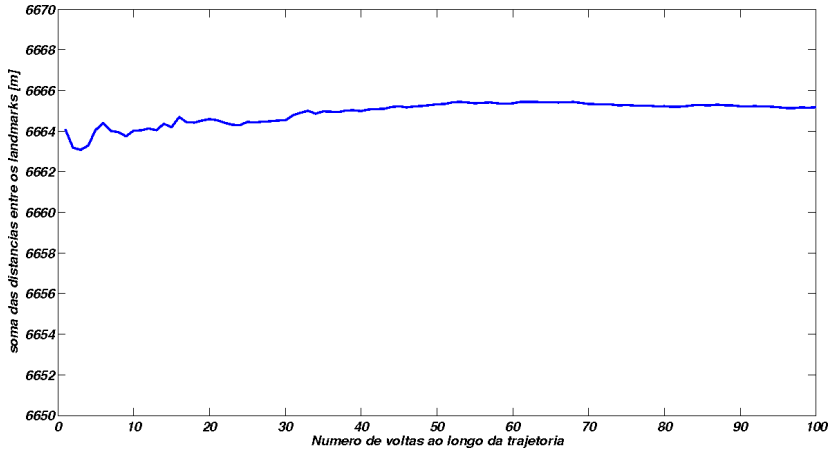


Figura 4.9: Soma das distâncias entre os *landmarks* no algoritmo FastSLAM.

biente aumentasse significativamente ao longo do tempo. A Figura 4.10 mostra a quantidade total de *landmarks* no mapa e a quantidade de *landmarks* efetivamente considerados durante o processo de estimação.

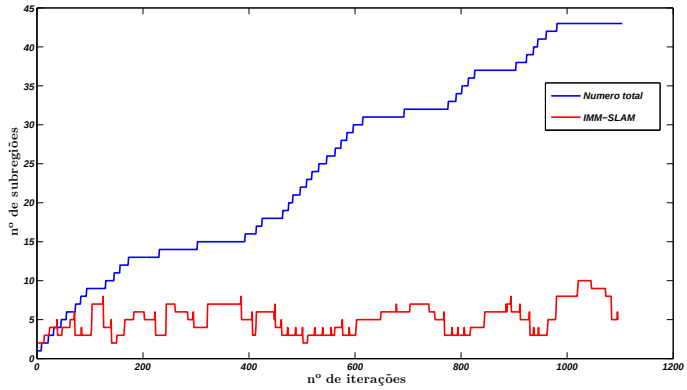


Figura 4.10: Número efetivo de *landmarks* em cada iteração do filtro IMM-SLAM.

De acordo com a Figura 4.10 fica evidente que o número efetivo de *landmarks* durante o processo de estimação permaneceu praticamente constante com o tempo, ao passo que, o número total de *landmarks* no mapa aumentou significativamente. Obviamente que o resultado mostrado na Figura 4.10 afeta profundamente o tempo de computação das etapas do algoritmo de estimação. Esta afirmação pode ser comprovada pela análise do gráfico da Figura 4.11.

Nesta figura mostra-se o tempo transcorrido para cada iteração dos filtros. No cálculo deste intervalo de tempo estão computadas todas as etapas do processo de estimação de cada filtro e nota-se claramente que a formulação híbrida para o problema de SLAM contribuiu significativamente para a redução da complexidade computacional do algoritmo. Ainda de acordo com o gráfico, esta característica fica mais acentudada à medida que o número de subregiões aumenta, ou seja, quando o robô é levado a navegar por grandes ambientes. Isto

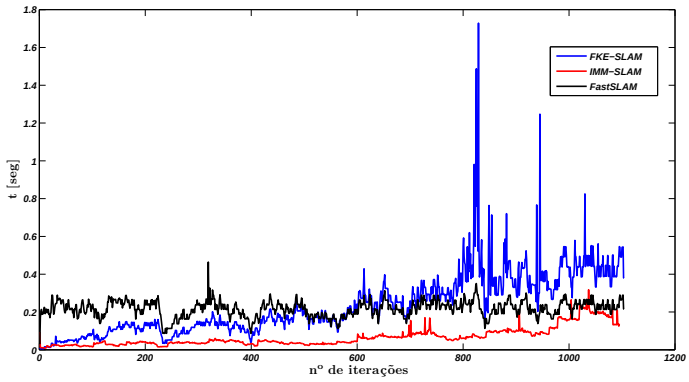


Figura 4.11: Tempo transcorrido em cada iteração.

mostra a viabilidade e a aplicabilidade do paradigma de sistemas dinâmicos híbridos para problema de localização e mapeamento em robótica móvel, principalmente quando o robô trabalha em ambientes de grande escala e por um longo período de operação.

5 CONCLUSÕES

5.1 CONSIDERAÇÕES FINAIS

Este trabalho tratou principalmente do tema de localização e mapeamento simultâneos em robótica móvel. Neste sentido, o robô é levado a navegar pelo ambiente ao seu redor e em um contexto de estimação estocástica é necessário prover estimativas da pose atual do robô, bem como de um conjunto de poses anteriores que compõem a trajetória percorrida até o momento. Além disso, faz-se necessário construir uma representação do mundo em volta do robô gerando um mapa que evidencia um conjunto de características notáveis do ambiente externo.

Durante a revisão bibliográfica procurou-se mostrar as principais dificuldades encontradas em resolver o problema de SLAM, bem como algumas metodologias abordadas durante todos esses anos. Fundamentalmente, este trabalho enfatizou uma abordagem desacoplada do problema de SLAM na qual o que se procura estimar são as poses do robô ao longo da trajetória utilizando dados previamente armazenados que formam um conjunto de submapas locais associados a alguma subregião do ambiente.

A grande contribuição desta dissertação está na formulação do problema de SLAM seguindo um paradigma de sistemas dinâmicos híbridos, mais especificamente do caso particular de sistemas a múltiplos modelos, em que uma variável discreta de estado denota possíveis mudanças de modelo matemático entre os diferentes modos de operação. Neste sentido, privilegiou-se o detalhamento matemático do amplamente difundido filtro IMM, algoritmo de notável e reconhecida importância até os dias de hoje em aplicações de sistemas híbridos a situações práticas, tendo ele influenciado de forma decisiva os resultados apresentados neste trabalho. De fato, verificou-se que uma pequena modificação no algoritmo do filtro IMM proporcionou um significativo ganho de desempenho com relação à complexidade computacional do algoritmo quando comparado com a abordagem mais tradicional que leva em conta apenas um único modelo matemático do sistema em espaço de estados. Este novo método também se mostrou mais eficiente mesmo quando comparado com o FastSLAM [10, 11], uma solução notadamente voltada para SLAM de grandes ambientes.

Pretende-se que os resultados trazidos neste trabalho motivem um estudo mais aprofundado da aplicação de sistemas dinâmicos híbridos no campo da robótica móvel. Em nenhum momento desta dissertação são feitas demonstrações teóricas mais aprofundadas com relação à formulação utilizada para resolver o problema de SLAM, contudo a idéia inicial desenvolvida aqui pode ser trabalhada de forma a demonstrar matematicamente que a aplicação de uma formulação híbrida para o problema de SLAM pode ser uma solução bem mais elegante e que generaliza as abordagens difundidas até o momento, principalmente o

FastSLAM [10, 11], HMT-SLAM [86] e Graph-SLAM [34].

5.2 PROPOSTAS PARA TRABALHOS FUTUROS

O trabalho desenvolvido até aqui não gera uma representação global do ambiente. Portanto, uma primeira tarefa a se cumprir de forma a complementar os resultados obtidos nesta dissertação seria o estudo e implementação de elementos acessórios ao processo de estimação, tal como a manutenção de um mapa global do ambiente. Isto implica no desenvolvimento de um método robusto que integrasse todos os dados disponíveis, tais como os conjuntos definidos no Capítulo 3 de forma que o mapa global pudesse ser atualizado de tempos em tempos levando em consideração a natureza dinâmica do ambiente ao redor do robô. Na verdade, esta etapa pode ser incorporada separadamente ao algoritmo de estimação, dado que este procedimento não possui uma alta taxa de requisição e é sobretudo necessário para fins de planejamento de rotas.

Por fim, de acordo com o que foi mencionado anteriormente os resultados apresentados nesta dissertação dão margem a uma investigação mais profunda da utilização do paradigma de modelagem de sistemas híbridos na resolução do problema de SLAM. Esta formulação pode levar à construção de um arcabouço geral em que diferentes sistemas podem ser avaliados, independentemente da plataforma (robô, sensores e atuadores) ou dos algoritmos de estimação utilizados.

[1] M. Maimone, J. Biesiadecki, E. Tunstel, Y. Cheng, and C. Leger, "Surface navigation and mobility intelligence on the mars exploration rovers," in *Inteligence for Space Robotics*. TSI Press, 2006, pp. 45–69.

[2] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, September 2005.

[3] Z. Kurt-Yavuz and S. Yavuz, "A comparison of EKF, UKF, FastSLAM2.0, and UKF-based FastSLAM algorithms," in *IEEE 16th International Conference on Intelligent Engineering Systems (INES)*, Jun 2012, pp. 37–43.

[4] R. Smith, M. Self, and P. Cheeseman, "A stochastic map for uncertain spatial relationships," in *Proceedings of the 4th International Symposium on Robotics Research*, 1988, pp. 467–474.

[5] J. Artieda, J. Sebastian, P. Campoy, J. Correa, I. Mondragón, C. Martínez, and M. Olivares, "Visual 3-D SLAM from UAVs," *Journal of Intelligent and Robotic Systems*, vol. 55, no. 4-5, pp. 299–321, Aug 2009.

[6] A. P. Gee, D. Chekhlov, A. Calway, and W. Mayol-Cuevas, "Discovering higher level structure in visual SLAM," *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 980–990, Oct 2008.

[7] N. Tomomi and T. Kanji, "Dictionary-based map compression using geometric priors," in *IEEE International Conference on Robotics and Biomimetics (ROBIO).*, Dec 2011, pp. 2599–2604.

[8] J. Schwendner, "Map segmentation based SLAM using embodied data," in *IEEE Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, Sept 2012, pp. 352–357.

[9] L. Paz, J. Tardos, and J. Neira, "Divide and conquer: EKF-SLAM in $O(n)$," *IEEE Transactions on Robotics.*, vol. 24, no. 5, pp. 1107–1120, Oct. 2008.

[10] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "FastSLAM: A factored solution to the simultaneous localization and mapping problem," in *In Proceedings of the AAAI National Conference on Artificial Intelligence*, 2002, pp. 593–598.

[11] M. Montemerlo and S. Thrun, "FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges," in *In Proc. of the Int. Conf. on Artificial Intelligence (IJCAI)*, 2003, pp. 1151–1156.

[12] M. Wu, F. Pei, and H. Ju, "A SLAM system based on improved distributed UPF," in *International Conference on Mechatronics and Automation (ICMA)*, Aug. 2012, pp. 2606–2611.

[13] C. Qiu, X. Zhu, and X. Zhao, "Vision-based unscented FastSLAM for mobile robot," in *10th World Congress on Intelligent Control and Automation (WCICA)*, Jul. 2012, pp. 3758–3763.

[14] J. A. Castellanos, R. Martínez-cantín, J. D. Tardós, and J. Neira, "Robocentric map joining: Improving the consistency of EKF-SLAM," *Robotics and Autonomous Systems*, vol. 55, pp. 21–29, 2007.

[15] J. A. Castellanos, J. D. Tardós, and G. Schmidt, "Building a global map of the environment of a mobile robot: the importance of correlations," in *IEEE International Conference on Robotics and Automation*, 1997, pp. 1053–1059.

[16] S. J. Julier and J. K. Uhlmann, "A counter example to the theory of simultaneous localization and map building," in *IEEE International Conference on Robotics and Automation*, 2001.

[17] S. Li and P. Ni, "Square-root Unscented Kalman filter based simultaneous localization and mapping," in *IEEE International Conference on Information and Automation (ICIA)*, Jun. 2010, pp. 2384–2388.

[18] M. Anjum, J. Park, W. Hwang, H. il Kwon, J. hyeon Kim, C. Lee, K. soo Kim, and D. il Danr Cho, "Sensor data fusion using Unscented Kalman filter for accurate localization of mobile robots," in *International Conference on Control Automation and Systems (ICCAS)*, Oct. 2010, pp. 947–952.

[19] L. Qu, S. He, and Y. Qu, "An SLAM algorithm based on improved UKF," in *24th Chinese Control and Decision Conference (CCDC)*, May. 2012, pp. 4154–4157.

[20] S. Kuifeng, D. ZhiDong, and H. Zhen, "Novel SLAM algorithm for UGVs based on Unscented Kalman filtering," in *IEEE International Conference on Computer Science and Automation Engineering (CSAE)*, vol. 3, May. 2012, pp. 63–67.

[21] A. Monjazeb, J. Sasiadek, and D. Necsulescu, "An approach to autonomous navigation based on Unscented HybridSLAM," in *17th International Conference on Methods and Models in Automation and Robotics (MMAR)*.

[22] X. Ji, H. Zhang, D. Hai, and Z. Zheng, "An incremental SLAM algorithm with backtracking revisable data association for mobile robots," in *IEEE International Conference on Mechatronics and Automation*, Aug. 2008, pp. 831–839.

[23] J. Xiong and Y. Li, "A hybrid data association strategy in SLAM," in *4th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)*, vol. 1, Aug. 2012, pp. 348–351.

[24] S. Ma, S. Guo, M. Wang, and B. Li, "Data association for a hybrid metric map representation," in *IEEE Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, Sept. 2012, pp. 168–173.

[25] C. Estrada, J. Neira, and J. Tardos, "Hierarchical SLAM: Real-time accurate mapping of large environments," *IEEE Transactions on Robotics*, vol. 21, no. 4, pp. 588–596, Aug. 2005.

[26] B. Zhao and X. Zhu, "Fast large-scale SLAM with improved accuracy in mobile robot," in *International Conference on Robotics and Biomimetics (ROBIO)*, Dec. 2010, pp. 791–796.

[27] D. Moratuwage, B.-N. Vo, and D. Wang, "A hierarchical approach to the multi-vehicle SLAM problem," in *International Conference on Information Fusion (FUSION)*, Jul. 2012, pp. 1119–1125.

[28] J. D. Tardos, J. Neira, P. M. Newman, and J. J. Leonard, "Robust mapping and localization in indoor environments using sonar data," *Int. J. Robotics Research*, vol. 21, pp. 311–330, 2002.

[29] S. B. Williams, G. Dissanayake, and H. Durrant-Whyte, "An efficient approach to the simultaneous localization and mapping problem," in *IEEE International Conference on Robotics and Automation*, vol. 1, Aug. 2002, pp. 406–411.

[30] L. Zhao, S. Huang, L. Yan, J. Wang, G. Hu, and G. Dissanayake, "Large-scale monocular SLAM by local bundle adjustment and map joining," in *International Conference on Control Automation Robotics Vision (ICARCV)*, Dec. 2010, pp. 431–436.

[31] J. Leonard and P. Newman, "Consistent, convergent, and constant-time SLAM," in *Proceedings of the 18th international joint conference on Artificial intelligence*, 2003, pp. 1143–1150.

[32] M. Bosse, P. Newman, J. Leonard, and S. Teller, "Simultaneous localization and map building in large-scale cyclic environments using the atlas framework," *The International Journal of Robotics Research*, vol. 23, no. 12, pp. 1113–1139, 2004.

[33] S. Huang, Z. Wang, and G. Dissanayake, "Exact state and covariance sub-matrix recovery for submap based sparse EIF-SLAM algorithm," May. 2008, pp. 1868 –1873.

[34] S. Thrun and M. Montemerlo, "The GraphSLAM algorithm with applications to large-scale mapping of urban structures," *International Journal on Robotics Research*, vol. 25, no. 5, pp. 403–430, 2006.

[35] J. Folkesson and H. Christensen, "Closing the loop with Graphical SLAM," *IEEE Transactions on Robotics*, vol. 23, no. 4, pp. 731–741, Aug. 2007.

[36] H. Chang, C. Lee, Y.-H. Lu, and Y. Hu, "P-SLAM: Simultaneous localization and mapping with environmental-structure prediction," *IEEE Transactions on Robotics*, vol. 23, no. 2, pp. 281–293, Apr. 2007.

[37] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with rao-blackwellized particle filters," *IEEE Transactions on Robotics*, vol. 23, no. 1, pp. 34–46, Feb. 2007.

[38] A. Selvatici and A. Costa, "Fast loopy belief propagation for topological SAM," in *IEEE International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2007, pp. 664–669.

[39] F. Dellaert and M. Kaess, "Square root SAM: Simultaneous localization and mapping via square root information smoothing," *International Journal of Robotics Reasearch*, vol. 25, no. 12, pp. 1181–1203, Dec. 2006.

[40] R. Eustice, H. Singh, and J. Leonard, "Exactly sparse delayed-state filters for view-based SLAM," *IEEE Transactions on Robotics*, vol. 22, no. 6, pp. 1100–1114, Dec. 2006.

[41] E. Olson, J. Leonard, and S. Teller, "Fast iterative alignment of pose graphs with poor initial estimates," in *IEEE International Conference on Robotics and Automation (ICRA)*, May. 2006, pp. 2262–2269.

[42] G. Grisetti, D. Lodi Rizzini, C. Stachniss, E. Olson, and W. Burgard, "Online constraint network optimization for efficient maximum likelihood map learning," in *IEEE International Conference on Robotics and Automation (ICRA)*, May. 2008, pp. 1880–1885.

[43] R. Goebel, R. Sanfelice, and A. Teel, "Hybrid dynamical systems," *IEEE Control Systems*, vol. 29, no. 2, pp. 28–93, Apr. 2009.

[44] S. Paoletti, A. L. Juloski, G. Ferrari-Trecate, and R. Vidal, "Identification of hybrid systems: A tutorial." *Eur. J. Control*, vol. 13, no. 2-3, pp. 242–260, 2007.

[45] R. Jian, W. Xiu, Z. Xiaochum, and Z. Haitao, "A hybrid neural network-based IE and IMM architecture for target tracking," in *Workshop on Power Electronics and Intelligent Transportation System*, Aug. 2008, pp. 214–217.

[46] M. Hartman, N. Bauer, and A. Teel, "Robust finite-time parameter estimation using a hybrid systems framework," *IEEE Transactions on Automatic Control*, vol. 57, no. 11, pp. 2956–2962, Nov. 2012.

[47] J. Civera, A. Davison, and J. Montiel, "Interacting multiple model monocular SLAM," in *IEEE International Conference on Robotics and Automation*, May. 2008, pp. 3704–3709.

[48] Y. Yingmin and L. Ding, "Robot simultaneous localization and mapping based on non-linear interacting multiple model," in *International Workshop on Intelligent Systems and Applications*, May. 2009, pp. 1–6.

[49] E. Mazor, A. Averbuch, Y. Bar-Shalom, and J. Dayan, "Interacting multiple model methods in target tracking: a survey," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 34, no. 1, pp. 103–123, Jan. 1998.

[50] H. Blom and Y. Bar-Shalom, "The interacting multiple model algorithm for systems with Markovian switching coefficients," *IEEE Transactions on Automatic Control*, vol. 33, no. 8, pp. 780–783, Aug. 1988.

[51] X.-R. Li and Y. Bar-Shalom, "Multiple-model estimation with variable structure," *IEEE Transactions on Automatic Control*, vol. 41, no. 4, pp. 478–493, Apr. 1996.

[52] C. E. Seah and I. Hwang, "State estimation for stochastic linear hybrid systems with continuous-state-dependent transitions: An IMM approach," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 45, no. 1, pp. 376–392, Jan. 2009.

[53] H. A. P. Blom, "An efficient filter for abruptly changing systems," in *The 23rd IEEE Conference on Decision and Control*, vol. 23, Dec. 1984, pp. 656–658.

[54] G. Campion, G. Bastin, and B. d'Andréa Novel, "Structural properties and classification of kinematic and dynamic models of wheeled mobile robots." in *ICRA*, 1993, pp. 462–469.

[55] M. W. M. G. Dissanayake, P. Newman, S. Clark, H. F. Durrant-Whyte, and M. Csorba, "A solution to the simultaneous localization and map building (SLAM) problem," *IEEE Transactions on Robotics and Automation*, vol. 17, no. 3, pp. 229–241, June 2001.

[56] J. A. Castellanos, J. Neira, and J. D. Tardós, "Limits to the consistency of EKF-Based SLAM," in *5th IFAC Symposium on Intelligent Autonomous Vehicles*, July 2004.

[57] A. H. Jazwinski, *Stochastic Processes and Filtering Theory*. Academic Press, Apr. 1970.

[58] D. Simon, *Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches*. Wiley-Interscience, 2006.

[59] Y. Bar-Shalom and T. E. Fortmann, *Tracking and data association*. Academic Press, 1988.

[60] J. A. Castellanos and J. D. Tardós, *Mobile Robot Localization and Map Building: A Multisensor Fusion Approach*. Kluwer Academic Publishers, 2000.

[61] J. E. Guivant and E. M. Nebot, "Optimization of the simultaneous localization and map-building algorithm for real-time implementation," *IEEE Transactions on Robotics and Automation*, vol. 17, no. 3, pp. 242–257, June 2001.

[62] T. Bailey and H. Durrant-Whyte, "Simultaneous localization and mapping (SLAM): part II," *IEEE Robotics & Automation Magazine*, vol. 13, no. 3, pp. 108–117, Sept. 2006.

[63] J. J. Leonard and H. J. S. Feder, "Decoupled stochastic mapping," *IEEE Journal of Oceanic Engineering*, vol. 26, no. 4, pp. 561–571, October 2001.

[64] S. Thrun, Y. Liu, D. Koller, A. Ng, Z. Ghahramani, and H. Durrant-Whyte, "Simultaneous localization and mapping with sparse extended information filters," *International Journal of Robotics Research*, vol. 23, no. 7/8, 2004.

[65] V. Ila, J. Andrade-Cetto, R. Valencia, and A. Sanfeliu, "Vision-based loop closing for delayed state robot mapping." in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, december 2007, pp. 3892–3897.

[66] M. Kaess, A. Ranganathan, and F. Dellaert, "iSAM: Incremental smoothing and mapping," *IEEE Trans. on Robotics (TRO)*, vol. 24, no. 6, pp. 1365–1378, Dec. 2008.

[67] A. G. O. Mutambara, *Decentralized Estimation and Control for Multisensor Systems*, 1st ed. Boca Raton, FL, USA: CRC Press, Inc., 1998.

[68] R. Eustice, M. Walter, and J. Leonard, "Sparse extended information filters: Insights into sparsification," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, August 2005, pp. 641–648.

[69] M. Walter, R. Eustice, and J. Leonard, "Exactly sparse extended information filters for feature-based SLAM," *International Journal of Robotics Research*, vol. 26, no. 4, pp. 335–359, Apr. 2007.

[70] K. Ni, D. Steedly, and F. Dellaert, "Tectonic SAM: exact, out-of-core, submap-based SLAM," in *In Proc. IEEE International Conference on Robotics and Automation*, 2007, pp. 1678–1685.

[71] S. Huang, Z. Wang, and G. Dissanayake, "Sparse local submap joining filter for building large-scale maps." *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 1121–1130, april 2008.

[72] C. Cadena and J. Neira, "SLAM in $O(logn)$ with the combined kalman - information filter," *Robotics and Autonomous Systems*, vol. 58, no. 11, pp. 1207–1219, Nov. 2010.

[73] K. Konolige and M. Agrawal, "FrameSLAM: From bundle adjustment to real-time visual mapping." *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 1066–1077, May 2008.

[74] V. Ila, J. M. Porta, and J. Andrade-Cetto, "Information-based compact pose SLAM," *IEEE Transactions on Robotics*, vol. 26, no. 1, pp. 78–93, 2010.

[75] M. Montemerlo and S. Thrun, "Simultaneous localization and mapping with unknown data association using FastSLAM," in *ICRA*, 2003, pp. 1985–1991.

[76] T. Bailey, J. I. Nieto, J. E. Guivant, M. Stevens, and E. M. Nebot, "Consistency of the EKF-SLAM algorithm." in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2006, pp. 3562–3568.

[77] J. K. . Uhlmann, "Introduction to the algorithmics of data association in multiple-target tracking," in *Handbook of multisensor data fusion: theory and practice*. CRC Press, 2008, ch. 4.

[78] J. Folkesson and H. Christensen, "Graphical slam - a self-correcting map," in *In IEEE Intl. Conf. on Robotics and Automation (ICRA*, 2004, pp. 383–390.

[79] E. Olson, J. Leonard, and S. Teller, "Fast iterative optimization of pose graphs with poor initial estimates," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2006, pp. 2262–2269.

[80] G. Grisetti, R. Kümmerle, C. Stachniss, and W. Burgard, "A tutorial on graph-based slam," *IEEE Transactions on Intelligent Transportation Systems Magazine*, vol. 2, no. 4, pp. 31–43, 2010.

[81]

[82] M. Golfarelli, D. Maio, and S. Rizzi, "Elastic correction of dead-reckoning errors in map building," in *In Intl. Conf. on Intelligent Robots and Systems*, 1998, pp. 905–911.

[83] T. Duckett, "Learning globally consistent maps by relaxation," in *In Proceedings of the IEEE International Conference on Robotics and Automation*, 2000, pp. 3841–3846.

[84] U. Frese and G. Hirzinger, "Simultaneous localization and mapping - a discussion," 2001.

[85] K. Konolige, "Large-scale map-making," in *Proceedings of the National Conference on AI (AAAI)*, 2004.

[86] J.-L. Blanco, J.-A. Fernandez-Madrigal, and J. Gonzalez, "Toward a unified bayesian approach to hybrid metric-topological slam," *IEEE Transactions on Robotics*, vol. 24, no. 2, pp. 259–270, april 2008.

[87] J. L. Blanco, J. Gonzalez, and J. A. Fernández-madrigal, "Consistent observation grouping for generating metric-topological maps that improves robot localization," in *in ICRA06*, 2006, pp. 818–823.

[88] "Hierarchical map building using visual landmarks and geometric constraints," in *In Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2005, pp. 7–12.

[89] M. Montemerlo and S. Thrun, *FastSLAM: A Scalable Method for the Simultaneous Localization and Mapping Problem in Robotics*. Springer, 2007.

[90] K. Murphy, "Bayesian map learning in dynamic environments," in *In Neural Info. Proc. Systems (NIPS*. MIT Press, pp. 1015–1021.

[91] A. Doucet, S. Godsill, and C. Andrieu, "On sequential monte carlo sampling methods for bayesian filtering," *Statistics and Computing*, vol. 10, pp. 197–208, 2000.

[92] A. Doucet and A. M. Johansen, "A tutorial on particle filtering and smoothing: fifteen years later," 2011.

[93] F. Dellaert, D. Fox, W. Burgard, and S. Thrun, "Monte carlo localization for mobile robots," in *IEEE International Conference on Robotics and Automation (ICRA99)*, May 1999.

[94] D. B. Rubin, "Using the sir algorithm to simulate posterior distributions," in *Bayesian Statistics 3*. Oxford Univ. Press, 1998.

[95] R. Smith, M. Self, and P. Cheeseman, "Estimating uncertain spatial relationships in robotics," in *Autonomous Robot Vehicles*, I. J. Cox and G. T. Wilfong, Eds. Springer Verlag, 1990, pp. 167–193.

[96] C. Peter, T. James, D. M., N. P., D.-W. H., C. S., and C. M., "An experimental and theoretical investigation into simultaneous localisation and map building," in *Experimental Robotics VI*, ser. Lecture Notes in Control and Information Sciences. Springer Berlin / Heidelberg, 2000, vol. 250, pp. 265–274.

[97] D. Magill, "Optimal adaptive estimation of sampled stochastic processes," *Automatic Control, IEEE Transactions on*, vol. 10, no. 4, pp. 434–439, october 1965.

[98] W. Blair, G. Watson, and T. Rice, "Tracking maneuvering targets with an interacting multiple model filter containing exponentially-correlated acceleration models," in *System Theory, 1991. Proceedings., Twenty-Third Southeastern Symposium on*, march 1991, pp. 224–228.

[99] P. H. R. Q. A. Santana, "Filtragem estocástica para sistemas híbridos e suas aplicações em robótica aérea," Master's thesis, Universidade de Brasília, 2011.

[100] M. Hofbaur and B. Williams, "Hybrid estimation of complex systems," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 34, no. 5, pp. 2178–2191, october 2004.

[101] Y. Boers and H. Driessen, "A multiple model multiple hypothesis filter for markovian switching systems," *Automatica*, vol. 41, no. 4, pp. 709–716, 2005.

[102] C. E. Seah and I. Hwang, "Stability analysis of the interacting multiple model algorithm," in *American Control Conference, 2008*, june 2008, pp. 2415–2420.

[103] T. Kirubarajan and Y. Bar-Shalom, "Kalman filter versus imm estimator: when do we need the latter?" *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 39, no. 4, pp. 1452–1457, october 2003.

[104] Y. Boers and J. Driessen, "Interacting multiple model particle filter," *Radar, Sonar and Navigation, IEE Proceedings* -, vol. 150, no. 5, pp. 344–349, october 2003.

[105] I. Hwang, H. Balakrishnan, and C. Tomlin, "State estimation for hybrid systems: applications to aircraft tracking," *Control Theory and Applications, IEE Proceedings* -, vol. 153, no. 5, pp. 556–566, september 2006.

[106] A. Houles and Y. Bar-Shalom, "Multisensor tracking of a maneuvering target in clutter," *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 25, no. 2, pp. 176–189, march 1989.

[107] X. Li and Y. Bar-Shalom, "Design of interacting multiple model algorithm for tracking in air traffic control systems," in *Decision and Control, 1993., Proceedings of the 32nd IEEE Conference on*, vol. 1, december 1993, pp. 906–911.

[108] M. Yeddanapudi, Y. Bar-Shalom, and K. Pattipati, "Imm estimation for multitarget-multisensor air traffic surveillance," *Proceedings of the IEEE*, vol. 85, no. 1, pp. 80–96, 1997.

[109] P. Santana, G. Borges, and J. Ishihara, "Hybrid data fusion for 3d localization under heavy disturbances," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, october 2010, pp. 2425–2430.

[110] Y. Xue and T. Runolfsson, "State estimation and mode detection for stochastic hybrid system," in *Intelligent Control, 2008. ISIC 2008. IEEE International Symposium on*, september 2008, pp. 625–630.

[111] V. Jilkov and X. Li, "Online bayesian estimation of transition probabilities for markovian jump systems," *Signal Processing, IEEE Transactions on*, vol. 52, no. 6, pp. 1620–1630, june 2004.

[112] L. Bertuccelli and J. How, "Estimation of non-stationary markov chain transition models," in *Proceedings of 47th IEEE Conference on Decision and Control*, december 2008, pp. 55–60.

[113] J. A. Castellanos, J. M. M. Nez, J. Neira, and J. D. Tardos, "Experiments in multisensor mobile robot localization and map building," in *In 3rd IFAC Symposium on Intelligent Autonomous Vehicles, Mach'id*, 1998, pp. 173–178.

[114] H. Choset, K. M. Lynch, S. Hutchinson, G. A. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun, *Principles of Robot Motion: Theory, Algorithms, and Implementations*. Cambridge, MA: MIT Press, June 2005.

[115] A. J. Davison and N. Kita, "Sequential localisation and map-building in computer vision and robotics," in *Revised Papers from Second European Workshop on 3D Structure from Multiple Images of Large-Scale Environments*. Springer-Verlag, 2001, pp. 218–234.

[116] J. J. Leonard, H. F. Durrant-Whyte, and I. J. Cox, "Dynamic map building for an autonomous mobile robot," *Int. J. Rob. Res.*, vol. 11, no. 4, pp. 286–298, aug 1992.

[117] S. J. Julier and J. K. Uhlmann, "Using covariance intersection for slam," *Robot. Auton. Syst.*, vol. 55, no. 1, pp. 3–20, jan 2007.

[118] A. Doucet, N. Gordon, and V. Krishnamurthy, "Particle filters for state estimation of jump markov linear systems," *IEEE Transactions on Signal Processing*, vol. 49, no. 3, pp. 613–624, march 2001.

[119] J. J. Leonard, H. Jacob, and S. Feder, "A computationally efficient method for large-scale concurrent mapping and localization," in *Proceedings of the Ninth International Symposium on Robotics Research*. Springer-Verlag, 1999, pp. 169–176.

[120] R. Van der Merwe and E. Wan, "The square-root unscented kalman filter for state and parameter-estimation," in *Acoustics, Speech, and Signal Processing, 2001. Proceedings. (ICASSP). 2001 IEEE International Conference on*, vol. 6, 2001, pp. 3461–3464.

[121] C. Chui and G. Chen, *Kalman filtering with real-time applications*. Springer-Verlag New York, 1987.

[122] G. Dudek and M. Jenkin, *Computational principles of mobile robotics*. New York, NY, USA: Cambridge University Press, 2000.

[123] J. Borenstein, L. Feng, and C. J. Borenstein, "Measurement and correction of systematic odometry errors in mobile robots," *IEEE Transactions on Robotics and Automation*, vol. 12, pp. 869–880, 1996.

[124] C. M. Wang, “Autonomous robot vehicles,” I. J. Cox and G. T. Wilfong, Eds., 1990, ch. Location estimation and uncertainty analysis for mobile robots, pp. 90–95.

,