Roberto José Giordano Barra

Combinação de Visão Monocular e Sonares Esparsos para a Localização de Robôs Móveis

Dissertação apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Mestre em Engenharia Elétrica.

Roberto José Giordano Barra

Combinação de Visão Monocular e Sonares Esparsos para a Localização de Robôs Móveis

Dissertação apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Mestre em Engenharia Elétrica.

Área de concentração:
Sistemas Digitais

Orientador:
Prof. Dra. Anna Helena Reali
Costa

Este exemplar foi revisado e alterado em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, 16 de abril de 2007

Assinatura do autor

Assinatura da orientadora

Ficha Catalográfica

Barra, Roberto José Giordano

Combinação de Visão Monocular e Sonares Esparsos para a Localização de Robôs Móveis. São Paulo, 2007. 183 p.

Dissertação (Mestrado) — Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Computação e Sistemas Digitais.

1. Filtros de Kalman. 2. Robótica. 3. Visão Computacional. I. Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Computação e Sistemas Digitais. II. t.

Agradecimentos

Agradeço

À minha orientadora, Anna Helena Reali Costa, pela paciência e dedicação.

Aos meus pais, pelo apoio.

E à minha namorada, pela companhia, suporte e compreensão.

Resumo

Um componente fundamental no sistema de um robô móvel consiste na habilidade de localizar-se acuradamente, o que envolve estimar sua postura em relação a uma representação global do espaço. A especificação geral de uma abordagem de localização baseada em dados sensoriais possui uma estimativa inicial da postura do robô e usa os dados coletados pelos sensores, em conjunto com um mapa do ambiente, para produzir uma estimativa mais precisa da postura, que oferece um valor de maior confiança em relação à postura real do robô. Uma dificuldade é que os dados sensoriais são corrompidos por erros de medidas derivados de diversas fontes, como ruídos, quantização, dispositivos de digitalização, deslizamentos do robô, entre outras. Sensores distintos medem diferentes propriedades físicas, corrompidas por diversos erros de medida. O uso de dados oriundos de vários sensores fornece informação redundante e complementar, que pode ser processada para derivar uma estimativa combinada com o objetivo de aumentar a confiança na estimativa final da postura. Nesta dissertação é proposto ELViS, um sistema que estima a localização de um robô móvel equipado com odômetros, uma câmera de vídeo e um semi-anel frontal de 8 sonares, o qual opera, com sucesso, em um ambiente interno, estruturado e estático. Assume-se que o robô navega sobre uma superfície plana e que diversos segmentos de retas possam ser identificados nas imagens do ambiente. Para aumentar a seletividade dos marcos visuais e diminuir a complexidade computacional no processamento e correspondência dos dados com os modelos, elementos do ambiente são representados por modelos minimalistas, possibilitando o uso do ELViS em um grande número de aplicações onde o custo ou tempo de execução sejam fatores limitantes. ELViS foi implementado e testado utilizando dois estimadores baseados em Filtro de Kalman. Os resultados, obtidos com robôs reais e em simulações, indicam direções bastante promissoras.

Abstract

A key component of a mobile robot system is the ability to localize itself accurately, which involves estimating its pose with respect to some global representation of space. The general specification of a sensor-based localization approach starts with an initial estimate of the robot's pose and uses sensor data in conjunction with a map to produce a refined pose estimate that has an increased confidence about the true pose of the robot. One of the main difficulties is that sensor data is corrupted by measurement errors. These errors can arise from noise, quantization, digitalization artifacts, wheel slippage, and other such sources. Different sensors measure different physical properties, which are corrupted by different sources of measurement errors. The use of data from multiple sensors provides redundant and complementary information that can be processed to obtain a combined estimate aiming at an increase in the confidence of the final pose estimate. In this work we propose ELViS, a system that estimates the localization of a mobile robot equipped with odometers, a video camera and a frontal semi-ring of 8 sonar sensors, and that operates successfully in stationary and structured indoor environments. It is assumed that the robot navigates on flat surfaces and that straight lines can be identified in the environment image acquired by the camera. To increase selectivity of the landmarks and reduce computational complexity in data processing and matching to the map, environment features are represented using minimalist models in the map. This allows the use of ELViS in a large number of applications where tight budget or execution time constraints exist. ELViS has been implemented and tested using two estimators based on the Kalman Filter. The results, obtained with the real robots and in series of simulation runs, indicate promising directions.

Lista de Figuras

1.1	Comparação entre sonares e visão monocular.	4
2.1	Comparação entre estimadores com referencial fixo e flutuante . .	8
2.2	Diagrama de fluxo da estimativa do Filtro de Kalman.	14
3.1	Diagrama genérico do Estimador Recursivo Bayesiano especializado com os modelos do ELViS	22
3.2	O eixo Z sistema de coordenadas intrínseco à câmera (Z_C) está alinhado ao eixo Z do sistema de coordenadas de referência. . . .	25
3.3	Fluxo de obtenção de observação dos diversos sensores em função do tempo.	26
3.4	Fluxo de execução do estimador no tempo.	28
4.1	Robô Pioneer 2 - DXe	30
4.2	Vista inferior do robô	31
4.3	Deslocamento do robô sob o ponto de vista de um observador localizado na roda direita.	32
4.4	Curva representando a <i>pdf</i> da posição do robô.	33
4.5	Conjunto de partículas distribuídas segundo uma distribuição semelhante à <i>pdf</i> real da postura do robô.	34
5.1	Mapa de Profundidades denso obtido com um sonar rotatório e o contorno da sala.	39
5.2	Mapa de Profundidades denso com um sonar rotatório obtido a partir de simulação.	40
5.3	Gráfico <i>orientação</i> $\times$ <i>profundidade</i> da resposta do sonar para uma parede.	41
5.4	Resposta do sonar ao observar uma parede.	42
5.5	Oito sonares formam um semicírculo na região frontal do robô. . .	43

5.6	Mapa de Profundidades Esperso obtido com sonares estáticos. . .	44
5.7	Mapa de Profundidade Linear, obtido de observações de um sonar em diversas posições sobre uma trajetória retilínea.	45
5.8	Representação de observações do sonar em diferentes posições ao longo de um movimento em MRU.	49
5.9	Uma parede é representada pela sua distância à origem, r_{wall} , e sua inclinação, θ_{wall} em relação ao eixo X	57
6.1	Modelo de Câmera <i>Pinhole</i>	64
6.2	Corredor contendo elementos geradores de retas verticais	67
6.3	Representação de uma Reta Vertical e sua projeção.	68
6.4	Diagrama de blocos do modelo de observação para a visão monocular.	69
6.5	Exemplo da aplicação do Filtro de Sobel.	73
6.6	Resultado da etapa de pré-processamento.	74
6.7	Representação do Perfil Médio de Cor.	79
6.8	Exemplo de ambigüidade na detecção das projeções de retas verticais	92
7.1	Ambiente de testes durante uma simulação.	115
7.2	Mapa do ambiente de testes e trajetória real do robô.	117
7.3	Trajetória da estimativa da posição com odômetros.	118
7.4	Gráfico com o erro da estimativa do UKF usando apenas odômetros, com dados reais.	119
7.5	Trajetória da estimativa da posição com sonar e odômetros com dados reais, para EKF e UKF.	120
7.6	Gráfico com o erro da estimativa da posição com sonar e odômetros com dados reais, para EKF e UKF.	121
7.7	Região que gera observações enganosas ao sonar.	121
7.8	Trajetória da estimativa da posição com sonar e odômetros com dados reais, para EKF e UKF, ignorando os 4 sonares frontais.	122
7.9	Gráfico com o erro da estimativa da posição com sonar e odômetros com dados reais, para EKF e UKF, ignorando os 4 sonares frontais.	123

7.10	Trajetória da estimativa da posição com visão e odômetros com dados reais, para EKF e UKF.	124
7.11	Gráfico com o erro da estimativa da posição com visão e odômetros com dados reais, para EKF e UKF.	125
7.12	Gráfico mostrando o número de marcos observados, retas verticais e projeções extraídas por iteração.	126
7.13	Gráfico com o erro RMS da estimativa da posição com os 4 sonares laterais e odômetros com dados artificiais, para EKF e UKF. . . .	128
7.14	Gráfico com o erro RMS da estimativa da posição com visão e odômetros com dados artificiais, para EKF e UKF.	129
7.15	Gráfico com o erro RMS da estimativa da posição com os 4 sonares laterais, visão e odômetros com dados artificiais, para EKF e UKF.	130
7.16	Trajetória da estimativa da posição com visão, os 4 sonares laterais e odômetros com dados reais.	131
7.17	Gráfico com o erro da estimativa da posição com visão, os 4 sonares laterais e odômetros com dados reais para o UKF.	132
7.18	Gráfico com a coordenada X do erro da estimativa da posição com visão, os 4 sonares laterais e odômetros com dados reais para o UKF.	133
7.19	Gráfico com a coordenada Y do erro da estimativa da posição com visão, os 4 sonares laterais e odômetros com dados reais para o UKF.	134
7.20	Gráfico com o erro da estimativa da posição com visão, os 4 sonares laterais e odômetros com dados reais para o UKF na presença de erro de calibração.	135
7.21	Trajetória da estimativa da posição com visão, os 4 sonares laterais e odômetros com dados reais na presença de erro de calibração. . .	136
7.22	Curvas de erro da trajetória do estimador usando os 4 sonares laterais.	137
7.23	Curvas de erro da trajetória do estimador usando visão monocular.	138
7.24	Curvas de erro da trajetória do estimador usando visão monocular e os 4 sonares laterais.	139
I.1	Padrão usado na calibração da câmera.	158
I.2	Posicionamento do robô do interior da estrutura de calibração . .	158

I.3	Padrão para alinhamento da câmera sobre o robô	159
-----	--	-----

Lista de Tabelas

6.1	Parâmetros	106
7.1	Tempos de Execução	138
7.2	Observações coletadas por sensor	140

Lista de Abreviaturas

- A-PF** Filtro de Partículas Adaptativo (*Adaptive Particle Filter*)
- ARTPF** Filtro de Partículas em Tempo Real Adaptativo (*Adaptive Real Time Particle Filter*)
- EKF** Filtro de Kalman Extendido (*Extended Kalman Filter*)
- EKF-PF** Filtro de Partículas EKF (*Extended Particle Filter*)
- GPS** Sistema de Posicionamento Global (*Global Positioning System*)
- JPDA** Associação de Dados usando Probabilidade Conjunta (*Joint Probability Data Association*)
- KLD-PF** Filtro de Partículas KLD (*KLD Particle Filter*)
- ML-EKF** Localização Markoviana - Filtro de Kalman Extendido (*Markov Localization - Extended Kalman Filter*)
- MC** Média de Cores
- MCMC** Método de Monte Carlo com Cadeia de Markov (*Markov chain Monte Carlo*)
- MHL** Localização Multi-hipótese (*Multi-Hypothesis Localization*)
- MHT** Teste Multi-hipótese (*Multi-Hypothesis Test*)
- Mix-MCL** Filtro de Partículas com Mistura de Distribuições Auxiliares (*Mixture Monte Carlo Localization*)
- MRU** Movimento Retilíneo Uniforme
- pdf*** Função Densidade de Probabilidade (*Probability Density Function*)
- PDF*** Função Densidade de Probabilidade Acumulada (*Cumulative Distribution Function*)
- PF** Filtro de Partículas (*Particle Filter*)

PMC Perfil Médio de Cor

RCD Região de Profundidade Constante (*Region of Constant Depth*)

RMS Erro Médio Quadrático (*Root Mean Square*)

RTPF Filtro de Partículas em Tempo Real (*Real Time Particle Filter*)

SCAAT Uma restrição por vez (*Single-constraint-at-a-time*)

SfM Estrutura a partir do Movimento (*Structure from Motion*)

SIR Amostragem por Imortância com Reamostragem (*Sample Importance-Resample*)

SLAM Localização e Mapeamento Simultâneos (*Simultaneous Localization And Mapping*)

SR-EKF Filtro de Kalman Extendido Raiz Quadrada (*Square root Extended Kalman Filter*)

SR-UKF Filtro de Kalman Unscented Raiz Quadrada (*Square root Unscented Kalman Filter*)

SRL Reinicialização Sensorial da Localização (*Sensorial Reinitialization Localization*)

UKF Filtro de Kalman Unscented (*Unscented Kalman Filter*)

UKF-PF Filtro de Partículas Unscented (*Unscented Particle Filter*)

UT Transformada Unscented (*Unscented Transform*)

Convenções e Lista de Símbolos

Na notação das fórmulas, as seguintes convenções foram utilizadas:

- letras maiúsculas em negrito são matrizes, exemplo: $\mathbf{P}$;
- $\mathbf{A}^T$ é a matriz transposta de $\mathbf{A}$;
- os elementos que compõem uma matriz são exibidos da seguinte forma:
 $\begin{pmatrix} x1 & x2 \\ x3 & x4 \end{pmatrix}$, sendo $x1$, $x2$, $x3$ e $x4$ os elementos da matriz;
- os elementos que compõem uma matriz com apenas uma linha podem ser exibidos da seguinte forma: $[x1, x2, x3]$, sendo $x1$, $x2$ e $x3$ os elementos da matriz;
- letras minúsculas em negrito são vetores (matrizes com apenas uma coluna), exemplo: $\mathbf{x}$;
- símbolos com uma barra em cima representam estimativas, exemplo: $\bar{\mathbf{x}}$ é uma estimativa do vetor aleatório $\mathbf{x}$;
- símbolos com um acento circunflexo em cima representam estimativas ótimas, exemplo: $\hat{\mathbf{x}}$ é uma estimativa ótima do vetor aleatório $\mathbf{x}$;
- índice entre parênteses é um índice temporal discreto, exemplo: $\mathbf{x}(\mathbf{i})$, $\mathbf{X}(\mathbf{i})$;
- índice superescrito em vetor ou escalar indica elemento de um conjunto; em uma matriz pode indicar coluna ou linha, exemplo: $\mathbf{X}^{(\mathbf{i})}$, $\mathbf{Y}^{(\mathbf{i})}(n)$, $\mathbf{p}^{(\mathbf{i})}$;
- $E_x[\cdot]$ é o operador expectativa em função da variável x ;
- variáveis, valores ou índices separados por $|$ representam uma correlação temporal de eventos, indicando que o elemento à direita de $|$ precede o elemento à esquerda. Usado para indicar ou referenciar uma probabilidade condicional. Por exemplo, $p(x|y)$ representa a probabilidade de ocorrer o evento x dado a ocorrência do evento y . Já em $\mathbf{x}(\mathbf{n}|\mathbf{n} - 1)$ o índice $n|n - 1$ indica que a distribuição de probabilidades que a variável aleatória $\mathbf{x}$ segue é relativa a um evento no instante n , considerando determinadas condições do instante $n - 1$;

Três sistemas de coordenadas são usados durante o trabalho, dois representando espaços tridimensionais e um representando um espaço bidimensional.

- (X, Y, Z) define o sistema de coordenadas associado ao ambiente em que o robô se desloca, fixo num ponto arbitrário. O plano definido pelos eixos X e Y , plano XY , coincide com o plano de deslocamento do robô e é chamado de plano horizontal. O eixo Z define a orientação vertical;
- (X_C, Y_C, Z_C) define um sistema de coordenadas associado à câmera de vídeo acoplada ao robô. O centro C do sistema de coordenadas corresponde ao centro da câmera, o eixo X_C é perpendicular ao plano de projeção da câmera, que por sua vez é paralelo ao plano definido pelos eixos Y_C e Z_C , o plano $Y_C Z_C$.
- (U, V) define um sistema de coordenadas associado ao plano de projeção da câmera. Sua origem se localiza no canto esquerdo inferior do plano de projeção, sendo U paralelo a Y_C e V paralelo a Z_C .

Os seguintes símbolos serão utilizados:

símbolo	descrição
x, y, θ	parâmetros que indicam a posição (x, y) e orientação (θ) sobre o plano de deslocamento do robô em relação a um referencial absoluto
$\mathbf{Y}(\mathbf{n})$	matriz contendo observações sensoriais coletadas a partir do instante inicial até o instante n
$\mathbf{corr}_y(\mathbf{i} \mathbf{i} - 1)$	vetor gerado pelo estimador a partir das observações para corrigir a estimativa prevista da postura, antes de passagem pelo bloco de ganho do estimador
$\mathbf{corr}_x(\mathbf{i} \mathbf{i} - 1)$	vetor gerado pelo estimador a partir das observações para corrigir a estimativa prevista da postura, após passagem pelo bloco de ganho do estimador
D_L	distância percorrida pela roda esquerda do robô entre dois instantes de tempo
D_R	distância percorrida pela roda direita do robô entre dois instantes de tempo
d_m	distância percorrida pelo centro de massa do robô entre dois instantes de tempo

continua. . .

símbolo	descrição
$\delta\theta_m$	variação da orientação do robô entre dois instantes de tempo
N	representação da distribuição probabilística normal, ou gaussiana
$f(\cdot)$	função não-linear que representa o modelo de dinâmica do robô
$\mathbf{F}$	matriz que representa o modelo de dinâmica do robô linearizado
d_{err}	variável aleatória que segue distribuição gaussiana e modela a incerteza associada à d_m
$\delta\theta_{err}$	variável aleatória que segue distribuição gaussiana e modela a incerteza associada à $\delta\theta_m$
$\sigma_{d_m(n)}^2$	variância de d_{err}
$\sigma_{\delta\theta_m(n)}^2$	variância de $\delta\theta_{err}$
$\sigma_{d_m}^{d_m}$	coeficiente que determina a participação do parâmetro d_m^2 no valor de $\sigma_{d_m(n)}^2$
$\sigma_{d_m}^{\delta\theta_m}$	coeficiente que determina a participação do parâmetro $\delta\theta_m^2$ no valor de $\sigma_{d_m(n)}^2$
$\sigma_{\delta\theta_m}^{d_m}$	coeficiente que determina a participação do parâmetro d_m^2 no valor de $\sigma_{\delta\theta_m(n)}^2$
$\sigma_{\delta\theta_m}^{\delta\theta_m}$	coeficiente que determina a participação do parâmetro $\delta\theta_m^2$ no valor de $\sigma_{\delta\theta_m(n)}^2$
$\mathbf{obs}(\mathbf{n})$	vetor que agrupa as observações dos odômetros, $d_m(n)$ e $\delta\theta_m(n)$
$\mathbf{F}_{obs}$	matriz que representa a parte do modelo de dinâmica do robô linearizado que corresponde a contribuição das observações dos odômetros
$\mathbf{P}_x$	matriz de covariância da estimativa da postura do robô
$\mathbf{Q}$	matriz de covariância aditiva indicando o aumento de incerteza na estimativa da postura do robô relativo às observações dos odômetros
$\mathbf{Q}_{obs}$	matriz de covariância das observações dos odômetros
β	ângulo interno do arco de circunferência que representa o alcance de um sonar

continua...

símbolo	descrição
k	número de observações do sonar que estão sendo usadas na etapa de validação
k_{min}	número mínimo de observações do sonar necessárias para validar um elemento sendo observado
α	ângulo formado na intersecção entre elemento observado pelo sonar e a trajetória do robô
$\gamma_{j,j-1}$	relação adotada na validação das observações dos sonares que, em caso da validação confirmar o elemento observado, representa o seno de α
χ^2	representa a própria distribuição probabilística χ^2
s_γ^2	variância do conjunto formado pelos $k - 1$ $\gamma_{j,j-1}$ obtidos durante a validação
$\sigma_{obsmedia}^2$	variância da <i>pdf</i> gaussiana que a variável aleatória γ deveria obedecer caso a hipótese que esteja se observando um elemento válido seja verdadeira
$\omega_{j,j-1}$	vetor composto pelas variáveis $\frac{d_{k,1}}{k-1}$ e $y_{j,j-1}$, usadas para obter a variável aleatória $\gamma_{j,j-1}$
$\Sigma\omega_{j,j-1}$	matriz de covariância do vetor $\omega_{j,j-1}$
$\mathbf{F}_\gamma$	linearização da função que recebe como entrada $\omega_{j,j-1}$ e retorna $\gamma_{j,j-1}$
r_{wall}, θ_{wall}	parâmetros que descrevem um elemento mapeado usado pelo modelo de observação dos sonares
$h(.)$	função não-linear que representa o modelo de observação. Recebe como entrada a postura do robô e retorna a observação esperada do sensor para a postura entrada
$\mathbf{H}$	linearização da função $h(.)$
u, v	coordenadas de um ponto no plano de projeção da câmera de vídeo
u_0, v_0	coordenadas que indicam o centro do plano de projeção da câmera de vídeo
f_u, f_v	parâmetros do modelo de câmera adotado representando a distância focal na direção das coordenadas u e v , respectivamente

continua...

símbolo	descrição
m	parâmetro que caracteriza uma projeção de reta vertical extraída de um quadro de vídeo. Indica o valor da tangente entre a reta e a direção vertical da imagem
$\mathbf{R}$	matriz de rotação usada na equação de transformação de coordenadas
$\mathbf{C}$	matriz de translação usada na equação de transformação de coordenadas
C_{ij}	correlação cruzada entre os <i>PMCs</i> das projeções das retas verticais i e j
$F_{shift}^{(k)}$	função auxiliar usada para determinar o conjunto de associações entre projeções de retas verticais e elementos mapeados. Essa função concentra as características espaciais das associações
Sc_k	função usada para determinar o conjunto de associações entre projeções de retas verticais e elementos mapeados. Essa função combina características espaciais e fotométricas das associações
u_{shift}	distância entre a projeção esperada de um elemento mapeado e a projeção do quadro atual associada à mesma
$lim_{start},$ lim_{end}	limiares dos valores dos pixels usados durante o afinamento das bordas ressaltadas na convolução do quadro com o filtro apresentado na equação 6.4
Lh	valor adotado para o número de pixels que devem ser observados antes de interromper a geração de uma sequência a partir da imagem pré-processada
$range_{deriv}$	parâmetro que define o número de pixels de uma cadeia que são considerados para determinar a inclinação do segmento em torno de cada pixel com objetivo de determinar se a cadeia pode ser considerada ou não um segmento de reta, considerando que todos os pixels de um segmento precisam apresentar inclinação similar

continua...

símbolo	descrição
lim_{deriv}	parâmetro que define o quanto a inclinação local dos pixels de uma cadeia podem variar para que a cadeia possa ser considerada um segmento de reta
u_{prox}, m_{prox}	limiares que indicam que, se as diferenças entre os valores dos parâmetros u e m , respectivamente, de dois segmentos ficarem abaixo de u_{prox} e m_{prox} , então os segmentos fazem parte da mesma projeção de reta vertical
lim_{numpix}	determina o número mínimo de pixels necessários para que um conjunto de seqüências de pixels que representa uma reta vertical seja aceito
m_{min}, m_{max}	delimitam a faixa de inclinações que um conjunto de cadeias de pixels que representa uma reta vertical pode apresentar para ser considerado uma reta vertical –
$n_{lateral}$	indica o comprimento dos vetores compõem o PMC . O comprimento de cada vetor é dado por $1 + 2 \cdot n_{lateral}$
α_{prox}	determina o espalhamento dos <i>pontos sigma</i> na transformada Unscented usada para determinar a janela de busca espacial. Quanto menor α , mais próximos os <i>pontos sigma</i> ficarão e menor o risco de inconsistências serem geradas
x_{min}, x_{max}	valores do afastamento máximo e mínimo que um objeto pode estar da câmera para ser considerado
C_{janela}	determina o tamanho da janela de busca obtida quando mais de uma projeção estiver associada à reta vertical
C_{min}	limiar de corte para rejeitar associações entre projeções do quadro atual e do quadro anterior usando a medida de similaridade C_{ij} baseada nos $PMCs$
m_{projs}	número máximo de projeções usadas para obter o PMC das retas verticais

continua...

símbolo	descrição
u_{extra}	indica o quanto se permite que a projeção esperada de um marco potencialmente observado fique distante dos limites do plano de projeção
$du_{shift,1},$ $du_{shift,2},$ $du_{shift,3},$ $du_{shift,4}$	indicam as diferenças máximas entre os u_{shift} da primeira e da última associações de uma sequência de associações ordenada por u_{shift} para que a mesma seja aceita para o próximo passo do algoritmo de associação entre projeções extraídas do quadro de vídeo e elementos mapeados
$uShift_{mult}$	coeficiente usado na determinação do conjunto associações entre projeções extraídas do quadro de vídeo e elementos mapeados
$uShift_{att,lim}$	coeficiente usado na determinação do conjunto associações entre projeções extraídas do quadro de vídeo e elementos mapeados
$N_{proj,assoc}$	indica quantas vezes um mesmo elemento do ambiente deve ser identificado em quadros consecutivos através da associação de projeções entre quadros para poder ser considerado como sendo observado
$N_{marco,assoc}$	indica quantas vezes consecutivas um mesmo elemento mapeado deve ser associado a um mesmo elemento sendo observado para que a associação possa ser usada para corrigir a postura do robô

Conteúdo

1	Introdução	1
1.1	Objetivo	3
1.2	Justificativa	3
1.3	Contribuições	5
1.4	Organização do trabalho	5
2	Conceitos Fundamentais e Revisão da Literatura	6
2.1	Sensores	6
2.2	Localização	7
2.3	Combinação sensorial	9
2.3.1	Combinação entre sensor de profundidade e visão	10
2.4	Estimadores	12
2.5	Estimadores Bayesianos	12
2.6	Discussão	16
3	ELViS: Estimador de Localização com Visão e Sonares	17
3.1	Premissas Adotadas	17
3.2	Estimador	18
3.3	Tratamento Sensorial	19
3.4	Representação do Ambiente	20
3.5	Estrutura do Estimador	21
3.5.1	Modelo de Dinâmica	22
3.5.2	Modelo de Observação dos Sonares	23
3.5.3	Modelo de Observação da Visão Monocular	24

3.5.4	Processamento Assíncrono	26
4	Modelo de Dinâmica	29
4.1	Incerteza do deslocamento	32
4.2	Incerteza do modelo de dinâmica	35
4.3	Determinação dos parâmetros usados no modelo de dinâmica . . .	37
4.4	Resumo do Capítulo	37
5	Modelo de Observação: Sonares	38
5.1	RCDs	42
5.2	Adaptação do método RCD	43
5.3	Restrições adotadas	44
5.4	Descrição do modelo de observação	46
5.4.1	Operação	47
5.4.2	Validação das observações	48
5.4.3	Associação das observações com o mapa	54
5.4.4	Obtenção das observações esperadas para a estimativa da postura	56
5.5	Determinação dos parâmetros usados no modelo de observação do sonar	57
5.6	Resumo do Capítulo	60
6	Modelo de Observação: Visão Monocular	61
6.1	Modelo de Câmera	62
6.2	Objeto Observado	65
6.3	Arquitetura	67
6.4	Extração das projeções a partir dos quadros	70
6.4.1	Formato das projeções das retas verticais	70
6.4.2	Organização da etapa de extração	71
6.4.3	Pré-processamento	72

6.4.4	Detecção das projeções	75
6.4.5	Representação das projeções	77
6.4.6	Determinação do Perfil Médio de Cor	78
6.5	Associação de projeções às retas verticais	78
6.5.1	Restrição por janela de busca espacial	80
6.5.2	Restrição por Perfil Médio de Cor	89
6.5.3	Determinação do conjunto de associações válidas	90
6.5.4	Considerações sobre a medida de similaridade	91
6.5.5	Algoritmo para determinar as associações válidas entre re- tas verticais e projeções	92
6.6	Modelo de Marcos	94
6.6.1	Associação entre marcos e retas verticais	95
6.6.2	Formação e comparação entre os conjuntos de projeção por disposição espacial	96
6.6.3	Descrição do algoritmo de associação	99
6.6.4	Obtenção das observações esperadas a partir dos marcos .	103
6.6.5	Jacobiano do Modelo de Marcos	103
6.7	Implementação - distância entre quadros processados	104
6.8	Valores dos parâmetros usados no Modelo de Observação de Visão Monocular	106
6.9	Resumo do Capítulo	110
7	Implementação e Resultados Experimentais	112
7.1	Implementação do ambiente de testes	113
7.2	Experimentos	116
7.2.1	Dados Reais: Sonar	120
7.2.2	Dados Reais: Visão	123
7.2.3	Comparação do UKF e EKF	127
7.2.4	Dados Reais: com uso de Visão e Sonares	131

7.2.5	Sensibilidade à posição inicial	135
7.2.6	Tempo de Execução	137
7.2.7	Comparação com Trabalhos Similares	140
7.3	Análise Geral	141
8	Conclusão e Trabalhos Futuros	143
8.1	Discussões	144
8.1.1	Estimadores	144
8.1.2	Sonares	145
8.1.3	Visão monocular	145
8.2	Trabalhos Futuros	146
8.2.1	Estimadores	147
8.2.2	Sonares	147
8.2.3	Visão monocular	148
	Apêndices	157
I	Calibração do Sistema	157
I.1	Determinação dos parâmetros do modelo de câmera	157
I.2	Determinação da posição e orientação da câmera sobre o robô . .	158
II	Estimadores Bayesianos	160
II.1	Introdução aos Estimadores	160
II.2	Estimadores Bayesianos	161
II.3	Estimadores Bayesianos Recursivos	161
II.4	Estimadores Lineares Gaussianos	163
II.5	Estimadores Não Lineares Gaussianos	166
II.5.1	Filtro de Kalman Estendido	167
II.5.2	Filtro de Kalman Unscented	169

II.6	Estimadores Não Lineares	175
II.6.1	Localização Markoviana	175
II.6.2	Amostragem por Importância com Reamostragem	176
II.7	Outros Estimadores	183

1 Introdução

Segundo Dudek e Jenkin (2000), a robótica móvel é uma área de estudos que se preocupa com o controle de veículos autônomos e semi-autônomos. Essa área combina conhecimento de diversas outras áreas, como inteligência artificial, mecânica, teoria de controle, entre outras, objetivando a navegação do robô com intuito de atingir alguma meta.

Dentre as diversas aplicações da robótica móvel, pode-se citar alguns exemplos:

- robôs para servir em hospitais, transportando suprimentos médicos, refeições ou medicamentos;
- mini-submarinos autônomos para exploração de naufrágios ou para verificações em plataformas de petróleo;
- dispositivos que permitam o acompanhamento médico de pacientes remotamente;
- robôs para exploração espacial, tanto no solo de outros planetas como em suas órbitas;
- robôs para entretenimento, como o Aibo, o cachorro-robô da Sony.

Para navegar com sucesso, é necessário determinar a postura do robô em relação aos elementos com os quais precisa interagir. Esse problema é conhecido como determinação da Localização, fundamental no estudo dos robôs móveis, e é considerado como um dos mais importantes na área, conforme Cox e Wilfong (1990). A determinação da Localização implica na determinação de um conjunto pré-determinado de parâmetros que define a postura do robô. A postura de um robô em um ambiente como o descrito pode ser determinada pelas suas coordenadas x e y na superfície de deslocamento em relação a um sistema cartesiano referencial (posição do robô no plano), e sua orientação θ (para onde a frente do robô aponta) em relação a esse sistema. Por exemplo, para um robô-avião, a

postura pode ser representada pelas três coordenadas do seu centro de massa, e suas inclinações em relação aos três eixos de coordenadas do ambiente.

Para determinar a postura do robô à medida que o mesmo se movimenta, é necessário obter alguma informação a respeito do deslocamento realizado. Essa informação pode ser: *i*) os acionamentos que provocam o deslocamento (como a aceleração sendo imposta ao robô), *ii*) as observações do deslocamento (o quanto o robô se moveu entre dois pontos, por exemplo), ou *iii*) as observações da própria postura do robô (a latitude e longitude em que o robô se encontra, por exemplo).

Seja qual for o tipo de informação disponível, um algoritmo de determinação da localização deve levar em conta o fato que essas informações estão corrompidas por sinais indesejados, chamados de ruídos ou incerteza da informação, e é impossível separar esses sinais das informações desejadas observando apenas as próprias informações.

O problema da determinação da localização pode ser dividido em duas categorias. A primeira, conhecida por Localização Global, consta em determinar a localização sem conhecer previamente a postura inicial. A segunda etapa é conhecida por Rastreamento ou Localização Local. Nessa categoria, existe uma estimativa inicial da postura, que deve ser atualizada conforme o robô se desloca.

O rastreamento é uma tarefa muito requisitada durante a navegação, pois é através da postura atualizada do robô que se pode tomar uma decisão do que fazer em seguida. Quanto menor o tempo gasto para determinar a postura, mais tempo pode ser destinado à tomada de decisão pelo algoritmo de navegação.

O foco deste trabalho está na Localização Local (aqui chamada apenas por Localização) de robôs móveis que se deslocam por superfícies planas em ambientes internos. Esse tipo de ambiente é caracterizado pela existência de superfícies planas, paredes verticais, portas, janelas, móveis, adornos e outros objetos, formando um ambiente cuja característica principal é a abundância de elementos geométricos retangulares, dispostos paralelamente ou ortogonalmente ao plano de deslocamento do robô. Como exemplos desse tipo de ambiente podem-se citar escritórios, escolas, apartamentos, hospitais e centros comerciais. É esperado que a maioria dos elementos presentes nesses ambientes não se movam. Exceção é feita para um número reduzido de pessoas circulando. O número de pessoas deve ser pequeno o suficiente de modo que elas não bloqueiem por muito tempo a observação do ambiente pelo robô.

1.1 Objetivo

Neste trabalho é proposto um estimador para rastrear a postura de um robô que se movimenta por um ambiente interno conhecido, fazendo uso de sonares e de uma câmera de vídeo rigidamente fixados a ele, de odômetros que monitoram o deslocamento do conjunto (robô e seus sensores) e que seja capaz de operar em tempo real fazendo uso de um processador de capacidade moderada (por ex., um Pentium 4 1.5GHz).

Reconhecendo que acurácia e baixo custo (financeiro e computacional) geralmente são requisitos conflitantes, procura-se atingir um equilíbrio entre a acurácia e o tempo de execução, fixando o processador.

1.2 Justificativa

Teoricamente, a qualidade da estimativa da postura que se pode obter é proporcional à qualidade da informação disponível sobre a postura. Quanto mais sensores observarem a postura ou o deslocamento do robô e quanto maior for a acurácia desses sensores, maior a probabilidade de se obter uma estimativa mais acurada da postura.

Além de potencialmente aumentar a acurácia da estimativa da postura, o uso de mais de uma fonte sensorial pode fornecer maior robustez contra erros grosseiros gerados pelas próprias fontes sensoriais, que poderiam conduzir a estimativas muito distantes da localização real do robô. Esses erros grosseiros são originados tanto de problemas de funcionamento dos sensores como também de má interpretação das leituras, que podem ocorrer por situações não previstas ou não modeladas. Com duas ou mais fontes sensoriais observando a postura, pode-se determinar com mais segurança quando uma observação que resulta em uma postura muito diferente da esperada é um erro grosseiro ou não.

A escolha pelo uso de sonares e visão monocular (cujo sensor é a câmera de vídeo) como fontes de informação para estimar a postura pode ser justificada pelo fato de que esses sensores apresentam características complementares. A primeira característica complementar diz respeito ao alcance dos sensores. Enquanto os sonares têm alcance limitado em alguns metros (3 metros para os sonares usados neste trabalho), a visão monocular atinge alcances bem maiores, da ordem de algumas dezenas de metros, podendo aumentar conforme a regulagem da distância focal da câmera.

A segunda característica complementar diz respeito à acurácia que se pode obter a partir de cada tipo de sensor. Os sonares são capazes de observar com precisão e acurácia a distância do robô a elementos do ambiente, porém têm dificuldade em determinar a direção em que eles se encontram; já a visão monocular consegue determinar a direção com acurácia, mas não a profundidade do elemento, conforme ilustra a figura 1.1.

Essas diferenças no modo em que observam o ambiente geram efeitos diferentes na estimativa de cada parâmetro da postura (x , y , θ). A visão monocular consegue fornecer melhores estimativas para o parâmetro θ , enquanto que os sonares oferecem melhores estimativas dos parâmetros x e y .

Figura 1.1: Uma leitura do sonar retorna a distância do objeto mais próximo dentro de um cone de atuação (esquerda), enquanto que um quadro capturado pela câmera pode fornecer a direção de um objeto, mas não sua profundidade (direita).

A escolha do uso de sonares em vez de sensores a laser se deve ao custo. Apesar de fornecer leituras muito mais acuradas, o sensor de profundidade baseado em laser ainda é um sensor caro, enquanto que os sonares estão presentes em diversas plataformas robóticas e são mais baratos do que o equivalente laser.

O custo também justifica o uso da visão monocular no lugar de visão estéreo. Apesar da informação de profundidade poder ser extraída com maior facilidade da visão estéreo e o custo de câmeras de qualidade regular não tornar proibitivo o uso de visão estéreo, o custo adicional em hardware (para aquisição) e em recursos computacionais (para processamento) para capturar e tratar quadros extraídos de duas câmeras torna mais atraente a adoção da visão monocular em relação à

visão estéreo.

1.3 Contribuições

As principais contribuições do trabalho são:

- Um novo método para determinar a postura de um robô móvel usando seqüências temporais de observações de sonares rigidamente acoplados ao robô, com o auxílio de um mapa do ambiente.
- Um novo método para determinar a postura de um robô móvel através de visão monocular, baseado na observação de elementos do ambiente que produzem projeções na forma de segmentos de reta alinhados na direção vertical, com o auxílio de um mapa do ambiente.
- Um novo método para resolver o problema da correspondência entre projeções de elementos observados em dois quadros de vídeo não seqüenciais. O método se limita a considerar elementos do ambiente que produzem projeções na forma de segmentos de reta vertical.
- Combinação dos métodos citados acima para tratar observações de sonares, visão monocular e odômetros utilizando um Filtro de Kalman com o objetivo de obter um estimador de localização mais confiável do que se fosse usado qualquer outra combinação de apenas dois dos três conjuntos de sensores utilizados.

1.4 Organização do trabalho

O restante deste trabalho está dividido do seguinte modo: O capítulo 2 apresenta uma revisão de conceitos fundamentais e do que já foi feito na área. O capítulo 3 descreve em linhas gerais o estimador proposto. O capítulo 4 descreve o modelo de dinâmica que modela o deslocamento do robô. O capítulo 5 descreve o modelo adotado para a resposta dos sonares, enquanto o capítulo 6 apresenta o modelo adotado no trabalho para lidar com a visão monocular. O capítulo 7 descreve os detalhes de implementação do sistema proposto e apresenta os experimentos realizados. A conclusão do trabalho é apresentado no capítulo 8.

2 Conceitos Fundamentais e Revisão da Literatura

Nesse capítulo é apresentada uma revisão de conceitos importantes para o melhor entendimento deste trabalho, juntamente com uma breve revisão da literatura das áreas abordadas no mesmo.

2.1 Sensores

Sensores são instrumentos que observam alguma característica ou condição do ambiente à sua volta e transformam essa observação em uma outra informação que possa ser interpretada, seja por computadores ou por seres humanos. Neste trabalho os sensores são classificados em dois grupos distintos: os sensores apropriados para observar deslocamento e os sensores apropriados para observar postura. Os sensores dos dois grupos podem ser usados para determinar a postura, porém com os sensores do primeiro grupo a postura é determinada através da avaliação do movimento do robô entre dois pontos, enquanto que nos do segundo grupo ela é determinada diretamente, sem levar em conta o deslocamento do robô.

Entre os sensores para observar deslocamento pode-se citar sensores de odometria (como contadores de giros fixados nos eixos das rodas) e sensores inerciais (que detectam acelerações). Os sensores citados são também classificados como sensores proprioceptivos, ou seja, sensores que observam o estado interno do robô. Os sensores próprios para observar deslocamento não se restringem apenas a sensores proprioceptivos. Sensores exteroceptivo (sensores que observam o ambiente, como sonares, visão, audição) também podem ser usados para observar deslocamento, dependendo de como suas observações são usadas.

A obtenção da postura do robô com esse tipo de sensor é realizada integrando o deslocamento do robô desde a posição inicial até a atual. A integração pode ser realizada por partes, à medida que observações dos sensores se tornam disponíveis ou quando é necessário determinar a postura. Assim, toda vez que a postura é

calculada, parte-se do último valor estimado, e a ele se integra o deslocamento decorrido desde então.

Os sensores para observar postura são sempre sensores exteroceptivos, como radar, laser ou sonar (tanto montados sobre plataformas rotativas como construídos na forma de vetores), GPS e câmeras de vídeo. A obtenção da postura é realizada através da determinação da posição relativa entre o robô e algum elemento de posição conhecida detectado pelo sensor.

Alguns sensores podem ser usados tanto como sensores para observar o deslocamento como para observar a postura. Neste trabalho, por exemplo, a visão monocular é usada para observar postura ou deslocamento, dependendo das informações disponíveis.

Em ambos os tipos de sensores é necessário processar a observação do sensor a partir da postura do robô. Esse processamento é contido num modelo de observação. Uma inovação no estudo da Localização que envolva a parte sensorial pode significar tanto o uso de um novo sensor como de um novo modelo de observação, que lide com algum sensor já usado anteriormente.

Outro fator importante em relação aos sensores é a acurácia da observação que ele é capaz de realizar. Todo sensor, ao realizar uma observação, sempre introduz algum ruído indesejado, muito difícil de ser separado do sinal desejado. Esse ruído pode resultar de inúmeras fontes, como perda de precisão em conversões analógico-digitais, componentes com leves desvios de especificação, ruído térmico, etc. Mais importante do que determinar a incerteza da observação é determinar como essa incerteza é propagada para a postura obtida. Isso também é função do modelo de observação.

2.2 Localização

O problema da Localização vem sendo muito estudado ao longo dos anos e diversas soluções já foram propostas. O ponto em comum de todas as abordagens analisadas é o uso de um ou mais sensores para determinar a postura do robô.

A solução mais simples consiste em manter a postura do robô através de observações do deslocamento (usando algum sensor de deslocamento). Esse tipo de determinação da localização local é conhecida na literatura como *dead-reckoning* e tem como grande vantagem a simplicidade e baixo custo computacional. A maior desvantagem do método está em depender de todas as observações realizadas até um dado instante para obter a estimativa da postura. Uma nova postura

obtida é corrompida tanto pelo ruído da observação mais recente obtida como pelo ruído acumulado das observações passadas, e a cada iteração a incerteza da estimativa da postura aumenta. Conforme o robô se desloca, a estimativa tende a divergir da postura real. Em alguns casos, em que os sensores apresentam erros pequenos e os deslocamentos são restritos, esse tipo de localização é aceitável. Foxlin (2005) apresenta uma solução para rastrear a posição de bombeiros em ambientes de alto risco através de sensores inerciais em seus sapatos, utilizando dead-reckoning.

Nas situações em que o erro acumulado não é aceitável, é necessário um modelo de observação que seja capaz de fornecer estimativas cujo erro não tenda a aumentar com o deslocamento. Para tal, o sensor necessita realizar observações tendo como base comparativa um referencial cuja postura seja conhecida (ou cuja estimativa seja conhecida a menos de um erro limitado superiormente). Esse é o caso dos sensores de postura. Essa diferença entre os tipos de sensores é melhor ilustrada na figura 2.1.

Figura 2.1: A estimativa da postura do robô que usa como referência a sua postura na iteração anterior tende a divergir (esquerda). Já a estimativa da postura do robô que usa um referencial fixo não tende a divergir (direita).

Os modelos de observação de sensores de postura necessitam conhecer um referencial fixo no ambiente cuja posição seja conhecida absolutamente (a menos de um dado erro) em relação ao sistema de coordenadas considerado. Isso implica em algum conhecimento prévio do ambiente por onde o robô se desloca. O modelo de observação de um sensor de profundidade, por exemplo, ao observar a distância do robô a um objeto, precisa conhecer a posição do objeto no ambiente para que seja possível estimar a posição do robô no ambiente. Um GPS funciona da mesma

forma. A posição do GPS é determinada graças ao conhecimento das posições dos satélites em relação à Terra.

Na literatura existem diversas formas de se incorporar o conhecimento do ambiente nos algoritmos de localização. Essas formas podem ser classificadas como: mapas de ocupação ou mapas de grades, mapas geométricos e mapas topológicos ou mapas de marcos.

Os mapas de grade discretizam o ambiente em células pequenas, e cada célula armazena a informação se aquela região do espaço está ocupada ou não. Mapas de grades permitem uma descrição com grande acurácia do ambiente sem impor restrições sobre a forma do mesmo, porém seu custo computacional em memória é grande, e aumenta conforme o tamanho do ambiente e a resolução desejada aumentam. Baltzakis, Argyros e Trahanias (2003) usam mapas formados por grades, também conhecidos como grades de ocupação.

Os mapas geométricos tentam compactar a informação dos mapas de grades representando construções do ambiente por modelos geométricos, como segmentos de retas, circunferências e elipses, por exemplo. Leonard e Durrant-Whyte (1991), Tardós (2002), Borges (2005), Bilgic e Turksen (1995), Chung e Kim (2002), Lu e Milios (1997) usam mapas geométricos para descrever o ambiente.

Os mapas topológicos não representam absolutamente o ambiente, como as abordagens anteriores. Um mapa topológico baseia-se em identificar certas regiões e objetos no ambiente, conhecidos por marcos, e determinar as relações entre essas regiões. Um mapa topológico pode ser visto como um grafo, onde os nós são os marcos e os arcos, as relações entre dois marcos. Os marcos devem ser facilmente distinguíveis do local onde estão. Fukuda (1995), Moon, Miura e Shirai (1999) usam marcos naturais presentes no ambiente, enquanto Briggs (2000), Kleeman (1992) e Lin e Tummala (1997) fazem uso de marcos adicionados artificialmente. Borenstein (1997) descreve métodos que utilizam tanto marcos naturais como marcos artificiais.

2.3 Combinação sensorial

Não é novidade que a combinação das observações de mais de um sensor pode produzir localizadores de melhor qualidade e soluções combinando mais de um sensor vêm sendo apresentadas há tempos.

A combinação mais comum encontrada na literatura é a de um sensor de deslocamento com um sensor de postura. Essa é uma combinação vantajosa devido

à complementaridade dos dois tipos de sensores: enquanto os sensores de deslocamento costumam fornecer estimativas precisas para deslocamentos pequenos, a baixo custo computacional e com uma taxa de amostragem alta, os sensores de postura costumam apresentar maior custo computacional e menor taxa de amostragem, porém conseguem manter a incerteza da estimativa da postura limitada superiormente. Trabalhos que adotam essa combinação são encontrados na literatura desde a década de 80. Davison e Murray (2002) usam visão ativa combinada com odometria, Leonard e Durrant-Whyte (1991) e Tardós (2002) combinam um sonar montado em plataforma rotatória (sonar rotatório) com odometria. Frin-trop, Jensfelt e Christensen (2006) usam visão monocular para extrair marcos do ambiente e corrigir a postura obtida através de odômetros. Já Kleeman (1992) combina odometria com uma rede de marcos ativos (um marco ativo é um marco que emite algum tipo de sinal, captado pelo robô).

A próxima evolução está em se combinar mais de um sensor de postura para refinar a precisão da estimativa. Os sensores de postura em geral têm pontos fortes e pontos fracos. Combinar dois ou mais deles é um método para tentar minimizar seus pontos fracos. Outra vantagem na combinação de dois ou mais sensores de postura é diminuir a chance de que uma falha em um sensor ou modelo de observação leve o estimador a gerar estimativas muito longe da realidade. Em geral essa abordagem é mais comum em problemas de mapeamento (AKBARALLY; KLEEMAN, 1996), porém alguns trabalhos que lidam com localização também a usam. Borges (2005), por exemplo, combina um laser rotatório, visão monocular e odometria para localizar o robô, enquanto Kortenkamp e Weymouth (1994) combinam sonar rotatório e visão monocular para obter um mapa e depois usá-lo para localizar o robô. Para determinar a localização de pessoas em diversos tipo de ambiente, Bertozzi (2006) combina dois conjuntos de visão estéreo: um operando na faixa da luz visível e outro na faixa do infra-vermelho.

2.3.1 Combinação entre sensor de profundidade e visão

Como já foi dito para sonares e visão monocular, a combinação mais genérica entre sensores de profundidade e visão apresenta vantagens similares devido à complementaridade apresentada. Nem toda combinação de sensor de profundidade e visão apresenta as mesmas características, mas em geral pode-se listar as seguintes características complementares:

- Sensores de profundidade (em especial os sonares e infravermelhos) têm seu alcance limitado (em alguns metros para sonares, em dezenas de centímetros

para infravermelhos), enquanto sensores de visão conseguem trabalhar com alcances bem maiores, da ordem de algumas dezenas de metros, podendo aumentar conforme a regulagem da distância focal da câmera.

- Sensores de profundidade têm grande capacidade de observar a distância do robô a elementos do ambiente, porém eles possuem dificuldades em determinar a direção em que se encontra o elemento em relação ao robô, em especial sonares. Mesmo laser e infravermelho, que possuem resolução angular bem maior que sonar, têm dificuldade em detectar pontos de descontinuidade espacial (fim de paredes, por exemplo) (BORGES, 2005). Já a visão tem boa acurácia angular, porém não consegue determinar com acurácia a profundidade que o elemento se encontra. A visão estéreo consegue obter alguma informação sobre profundidade, porém com acurácia bem menor que os sensores de profundidade.
- Sensores de profundidade conseguem observar descontinuidades espaciais, o que os permite detectar novos objetos sendo observados ou uma mudança na forma de um objeto observado. Já a visão tem maior facilidade em determinar descontinuidades fotométricas, que nem sempre indicam mudanças de forma. Por exemplo, para visão uma mudança de cor na parede se destaca mais do que um canto da parede, enquanto que um sensor de profundidade detectaria o canto, mas não a mudança de cor.

Vários trabalhos encontrados na literatura já mostraram combinações dessas duas técnicas, além de observações de deslocamento, seja para Localização ou para Mapeamento. Akbarally e Kleeman (1996) integram uma matriz de sonares com visão monocular para obter uma reconstrução 3-D de um ambiente interno. Kortenkamp e Weymouth (1994) integram um sonar rotatório com visão monocular para identificar *gateways* (que podem ser classificados como marcos) no ambiente. Bing, Lakshmanan e Hero (2001) combinam um radar rotatório e visão monocular com o objetivo de identificar a partir de um veículo os limites da estrada. A integração é realizada utilizando aproximações das funções de probabilidades de cada sensor. Lanthier, Nussbaum e Sheng (2004) integram visão estéreo, um anel de 24 sonares e um conjunto de 8 sensores infravermelhos para mapear um ambiente. Wilhelm, Böhme e Gross (2002) usam visão monocular e um sonar rotatório para detectar e rastrear pessoas.

Murarka, Modayil e Kuipers (2006) usam visão estéreo e dois lasers rotatórios para determinar um mapa local contendo áreas seguras por onde uma cadeira de rodas semi-autônoma pode transitar. Grandjean (1989) integra um laser rotatório

e visão estéreo com o objetivo de obter uma reconstrução 3-D de um ambiente interno. Borges (2005) integra um laser rotatório com visão monocular com o objetivo da obtenção de mapas 2-D do ambiente. Baltzakis, Argyros e Trahanias (2003) também integram um laser rotatório com visão monocular para mapear um ambiente.

2.4 Estimadores

Conceitualmente um estimador é uma função que aceita um vetor de observações e retorna um vetor indicando o valor de variáveis não observáveis diretamente, denominadas de estado, porém relacionadas de algum modo às observações (HAYKIN, 1995). Um estimador é a base de um algoritmo de Localização, responsável por combinar as observações da melhor forma para obter a postura, levando em conta as características do ruído de cada observação.

Caso as observações não fossem corrompidas por ruído, o estimador ficaria reduzido a apenas uma *aplicação* entre espaços vetoriais (entre o espaço de estados e o espaço de observações) (CALLIOLI; DOMINGUES; COSTA, 1995), bastando utilizar um conjunto mínimo de observações para determinar a postura, pois com observações livres de erro, a postura obtida também seria livre de erros, não necessitando de mais observações para aumentar a acurácia.

Como não existem observações sem ruído, o papel do estimador está em tentar obter a postura mais correta possível com as observações disponíveis. Para que um estimador seja capaz de combinar as observações de modo a minimizar a incerteza da postura, é necessário informar a ele o modelo do ruído que se acredita que esteja corrompendo cada observação. Em geral quase nunca as formas exatas desses ruídos são conhecidas, e mesmo quando são, as formas em geral são complexas demais para serem tratadas por algoritmos computacionalmente eficientes.

Existem diversos tipos de estimadores, e a maior diferença entre eles é como eles tratam o ruído das observações (e também o ruído da postura, que é obtido a partir dos ruídos das observações combinados).

Desde a introdução de um estimador conhecido por Filtro de Kalman (KALMAN, 1960), uma família de estimadores conhecidos por Estimadores Recursivos Bayesianos vem sendo muito utilizada na literatura para resolver problemas em que combinação sensorial e pequeno tempo de resposta são requisitos.

2.5 Estimadores Bayesianos

Primeiro é importante ressaltar que a formulação realizada a seguir e no decorrer do trabalho considera o tempo discretizado numa série de instantes, denominados de iterações. Os instantes começam na iteração zero, onde existe uma estimativa prévia do estado (neste trabalho a postura está contida no estado), representada por $\bar{\mathbf{x}}(\mathbf{0})$ (estimativa do estado $\mathbf{x}$ na iteração, ou instante, zero). A partir da iteração 1, é considerado que existe disponível uma observação relativa ao estado na dada iteração, representada por $\mathbf{y}(\mathbf{n})$ (observação $\mathbf{y}$ do estado $\mathbf{x}$ na iteração n).

Representando o estado sendo estimado (no caso, a postura) na iteração n pelo vetor aleatório $\mathbf{x}(\mathbf{n})$, e sendo $\mathbf{Y}(\mathbf{n}) = [\mathbf{y}(\mathbf{1}), \dots, \mathbf{y}(\mathbf{n})]$ uma matriz contendo as observações $\mathbf{y}(\mathbf{i})$ obtidas em n iterações, os estimadores Bayesianos são construídos com o objetivo de encontrar o valor de $\mathbf{x}(\mathbf{n})$ que maximize a função de densidade de probabilidades (ou *pdf*):

$$f_{\mathbf{x}|\mathbf{Y}}(\mathbf{x}(\mathbf{n})|\mathbf{Y}(\mathbf{n})) = \frac{f_{\mathbf{Y}|\mathbf{x}}(\mathbf{Y}(\mathbf{n})|\mathbf{x}(\mathbf{n}))f_{\mathbf{x}}(\mathbf{x}(\mathbf{n}))}{f_{\mathbf{Y}}(\mathbf{Y}(\mathbf{n}))}. \quad (2.1)$$

Diversos tipos de estimadores Bayesianos foram apresentados na literatura, e uma das grandes diferenças apresentadas entre eles está na forma como cada um aproxima a *pdf* apresentada na equação 2.1. Uma aproximação muito comum é considerar que $\mathbf{x}$ possa ser aproximado por um processo de Markov de primeira ordem, ou seja, que $\mathbf{x}(\mathbf{n})$ respeite uma *pdf* da forma dada por:

$$p(\mathbf{x}(\mathbf{n})|\mathbf{x}(\mathbf{n}-1)), \quad (2.2)$$

onde o valor assumido pelo estado $\mathbf{x}$ num dado instante está relacionado ao valor assumido pelo estado x no instante anterior, e a apenas ao instante anterior. Isso permite o desenvolvimento de estimadores mais eficientes, os já citados Estimadores Recursivos Bayesianos. Esses estimadores, em vez de considerar sempre todas as observações realizadas até então para obter uma estimativa, trabalham apenas com as observações obtidas na iteração corrente e com a estimativa obtida na iteração anterior. No apêndice II esses estimadores são tratados com maior profundidade.

Ao se trabalhar com estimadores as *pdfs* apresentadas em geral são contidas em modelos específicos com o objetivo de organizar e modularizar suas implementações. A *pdf* apresentada na equação 2.1 é contida num modelo que determina o valor da observação $\mathbf{y}(\mathbf{n})$ que deveria ser obtida caso o robô se encon-

trasse no estado $\mathbf{x}(\mathbf{n})$ fornecido ao modelo. Esse modelo é conhecido por modelo de observação.

O modelo que contém a ação da *pdf* da equação 2.2, determinando como $\mathbf{x}(\mathbf{n})$ pode ser obtido de $\mathbf{x}(\mathbf{n} - 1)$, é conhecido por modelo de dinâmica. Além da estimativa do estado na iteração anterior, o modelo de dinâmica também aceita como entrada alguma observação ou informação de controle que permita realizar a previsão.

A primeira família de Estimadores Recursivos Bayesianos é composta pelo Filtro de Kalman e suas variações. O Filtro de Kalman parte da suposição que todos os ruídos que perturbam o sistema (observações e sinais de controle) seguem uma distribuição normal com média zero.

Para os estimadores derivados de Filtro de Kalman, os modelos de dinâmica operam recebendo uma estimativa do estado anterior juntamente com a observação do deslocamento ou as entradas de controle do sistema, gerando uma estimativa do estado, segundo o deslocamento estimado.

Os modelos de observação operam recebendo uma estimativa do estado e retornando uma estimativa da observação que seria obtida caso o estado fosse a estimativa fornecida. A figura 2.2 ilustra o fluxo de dados de um Estimador Recursivo Bayesiano.

Pela figura pode-se notar que a observação é usada para gerar uma correção, $\text{corr}_{\mathbf{x}}(\mathbf{i}|\mathbf{i} - 1)$, que é aplicada ao vetor de estado. O bloco K é responsável por transformar a diferença entre a observação esperada ($\bar{\mathbf{y}}(\mathbf{i})$) e a observação mensurada ($\mathbf{y}(\mathbf{i})$) numa diferença entre o estado usado para obter a observação esperada ($\bar{\mathbf{x}}(\mathbf{i})$) e o estado real que gerou a observação mensurada ($\mathbf{x}(\mathbf{i})$), desconhecido). Além dessa transformação, K é responsável também por determinar a importância que é dada para a observação, ou seja, o quanto da correção fornecida pela observação será incorporado na estimativa do estado.

A importância dada à correção fornecida pela observação é tão maior quanto menor for a incerteza da observação em relação à incerteza da estimativa predita até o momento. Analisando por esse ângulo, pode-se dizer então que estimadores recursivos bayesianos (e os demais estimadores bayesianos) obtêm a estimativa do estado ponderando as contribuições de diversas fontes pela incerteza de suas *pdfs*. A incerteza de uma *pdf* pode ser entendida como o espalhamento da mesma. Por exemplo, uma *pdf* cujo gráfico esteja concentrado sobre um ponto terá pequena incerteza. Nas *pdfs* gaussianas, o desvio-padrão é uma boa medida da sua

Figura 2.2: Diagrama de fluxo da estimativa do Filtro de Kalman. O fluxo começa na entrada do modelo de dinâmica, recebendo a estimativa anterior, $\bar{\mathbf{x}}(i-1)$, a observação do movimento atual, $\mathbf{y}_{\text{mov}}(i)$, determinando a evolução da estimativa entre as iterações e gerando $\bar{\mathbf{x}}(i|i-1)$, que alimenta o modelo de observação, produzindo uma estimativa da observação, $\bar{\mathbf{y}}(i)$, a partir de $\bar{\mathbf{x}}(i|i-1)$. A diferença entre a estimativa da observação, $\bar{\mathbf{y}}(i)$, e a observação, $\mathbf{y}(i)$, alimenta o bloco K , conhecido por Ganho de Kalman, gerando a correção de $\text{corr}_x(i|i-1)$, que é usada para determinar a estimativa atual, $\bar{\mathbf{x}}(i)$.

incerteza.

Apesar da popularidade, o Filtro de Kalman original praticamente não é usado, pois ele é um estimador que trabalha com modelos (de dinâmica e de observação) lineares, e na grande maioria dos problemas abordados na literatura são utilizados modelos não-lineares. São então adotadas variações do Filtro de Kalman para modelos não lineares, sendo o mais disseminada o Filtro de Kalman Estendido (EKF) (SCHMIDT, 1966).

Leonard e Durrant-Whyte (1991), Qian, Chellappa e Zheng (2001), Nieto (2003), Welch (1996), Moon, Miura e Shirai (1999), Chiuso (2002), e Borges (2005) usam EKF em problemas de localização e mapeamento.

Mais recentemente Julier e Uhlmann (1997) introduziram uma nova variação do Filtro de Kalman, o Filtro de Kalman Unscented (UKF). O UKF é posterior ao EKF, não sendo ainda bem difundido, apesar de alguns trabalhos (MERWE, 2001), (ORDERUD, 2005) apontarem que o UKF pode obter estimativas melhores que o EKF. Bellotto e Hu (2006) faz uso do UKF para rastrear pessoas a partir de um robô móvel, enquanto Kraft (2003) o usa para resolver o problema de

rastreamento.

Em meados dos anos 90 uma segunda família de Estimadores Recursivos Bayesianos tornou-se popular: os estimadores baseados em Amostragem por Importância com Reamostragem (SIR, sigla do inglês) (THRUN, 2001). Eles são conhecidos por diversos nomes: *Condensation*, Filtro de Partículas, Localização Monte Carlo. Essa é uma técnica que apresenta estimadores mais precisos, porém elevando a complexidade computacional do estimador.

Alguns trabalhos optam por estimadores alternativos (não bayesianos), mas são a minoria. Nakamura e Xu (1989), por exemplo, apresentam uma arquitetura para combinação sensorial baseada em pesos determinados a partir da incerteza esperada de cada sensor, obtendo resultados muito similares aos da teoria bayesiana.

2.6 Discussão

Apesar das várias soluções apresentadas para determinar a localização de um robô, nenhuma delas se preocupou em apresentar uma solução para localização combinando sensores de profundidade e visão que possa ser implementada numa plataforma de baixo custo. Este trabalho apresenta uma solução focando o uso em plataformas de baixo custo. No trabalho é proposto o uso de modelos minimalistas e algoritmos de menor complexidade computacional, para diminuir a exigência de processamento, e de sensores mais acessíveis (sonares em vez de lasers e visão monocular em vez de visão estéreo ou omni-direcional) para atingir esse objetivo.

3 ELViS: Estimador de Localização com Visão e Sonares

As condições esperadas e as premissas adotadas para a execução da tarefa observada neste trabalho são as seguintes:

3.1 Premissas Adotadas

Resumindo o que já foi dito, os objetivos e condições esperadas adotadas neste trabalho são as seguintes:

1. Um robô se movimenta por ambiente interno estruturado conhecido;
2. Utiliza-se um estimador para rastrear a postura de um robô;
3. O estimador utiliza odômetros para determinar o deslocamento do robô, um anel de sonares e visão monocular;
4. O estimador deve apresentar baixo tempo de resposta;
5. Deve-se utilizar de hardware (processador, plataforma robótica e sensores) de baixo custo.

Visando atender os itens acima, é proposto neste trabalho o sistema ELViS: Estimador de Localização Visão-Sonar. ELViS foi projetado para operar utilizando um robô acionado por um par de motores diferenciais, que permitem ao robô realizar rotações sobre o próprio eixo. É considerado que o robô possui um sistema de odometria que permite obter boas estimativas para pequenos deslocamentos (entre meio metro e um metro) em linha reta, e que o robô esteja equipado com um anel semicircular fixo de sonares, que circundam sua metade frontal. É considerado também que o robô esteja equipado com uma câmera montada sobre ele, apontada na direção frontal do robô.

ELViS trabalha considerando o tempo uma grandeza discreta, onde cada intervalo de tempo é delimitado pela necessidade em se gerar uma estimativa da postura. A cada momento em que é produzida uma estimativa da postura uma iteração do estimador é executada, onde as observações disponíveis são processadas, atualizando a estimativa obtida na iteração anterior para obter a nova estimativa. A postura é representada pelo vetor $\mathbf{x}(\mathbf{n})$, composto de três variáveis, $x(n)$, $y(n)$ e $\theta(n)$. O índice n indica a iteração na qual a estimativa foi produzida. Na primeira iteração ($n = 1$), é considerado que existe uma estimativa prévia da postura, $\bar{\mathbf{x}}(0)$.

3.2 Estimador

ELViS combina as observações dos três sensores (odômetros, sonares e visão monocular) utilizando um Filtro de Kalman para modelos não-lineares. Neste trabalho o sistema utiliza duas variações, o EKF e o UKF. Apesar de trabalhos recentes apontarem o UKF como sendo superior ao EKF (MERWE, 2001), (JULIER; UHLMANN, 1997), (RISTIC, 2003), alguns trabalhos apontam que em alguns casos não há diferença perceptível entre os dois (ANDRADE-CETTO; VIDAL-CALLEJA; SANFELIU, 2005), (ST-PIERRE; GINGRAS, 2004) e optam pelo EKF devido a ganhos em tempo de execução (LAVIOLA, 2003), além do que o uso do EKF ainda está mais difundido que o uso do UKF. Uma discussão mais profunda das vantagens de cada filtro pode ser encontrada no apêndice II.

Ambos os estimadores são implementados usando uma variação nos algoritmos visando torná-los robustos contra instabilidades numéricas que podem surgir durante a operação. Mais detalhes sobre o problema e uma descrição dos algoritmos robustos são apresentados no apêndice II, seção II.5.

A escolha por um estimador derivado do Filtro de Kalman em vez de um baseado em Filtro de Partículas tem como motivador o tempo de resposta (complexidade computacional). Apesar de algumas adaptações do Filtro de Partículas possuírem uma complexidade computacional menor que a média (conseguem trabalhar com quantidades menores de partículas), ainda são mais lentos que os Filtros de Kalman e não geram estimativas muito mais acuradas. O uso de Filtros de Partículas pode ser justificado quando se trabalha com Localização Global, ou quando o robô está sujeito a ‘se perder’ com alguma frequência, ou seja, eventualmente estar em situações onde a estimativa da postura se torne inconsistente com as observações ou dúvida (onde posturas distintas podem ser soluções igualmente boas). Nesses casos, é justificável aumentar o custo na determinação da postura

para aumentar a confiabilidade na estimativa.

3.3 Tratamento Sensorial

Foi visto no capítulo anterior que trabalhos anteriores já fazem uso da tripla combinação de sensores de odometria, profundidade e visão para resolver problemas de mapeamento e localização, e alguns até fizeram uso da combinação específica odometria, sonar e visão monocular. Uma diferença importante entre a maioria dos trabalhos apresentados e o trabalho corrente é em como são aproveitadas as observações de cada sensor. Na maioria dos trabalhos revisados, as características complementares dos sensores são exploradas, unindo o que cada sensor oferece de melhor, porém ignorando a contribuição que um dos sensores (ou os dois) poderiam oferecer na característica em que são menos precisos (a informação de profundidade é ignorada no caso de um sensor de visão, por exemplo). No trabalho corrente as observações de cada sensor (sonares, visão e odometria) são usadas para determinar de forma independente um do outro a postura do robô. Essas estimativas completas da postura do robô são então combinadas.

Akbarally e Kleeman (1996) geram um modelo com linhas verticais e horizontais extraídas com a ajuda de uma matriz de sonares, e um segundo modelo com linhas verticais extraídas usando visão monocular. Em seguida é obtida uma intersecção entre os dois modelos (linhas de ambos os modelos que correspondem a um mesmo elemento no ambiente), e com base nessa intersecção um novo modelo é criado combinando as linhas dos dois modelos. Kortenkamp e Weymouth (1994) usam um sonar em base rotatória para identificar configurações geométricas distintas do ambiente ao redor (uma junção ‘T’ de corredores, uma porta à esquerda do robô, etc.), chamadas de *gateways*, e usa visão para diferenciar entre *gateways* que tenham configuração similar. Borges (2005) produz um mapa local formado por segmentos de reta obtidos usando observações de um laser rotativo. Em seguida, usando visão monocular, algumas retas verticais são detectadas e usadas para corrigir as extremidades dos segmentos já obtidos, enquanto outras são adicionadas ao mapa como novos elementos. Baltzakis, Argyros e Trahanias (2003) geram um mapa na forma de uma grade de ocupação com as observações do laser. Em seguida observações consecutivas da visão monocular são utilizadas a fim de determinar inconsistências no mapa obtido, além de adicionar regiões ocupadas no mapa com o intuito de evitar potenciais colisões. Lanthier, Nussbaum e Sheng (2004) combinam visão estéreo, um anel de 24 sonares e um conjunto de 8 sensores infravermelhos para mapear um ambiente. As observações são integradas a

um mapa de ocupação através de um Filtro de Kalman.

Uma das principais diferenças entre o tratamento da visão monocular apresentado neste trabalho e nos trabalhos encontrados na literatura está em considerar os quadros relacionados temporalmente, permitindo extrair alguma informação de profundidade a partir da visão monocular.

Arras, Tomatis e Siegwart (2000) combinam um laser rotativo e visão monocular através de um Filtro de Kalman para determinar a localização de um robô num ambiente interno. O método de integração da informação sensorial é o mais similar ao adotado neste trabalho. Schroeter (2005) combina visão omni-direcional com um anel de 24 sonares usando um Filtro de Partículas para determinar a localização de um robô numa loja de departamentos. A abordagem considera a informação completa de ambos os sensores. Wilhelm, Böhme e Gross (2002) usam visão monocular e um sonar rotativo para rastrear pessoas. Indícios de onde possa existir uma pessoa são extraídos de ambos os sensores e então alinhados para determinar a direção de maior probabilidade de se encontrar uma pessoa.

A maior vantagem em obter a postura completa (estimativas para x , y e θ) a partir de cada sensor não está no aumento da acurácia da postura final, mas sim no aumento da confiabilidade do estimador, seja no combate a erros grosseiros, seja numa eventual perda de um dos sensores. Outra possibilidade é a de um sensor ficar impossibilitado de fornecer observações por um período, seja por algum problema com o sensor, ou seja por não existir ao alcance do sensor elementos do ambiente que o estimulem.

Apesar de neste trabalho cada sensor poder ser tratado como um estimador único da postura do robô, é importante ressaltar que essa abordagem só é válida durante o processo de combinação sensorial para a obtenção da postura. Um processo anterior necessário para o funcionamento dos modelos de observação dos sonares e visão, conhecido por associação de dados (*data association*), faz uso das observações dos odômetros (na verdade, da melhor estimativa corrente da postura) para obter uma estimativa provisória do deslocamento do robô (não necessariamente sua postura). Esse processo consiste em associar (e validar a associação) de uma observação (ou seqüência de observações) a um elemento do ambiente, seja ele previamente mapeado ou não.

3.4 Representação do Ambiente

O conhecimento prévio do ambiente é o principal fator que diferencia a tarefa de Localização daquela de Mapeamento. Sem esse conhecimento prévio, *dead-reckoning* seria o único modo para se estimar a postura. A granularidade, a qualidade e a forma desse conhecimento são determinantes para o funcionamento dos modelos de observação. Não seria muito útil utilizar um sonar se o único conhecimento disponível fossem marcos pintados nas paredes, por exemplo.

Neste trabalho é pressuposto o conhecimento da planta baixa e de objetos estáticos presentes no ambiente que o sonar possa captar (objetos que, no plano de medição do sonar, possuam superfícies de pelo menos 30 centímetros). O sonar pode captar objetos menores, porém o sinal recebido pode apresentar comportamento diferente do esperado pelo modelo de observação do sonar, como será detalhado no capítulo 5. Apenas superfícies planas são registradas no mapa. É considerado que a planta possua apenas segmentos retos.

Além da planta e de objetos relevantes, o mapa guarda informação da posição em que ocorrem variações bruscas de cor ou iluminação (variações fotométricas), de modo a formar uma fronteira na forma de uma linha vertical entre dois blocos de cores distintos, do ponto de vista de um observador que está em pé e olhando para frente, de modo que um plano, que passe pelos seus olhos e pelo horizonte da sua observação, seja paralelo à superfície de deslocamento. Por exemplo, quinas e cantos formados por duas paredes usualmente geram esse tipo de fronteira, porém em geral são de menor importância pois o contraste nesses casos (em especial em cantos) é baixo. Batentes de portas, quadros, canaletas e adornos são fontes dessas fronteiras, chamadas neste trabalho de retas verticais. A posição de algumas dessas retas verticais é armazenada no mapa, juntamente com uma descrição da distribuição de cores na sua vizinhança, conforme será detalhado no capítulo 6.

A planta baixa do ambiente e os objetos relevantes (móveis, por exemplo) são armazenados na forma de conjuntos de segmentos de retas, formando polígonos fechados. As retas verticais são armazenadas como pontos sobre os segmentos, juntamente com a sua descrição local da distribuição de cores (vetor de características). O objetivo do vetor de característica associado a cada reta é tentar torná-las únicas, ou pelo menos diminuir a chance de duas retas verticais próximas serem confundidas. Por esse motivo as retas verticais existentes no ambiente e mapeadas na representação desse ambiente serão chamadas de marcos neste trabalho.

3.5 Estrutura do Estimador

Como já dito, são usadas duas variantes do Filtro de Kalman, o EKF e o UKF. Tanto para um como para o outro, a estrutura e o fluxo de dados é similar ao exibido na figura 2.2. Quando se leva em conta as características dos modelos do ELViS, a estrutura passa a ser a apresentada na figura 3.1. A seguir são detalhados os modelos de observação e dinâmica adotados, apresentando o que cada um espera como entrada, como tratam esses dados e o que geram de saída.

Figura 3.1: O diagrama genérico do Estimador Recursivo Bayesiano é especializado com os modelos do ELViS. Como existem dois modelos de observação (o de visão e o de sonar), as observações ($\mathbf{y}_1(i)$ e $\mathbf{y}_2(i)$) e observações esperadas ($\bar{\mathbf{y}}_1(i)$ e $\bar{\mathbf{y}}_2(i)$) devem ser combinadas num único vetor, $\mathbf{y}_T(i)$ e $\bar{\mathbf{y}}_T(i)$, respectivamente.

3.5.1 Modelo de Dinâmica

A função do modelo de dinâmica é modelar o resultado da ação que ocorre no sistema (no caso, o robô) entre iterações consecutivas. Para realizar isso o modelo de dinâmica leva em conta o funcionamento do sistema, como ele opera, como os sinais de controle gerados atuam e como o ambiente interage com o sistema, a fim de determinar qual deveria ser o estado na iteração i sendo que o sistema estava num dado estado na iteração $i - 1$.

Para modelar a dinâmica de um robô desse modo, um modelo específico deveria ser criado para o robô, levando em conta os sinais que são enviados para acionar os motores, sua velocidade e aceleração, além de uma série de parâmetros que podem variar de robô para robô e iteração para iteração devido ao processo de fabricação, desgaste, consumo de bateria, etc.

Uma segunda alternativa para se construir um modelo de dinâmica é, em vez de se observar os sinais de controle, observar apenas um conjunto de sensores que monitoram algumas variáveis resultantes das ações de controle, como a velocidade do robô, por exemplo, e a partir dessas observações construir um modelo de dinâmica mais simples, eliminando a parte mais complexa, e obtendo um modelo menos sensível à variação nos valores de parâmetros internos, muitas vezes difíceis de estimar.

Além de ser um modelo mais simples e menos sensível à variação de parâmetros do robô, a segunda alternativa tem como vantagens um vetor de variáveis de estado mais enxuto, e também maior acurácia na predição, pois quanto mais complexo o sistema que se deve modelar e quanto mais parâmetros existirem, maior a probabilidade de perda de acurácia. Como desvantagens, pode-se dizer que o modelo de dinâmica construído desse modo perde sua capacidade preditiva, pois só se tem acesso às observações após as ações terem sido executadas, além de que se incorre no risco de se ter erros grosseiros nas respostas do modelo, pois não se está modelando fisicamente o comportamento do robô, mas sim se está confiando em sensores, que podem fornecer observações errôneas.

O fato do modelo não ser mais preditivo não implica em problemas para o algoritmo de localização. Como uma iteração do modelo de dinâmica é muito mais rápida do que uma iteração dos modelos de observação dos outros sensores, o ganho de tempo que se teria prevendo antecipadamente a posição do robô não seria perceptível. Quanto aos erros grosseiros, apesar de existir a possibilidade dos sensores que medem o deslocamento do robô falharem de modo grosseiro, a possibilidade é pequena, e erros muito grosseiros podem ser limitados impondo restrições nos valores esperados através de limites máximos para a aceleração, por exemplo.

Neste trabalho é escolhido usar um modelo de dinâmica baseado nas observações dos odômetros. O modelo recebe a estimativa da postura da iteração anterior, as observações oriundas dos odômetros, e retorna a ‘previsão’ da postura na iteração atual. O modelo de dinâmica é detalhado no capítulo 4.

3.5.2 Modelo de Observação dos Sonares

Os sonares utilizados no sistema ELViS funcionam medindo o tempo decorrido entre a emissão de um curto pulso sonoro de alta frequência e a sua recepção. Conhecendo as características do meio de propagação do pulso, é possível determinar a distância percorrida pelo pulso, logo a distância entre o sonar e algum obstáculo. Como já dito, o robô utilizado conta com um anel semicircular de sonares, localizado na sua parte frontal, composto por oito sonares.

Para não ocorrer interferência entre os sonares, cada um é acionado em instantes distintos. Caso o robô esteja em movimento, cada resposta é obtida em uma posição diferente. Por esse motivo e por outros que serão abordados no capítulo 5, cada sonar é tratado como um sensor independente. Então os sonares representam 8 sensores distintos, com um modelo de observação específico para cada um. A diferença entre os modelos de observação está apenas na informação sobre a posição do sonar em relação ao robô. Essa abordagem é coerente com a proposta de extrair o máximo de informação de cada sensor.

O modelo de observação de um sonar recebe como entrada uma estimativa da postura do robô, e tendo como referência o mapa do ambiente, determina qual seria a observação esperada do sonar para a postura recebida. O modelo de sonar adotado trabalha apenas com reflexões originadas por superfícies planas. Para gerar a observação esperada, o modelo precisa primeiro determinar se a observação obtida é originada de uma superfície plana e em seguida identificar qual superfície mapeada é a mais provável fonte da observação, ou se nenhuma delas o for, indicar como sendo um objeto desconhecido no ambiente. Os detalhes de funcionamento do modelo de observação dos sonares estão descritos no capítulo 5.

3.5.3 Modelo de Observação da Visão Monocular

A visão monocular fornece uma sequência de imagens, ou quadros, registradas numa taxa constante, geralmente de 30 quadros por segundo. Os quadros são fornecidos por uma câmera montada sobre o robô, alinhada como mostra a figura 3.2, de modo que as transições fotométricas bruscas registradas no mapa como retas verticais apareçam nos quadros capturados também como retas verticais.

O modelo de observação de visão monocular adotado neste trabalho opera de modo similar ao modelo de observação dos sonares, associando observações a elementos mapeados. O primeiro passo do modelo de observação é a detecção

Figura 3.2: O eixo Z sistema de coordenadas intrínseco à câmera (Z_C) está alinhado ao eixo Z do sistema de coordenadas de referência.

de elementos do ambiente, para os quais representações matemáticas (modelos) são extraídas. Os elementos procurados nos quadros são transições fotométricas verticais bruscas, ou seja, as retas verticais do ambiente. Após extrair retas verticais de um quadro, é determinada a correspondência entre o conjunto de retas verticais extraídas no quadro atual e o conjunto extraído no quadro anterior. A correspondência implica em encontrar pares de retas verticais (com uma reta de cada conjunto) que são projeções de um mesmo elemento ou região do ambiente.

Em seguida, tenta-se associar a cada reta vertical algum marco conhecido do mapa. Tendo um subconjunto de retas associadas, elas são validadas durante alguns quadros e aquelas que passarem pela validação são usadas para corrigir a estimativa da postura do robô. Essa validação implica em assegurar, entre quadros consecutivos, que as mesmas retas verticais extraídas são associadas sempre aos mesmos marcos. Após a validação, os marcos associados a retas verticais válidas são usados para gerar as observações esperadas.

O modelo de observação trabalha com conjuntos pequenos de retas verticais (entre 10 e 30, em média), que se reduz ainda mais conforme retas são eliminadas pelas etapas de validação, o que possibilita o uso de um algoritmo de complexidade computacional (em tempo de execução) similar ao do modelo de observação dos sonares, uma vez extraídas as projeções dos quadros.

Maiores detalhes sobre o tratamento sensorial da visão monocular estão no capítulo 6.

3.5.4 Processamento Assíncrono

A cada iteração, ELViS gera uma previsão da postura usando os dados dos odômetros e, na sequência, corrige essa estimativa usando as observações dos sonares e da visão. Observando o diagrama do estimador da figura 3.1, e considerando que todas as observações vão estar sempre disponíveis, o fluxo de dados não apresenta maiores problemas. Porém, ao ser considerado o diagrama de obtenção das observações ao longo do tempo, mostrado na figura 3.3, pode-se ver claramente que as observações não ocorrem de modo sincronizado.

Figura 3.3: Fluxo de obtenção de observação dos diversos sensores em função do tempo.

Para lidar com esse problema existem dois caminhos distintos. O primeiro seria sincronizar artificialmente as observações, executando o estimador apenas quando todas as observações se encontrarem disponíveis. Desse modo, ao se processar as observações, apenas a última obtida seria a partir da postura atual do robô. As demais estariam defasadas por um certo período, aumentando a imprecisão das correções obtidas a partir dessas observações. Existem métodos para corrigir as observações levando em conta o deslocamento entre a obtenção da observação e a execução do estimador, porém eles não conseguem corrigir com perfeição, além de introduzirem correlações entre as observações, o que pode trazer implicações na utilização dos estimadores selecionados (EKF e UKF).

O segundo caminho seria tratar cada observação individualmente, no instante em que são obtidas, sem esperar por observações de todos os sensores. É bem possível que a partir de apenas as observações individuais de um dado sensor não seja possível se obter a postura. Nesse caso, é dito que a observação (ou mesmo um conjunto de observações) é incompleto.

Observações completas e incompletas são termos usados para indicar se a partir do conjunto de observações é possível obter o estado do sistema (a postura,

no caso) ou não, ou seja, se o sistema é ou não observável. Por exemplo, um sistema de localização que confia apenas numa bússola para determinar a postura seria não-observável, pois a bússola não permite determinar a posição (x, y) . Logo, a bússola gera observações incompletas. Um GPS também gera observações incompletas, pois é capaz de determinar apenas a posição. Ao combinar os dois sensores, porém, obtém-se um sistema observável.

Esse tipo de abordagem é conhecida por Rastreamento Incremental com Informações Incompletas, (SCAAT, sigla do inglês) (WELCH; BISHOP, 1997). Welch (1996) apresenta um método baseado em Filtro de Kalman para integrar observações incompletas, e prova que, integrar conjuntos de observações *localmente* incompletas (observações incompletas apenas durante um intervalo de tempo) porém *globalmente* completas equivale a integrar um conjunto de observações completas.

SCAAT diz que integrar as observações da bússola e do GPS em uma única iteração ou em duas iterações distintas geraria o mesmo resultado. É claro que para integrar as duas observações na mesma iteração seria necessário que ambas estivessem disponíveis ao mesmo tempo, caso contrário seria necessário sincronizá-las de alguma forma, tornando os resultados dessa abordagem inferiores aos resultados da abordagem SCAAT.

No trabalho aqui desenvolvido, as observações dos odômetros, dos odômetros e visão, e dos odômetros e sonar são interpretadas separadamente com o uso do SCAAT. As observações dos odômetros são sempre usadas porque SCAAT permite apenas separar em iterações distintas observações tratadas como observações (por modelos de observação). Os odômetros neste trabalho são usados como medidas para o modelo de dinâmica. O uso das observações dos odômetros em todas as iterações não causa grande transtorno devido à inserção de atraso entre a captação e o processamento das observações, pois como pode ser visto na figura 3.3 os odômetros geram observações a uma taxa muito maior que os outros sensores, o que permite ter sempre observações dos odômetros próximas às dos sonares ou da visão.

Fazendo uso da técnica SCAAT, a dinâmica de execução de ELViS segue temporalmente o fluxo mostrado na figura 3.4.

Figura 3.4: Fluxo de execução do estimador no tempo. A cada obtenção de uma observação do sonar e visão o estimador é executado, ou a execução é disparada apenas com a informação do odômetro, após um certo deslocamento do robô sem executar outras observações.

4 Modelo de Dinâmica

O modelo de dinâmica descreve como o estado, ou postura, do robô evolui entre dois instantes, ou iterações, consecutivas. A evolução é dada pelo deslocamento do robô, resultado de comandos enviados ao robô. Para modelar a dinâmica, é necessário conhecer a mecânica do deslocamento do robô.

O robô utilizado neste trabalho é um Pioneer 2 - DXe (figura 4.1), da empresa ActivMedia Robotics. O robô se desloca usando duas rodas laterais controladas por motores independentes. As rodas com movimento independente permitem que o robô realize curvas, incluindo rotações sobre o próprio eixo, acionando cada roda com velocidades diferentes, ou mesmo em sentido contrário. Esse modo de controle de deslocamento é conhecido por direção diferencial, e é muito comum entre robôs móveis experimentais.

Como já foi dito em capítulos anteriores, pode-se optar por duas abordagens para se obter o modelo de dinâmica: observando os sinais de controle do sistema ou observando algum sensor que monitora estados internos do sistema. A segunda opção é escolhida pois ela permite eliminar grande parte da complexidade do modelo, permitindo que o vetor de estado seja representado apenas pela postura do robô. No primeiro caso, seria necessário adicionar pelo menos as velocidades e acelerações do robô ao vetor estado.

O diagrama do robô visto por baixo é esboçado na figura 4.2. Com base nessa visão é construído o modelo de dinâmica. É considerado que entre duas iterações as rodas percorram distâncias distintas, d_L e d_R . Essas distâncias são obtidas através de *encoders* colocados nos eixos de cada roda, que monitoram a quantidade de rotação de cada uma. A distância média percorrida pelo robô é dada por:

$$d_m = \frac{d_R + d_L}{2}. \quad (4.1)$$

O desvio da orientação é obtido mudando o ponto de vista para um observador sobre uma das rodas. Esse observador verá o movimento descrito pela figura 4.3.

Figura 4.1: Robô Pioneer 2 - DXe

Nessa situação tem-se que o movimento observado da roda direita, dado por $d_R - d_L$, representa um arco da circunferência de raio b , sendo b a distância entre os centros das rodas. Então $\delta\theta_m$ é dado por:

$$\delta\theta_m = \frac{d_R - d_L}{b}. \quad (4.2)$$

O robô usado neste trabalho já fornece os valores de d_m e $\delta\theta_m$ como as observações internas do robô. O estado $\mathbf{x}(\mathbf{n})$ é descrito conforme a equação a seguir, sendo que $x(n)$, $y(n)$ e $\theta(n)$ indicam a postura do robô:

$$\mathbf{x}(\mathbf{n}) = [x(n), y(n), \theta(n)]^T. \quad (4.3)$$

O modelo de dinâmica adotado é:

$$\begin{aligned} x(n) &= x(n-1) + d_m(n)\cos(\theta(n)) \\ y(n) &= y(n-1) + d_m(n)\sen(\theta(n)) \\ \theta(n) &= \theta(n-1) + \delta\theta_m(n) \end{aligned} \quad (4.4)$$

Esse modelo é uma aproximação bem simples que só deve ser usado quando

Figura 4.2: Vista inferior do robô. No modelo de direção diferencial cada roda assume uma velocidade diferente, v_R e v_L , percorrendo distâncias distintas entre duas iterações, d_R e d_L . A partir das distâncias percorridas pelas rodas é possível determinar o deslocamento d_m e a rotação $\delta\theta_m$ do centro do robô; r indica o raio médio das rodas e b a distância entre os centros das rodas.

se considera pequenos deslocamentos entre iterações consecutivas, além de não levar em conta características dinâmicas do robô (acelerações, velocidades), que poderiam ser relevantes caso o robô fosse se deslocar em altas velocidades. Porém é considerado que o robô se desloca em baixas velocidades, da ordem de $0,2m/s$ e que a taxa de execução do modelo de dinâmica seja da ordem de $50ms$. Vários autores que trabalham com robôs em condições similares adotam o mesmo modelo sem reportar problemas (LEONARD; DURRANT-WHYTE, 1991), (JENSFELT, 1999).

O Filtro de Kalman Estendido utiliza a matriz $\mathbf{F}(\mathbf{n})$ (ver o apêndice II.5.1 para detalhes), obtida derivando as funções descritas na equação 4.4 em relação ao estado, para propagar a incerteza e obter o Ganho de Kalman, como ilustram os passos descritos a seguir:

Partindo do modelo de dinâmica $f(\cdot)$ dado pela equação:

$$f(\mathbf{x}(\mathbf{n} - \mathbf{1})) = \begin{pmatrix} x(n) \\ y(n) \\ \theta(n) \end{pmatrix} = \begin{pmatrix} x(n - 1) + d_m(n)\cos(\theta_m(n - 1)) \\ y(n - 1) + d_m(n)\sen(\theta_m(n - 1)) \\ \theta(n - 1) + \delta\theta_m(n) \end{pmatrix}, \quad (4.5)$$

Figura 4.3: Deslocamento do robô sob o ponto de vista de um observador localizado na roda direita.

deseja-se obter:

$$\mathbf{F}(\mathbf{n}) = \frac{\partial f(\mathbf{x}(\mathbf{n}-1))}{\partial \mathbf{x}} \mid \mathbf{x} = \bar{\mathbf{x}}(\mathbf{n}-1).$$

Derivando os elementos de $f(\cdot)$ pelos elementos de $\mathbf{x}$ obtém-se:

$$\mathbf{F}(\mathbf{n}) = \begin{pmatrix} 1 & 0 & -d_m(n)\sin(\bar{\theta}_m(n-1)) \\ 0 & 1 & d_m(n)\cos(\bar{\theta}_m(n-1)) \\ 0 & 0 & 1 \end{pmatrix}. \quad (4.6)$$

4.1 Incerteza do deslocamento

É considerado que as incertezas das variáveis observadas, d_m e $\delta\theta_m$, seguem distribuições gaussianas e não-correlacionadas entre si e entre si mesmas em instantes distintos (não auto-correlacionadas). O modelo de dinâmica considerando as in-

certezas como variáveis distintas, $d_{err}(n)$ e $\delta\theta_{err}(n)$, é:

$$x(n) = x(n-1) + (d_m(n) + d_{err}(n))\cos(\theta(n-1)), \quad (4.7)$$

$$y(n) = y(n-1) + (d_m(n) + d_{err}(n))\sin(\theta(n-1)), \quad (4.8)$$

$$\theta(n) = \theta(n-1) + \delta\theta_m(n) + \delta\theta_{err}(n),$$

$$d_{err}(n) \sim N(0, \sigma_{d_m(n)}^2),$$

$$\delta\theta_{err}(n) \sim N(0, \sigma_{\delta\theta_m(n)}^2),$$

onde $\sigma_{d_m(n)}^2$ e $\sigma_{\delta\theta_m(n)}^2$ são as variâncias das incertezas $d_{err}(n)$ e $\delta\theta_{err}(n)$, respectivamente.

Essa aproximação permite simplificar o modelo e se adequar aos requisitos do Filtro de Kalman, porém não retrata a realidade fielmente. Segundo o modelo adotado, considerando uma operação de rastreamento usando apenas o modelo de dinâmica (localização tipo *dead reckoning*), a *pdf* do estado também apresentará distribuição gaussiana (pois é resultante de uma soma de gaussianas), e após um certo deslocamento, ao se traçar uma curva fechada indicando os pontos em que a *pdf* da posição do robô (x e y apenas) apresenta o mesmo valor (uma espécie de curva equipotencial, ou curva de nível) surgirá o desenho de uma elipse, como pode ser visto na figura 4.4.

Figura 4.4: Uma curva representando a *pdf* da posição do robô assumindo distribuição gaussiana da observação do modelo de dinâmica. A elipse representa um limite, no plano de deslocamento do robô, onde a *pdf* apresenta o mesmo valor (curva equipotencial). A seta indica a estimativa da postura do robô.

Na realidade, a elipse apenas aproxima o formato real dessa curva. A curva real se assemelha a uma meia-lua, conforme mostra a figura 4.5.

O desejado é que a elipse aproximada circunscreva a curva real, superestimando a incerteza real. Caso a incerteza seja subestimada, será dada mais importância do que deveria para a estimativa obtida pelo modelo de dinâmica, correndo o risco de ignorar em demasia informações fornecidas por outras observações.

Figura 4.5: Conjunto de partículas distribuídas segundo uma distribuição próxima à *pdf* real da postura do robô, quando utilizado apenas o modelo de dinâmica. A seta indica a estimativa da postura do robô.

A incerteza associada ao deslocamento do robô dependem do próprio deslocamento: quanto maior o deslocamento, maior a incerteza. Se não ocorrer deslocamento, não há incerteza. A incerteza num deslocamento de $10m$ é maior do que num deslocamento de $10cm$. Considerando isso, nesse trabalho adotou-se que as variâncias $\sigma_{d_m(n)}^2$ e $\sigma_{\delta\theta_m(n)}^2$ são proporcionais ao quadrado do valor de $d_m(n)$ e $\delta\theta_m(n)$.

O modelo adotado para a determinação de $\sigma_{d_m(n)}^2$ e $\sigma_{\delta\theta_m(n)}^2$ considera inicialmente a participação de $d_m(n)$ e $\delta\theta_m(n)$ no cálculo de ambas as variâncias da seguinte forma:

$$\begin{aligned}\sigma_{d_m(n)}^2 &= (d_m(n) \cdot \sigma_{d_m}^{d_m})^2 + (\delta\theta_m(n) \cdot \sigma_{d_m}^{\delta\theta_m})^2 \\ \sigma_{\delta\theta_m(n)}^2 &= (\delta\theta_m(n) \cdot \sigma_{\delta\theta_m}^{\delta\theta_m})^2 + (d_m(n) \cdot \sigma_{\delta\theta_m}^{d_m})^2,\end{aligned}$$

onde $\sigma_{d_m}^{d_m}$ indica a influência do deslocamento linear sobre a incerteza do deslocamento linear, $\sigma_{d_m}^{\delta\theta_m}$ indica a influência do deslocamento angular sobre a incerteza do deslocamento linear, $\sigma_{\delta\theta_m}^{d_m}$ indica a influência do deslocamento linear sobre a incerteza do deslocamento angular, e $\sigma_{\delta\theta_m}^{\delta\theta_m}$ indica a influência do deslocamento angular sobre a incerteza do deslocamento angular.

Foram realizados experimentos para determinar o valor de cada um desses coeficientes. Para determinar o valor dos coeficientes de d_m ($\sigma_{d_m}^{d_m}$ e $\sigma_{d_m}^{\delta\theta_m}$) realizou-se experimentos movimentando o robô sobre uma linha reta, considerando-se que o robô não realiza nenhuma mudança de orientação durante o deslocamento (eliminando a influência dos coeficientes de $\delta\theta_m$), e para determinar os coeficientes de $\delta\theta_m$ realizou-se experimentos com o robô rotacionando sobre o próprio eixo.

Como resultado dos experimentos foi determinado que o efeito de $\sigma_{d_m}^{\delta\theta_m}$ poderia ser desprezado por terem sido detectadas apenas variações muito pequenas na posição devido a rotação do robô e, ainda assim, apenas após o robô completar revoluções próximas a 360° sobre o próprio eixo, situações estas pouco comuns na operação normal do robô. Com base nos resultados dos experimentos, propõe-se a seguinte forma de cálculo dos valores $\sigma_{d_m(n)}^2$ e $\sigma_{\delta\theta_m(n)}^2$:

$$\begin{aligned}\sigma_{d_m(n)}^2 &= (d_m(n) \cdot \sigma_{d_m}^{d_m})^2 \\ \sigma_{\delta\theta_m(n)}^2 &= (\delta\theta_m(n) \cdot \sigma_{\delta\theta_m}^{\delta\theta_m})^2 + (d_m(n) \cdot \sigma_{\delta\theta_m}^{d_m})^2.\end{aligned}$$

4.2 Incerteza do modelo de dinâmica

Para satisfazer as descrições do EKF e do UKF (mostradas no apêndice II.5), é necessário obter a matriz de covariância do estado do modelo de dinâmica, $\mathbf{Q}$. Como é considerado que as observações do modelo de dinâmica não são correlacionadas com as variáveis que compõem o estado, então caso o sistema fosse linear sobre essas variáveis (estado e observações), poder-se-ia separar o sistema em duas matrizes, cada uma relativa a um dos conjuntos de variáveis não-correlacionadas:

$$\mathbf{x}(\mathbf{n}) = f(\mathbf{x}(\mathbf{n}-1), \mathbf{obs}(\mathbf{n})) \sim \mathbf{F}(\mathbf{n})\mathbf{x}(\mathbf{n}-1) + \mathbf{F}_{\mathbf{obs}}(\mathbf{n})\mathbf{obs}(\mathbf{n}), \quad (4.9)$$

onde $\mathbf{obs}(\mathbf{n}) = [d_m(n), \delta\theta_m(n)]^T$ é o vetor de observações e $\mathbf{F}_{\mathbf{obs}}(\mathbf{n})$ é a matriz que representa a parte do sistema linear relativa às observações. $\mathbf{F}(\mathbf{n})$ é a matriz que representa a parte do sistema linear relativa ao estado. Tanto $\mathbf{F}_{\mathbf{obs}}(\mathbf{n})$ como $\mathbf{F}(\mathbf{n})$ podem representar sistemas lineares ou linearizações de sistemas para os respectivos conjuntos de variáveis.

Porém o modelo de dinâmica não pode ser escrito na forma da equação 4.9 pois não é linear. Contudo, para determinar a propagação da incerteza originária das observações do modelo, pode ser feita uma aproximação, similar à usada no EKF. O EKF lineariza um sistema não-linear usando uma aproximação de primeira ordem de Taylor ao redor da melhor aproximação até então da estimativa do estado (veja as equações II.15 e II.16, no apêndice II para detalhes). Então as matrizes $\mathbf{F}(\mathbf{n})$ e $\mathbf{F}_{\mathbf{obs}}(\mathbf{n})$ podem ser obtidas derivando as equações em relação do conjunto de variáveis que cada uma atua. Como já visto, $\mathbf{F}(\mathbf{n})$ é dada pela

equação 4.6, enquanto $\mathbf{F}_{\text{obs}}(\mathbf{n})$ é dada por:

$$\mathbf{F}_{\text{obs}}(\mathbf{n}) = \begin{pmatrix} \cos(\bar{\theta}(n-1)) & 0 \\ \sin(\bar{\theta}(n-1)) & 0 \\ 0 & 1 \end{pmatrix} \quad (4.10)$$

A expansão de Taylor do modelo de dinâmica, com o sistema linearizado já separado nas duas matrizes, é:

$$\begin{aligned} f(\mathbf{x}(\mathbf{n}-1), \text{obs}(\mathbf{n})) &= f(\bar{\mathbf{x}}(\mathbf{n}-1), \overline{\text{obs}}(\mathbf{n})) + \mathbf{F}(\mathbf{n})(\mathbf{x}(\mathbf{n}-1) - \bar{\mathbf{x}}(\mathbf{n}-1)) + \\ &\quad \mathbf{F}_{\text{obs}}(\mathbf{n})(\text{obs}(\mathbf{n}) - \overline{\text{obs}}(\mathbf{n})), \end{aligned} \quad (4.11)$$

onde $\overline{\text{obs}}(\mathbf{n})$ é a estimativa da observação do modelo de dinâmica, ou seja, os valores obtidos da observação (dos odômetros).

Para obter $\mathbf{Q}$ a partir de 4.11 basta calcular a covariância da função $f(\mathbf{x}(\mathbf{n}), \text{obs}(\mathbf{n}))$ usando a linearização realizada acima, e isolar a contribuição originária do vetor de observações:

$$\begin{aligned} &E[(f(\mathbf{x}(\mathbf{n}-1), \text{obs}(\mathbf{n})) - f(\bar{\mathbf{x}}(\mathbf{n}-1), \overline{\text{obs}}(\mathbf{n}))) \cdot \\ &\quad (f(\mathbf{x}(\mathbf{n}-1), \text{obs}(\mathbf{n})) - f(\bar{\mathbf{x}}(\mathbf{n}-1), \overline{\text{obs}}(\mathbf{n})))^T] \\ &= E[(\mathbf{F}(\mathbf{n})(\mathbf{x}(\mathbf{n}-1) - \bar{\mathbf{x}}(\mathbf{n}-1)) + \mathbf{F}_{\text{obs}}(\mathbf{n})(\text{obs}(\mathbf{n}) - \overline{\text{obs}}(\mathbf{n}))) \cdot \\ &\quad (\mathbf{F}(\mathbf{n})(\mathbf{x}(\mathbf{n}-1) - \bar{\mathbf{x}}(\mathbf{n}-1)) + \mathbf{F}_{\text{obs}}(\mathbf{n})(\text{obs}(\mathbf{n}) - \overline{\text{obs}}(\mathbf{n})))^T] \\ &= E[\mathbf{F}(\mathbf{n})(\mathbf{x}(\mathbf{n}-1) - \bar{\mathbf{x}}(\mathbf{n}-1))(\mathbf{x}(\mathbf{n}-1) - \bar{\mathbf{x}}(\mathbf{n}-1))^T \mathbf{F}^T(\mathbf{n})] + \\ &\quad E[\mathbf{F}_{\text{obs}}(\mathbf{n})(\text{obs}(\mathbf{n}) - \overline{\text{obs}}(\mathbf{n}))(\text{obs}(\mathbf{n}) - \overline{\text{obs}}(\mathbf{n}))^T \mathbf{F}_{\text{obs}}^T(\mathbf{n})]. \end{aligned} \quad (4.12)$$

Considerando que a covariância de $\mathbf{x}(\mathbf{n})$ é $\mathbf{P}_{\mathbf{x}}(\mathbf{n})$ e que:

$$\mathbf{Q}_{\text{obs}}(\mathbf{n}) = \begin{pmatrix} \sigma_{d_m(n)}^2 & 0 \\ 0 & \sigma_{\delta\theta_m(n)}^2 \end{pmatrix}, \quad (4.13)$$

substituindo na equação 4.12, a covariância aproximada da função $f(\mathbf{x}(\mathbf{n}), \text{obs}(\mathbf{n}))$ é dada por:

$$\begin{aligned} &\mathbf{F}(\mathbf{n})\mathbf{P}_{\mathbf{x}}(\mathbf{n}-1)\mathbf{F}^T(\mathbf{n}) + \mathbf{F}_{\text{obs}}(\mathbf{n})\mathbf{Q}_{\text{obs}}(\mathbf{n})\mathbf{F}_{\text{obs}}^T(\mathbf{n}) \\ &= \mathbf{F}(\mathbf{n})\mathbf{P}_{\mathbf{x}}(\mathbf{n}-1)\mathbf{F}^T(\mathbf{n}) + \mathbf{Q}(\mathbf{n}) \end{aligned} \quad (4.14)$$

A partir dessa dedução, obtém-se que $\mathbf{Q}(\mathbf{n}) = \mathbf{F}_{\text{obs}}(\mathbf{n})\mathbf{Q}_{\text{obs}}(\mathbf{n})\mathbf{F}_{\text{obs}}^T(\mathbf{n})$.

4.3 Determinação dos parâmetros usados no modelo de dinâmica

O modelo de dinâmica depende dos valores dos parâmetros das variâncias das observações dos odômetros, $\sigma_{d_m}^{d_m}$, $\sigma_{\delta\theta_m}^{d_m}$ e $\sigma_{\delta\theta_m}^{\delta\theta_m}$, e dependem do tipo de superfície na qual o robô se desloca.

Neste trabalho o robô é testado apenas sobre superfícies lisas de concreto ou granito. Para determinar os valores dos parâmetros para esse tipo de superfície, foram realizados experimentos que consistiram em 10 deslocamentos do robô andando numa linha reta de 10 metros e 10 rotações de 180 graus. A partir desses dados, os valores obtidos foram os seguintes:

$$\begin{aligned}\sigma_{d_m}^{d_m} &= 0,033, \\ \sigma_{\delta\theta_m}^{\delta\theta_m} &= 0,035, \\ \sigma_{\delta\theta_m}^{d_m} &= 0,001.\end{aligned}$$

4.4 Resumo do Capítulo

A cada iteração efetua-se uma observação dos odômetros, $\mathbf{obs}(\mathbf{n}) = [d_m(n), \delta\theta_m(n)]^T$. Com base na estimativa da postura na iteração anterior, $\bar{\mathbf{x}}(\mathbf{n} - 1)$, é predita a postura atual, $\bar{\mathbf{x}}(\mathbf{n}|\mathbf{n} - 1)$, usando o modelo de dinâmica, $f(\cdot)$:

$$\bar{\mathbf{x}}(\mathbf{n}) = f(\bar{\mathbf{x}}(\mathbf{n} - 1), \mathbf{obs}(\mathbf{n})) \sim \mathbf{F}(\mathbf{n})\mathbf{x}(\mathbf{n} - 1) + \mathbf{F}_{\mathbf{obs}}(\mathbf{n})\mathbf{obs}(\mathbf{n})$$

Tanto o EKF como o UKF (os estimadores usados neste trabalho) consideram que a postura é uma variável aleatória que segue uma distribuição gaussiana. Para descrever a postura totalmente, então, é necessário determinar, além da sua média ($\bar{\mathbf{x}}(\mathbf{n})$), a sua covariância ($\mathbf{P}_{\mathbf{x}}(\mathbf{n})$). Aplicando as técnicas de linearização descritas nesse capítulo, obtém-se o seguinte:

$$\mathbf{Q}(\mathbf{n}) = \mathbf{F}_{\mathbf{obs}}(\mathbf{n})\mathbf{Q}_{\mathbf{obs}}(\mathbf{n})\mathbf{F}_{\mathbf{obs}}^T(\mathbf{n}),$$

onde $\mathbf{Q}_{\mathbf{obs}}$ é dado pela equação 4.13, e F_{obs} é dado pela equação 4.10.

5 Modelo de Observação: Sonares

O sonar é um sensor de profundidade, que opera emitindo e captando ondas moduladas de som, e com base no tempo entre emissão e recepção, calcula a distância do alvo. Um sensor de profundidade ideal detecta a distância do objeto que está localizado na direção que o sensor está apontado. Um sensor de profundidade que usa laser tem um comportamento próximo a um sensor de profundidade ideal, por exemplo. A partir de um sensor de profundidade de laser montado numa base rotativa, é possível gerar um mapa de profundidades¹ cuja interpretação visual é intuitiva.

O sonar não funciona como um laser. Um mapa de profundidades gerado por um sonar montado numa base rotativa gera um mapa de profundidades cuja interpretação visual não é intuitiva, como pode ser visto na figura 5.1, fazendo parecer que as observações do sonar não são confiáveis.

Das figuras 5.1 e 5.2 pode-se reparar que, em algumas regiões, as observações dos sonares se aproximam das paredes, em especial quando a parede e o sonar estão alinhados de tal modo que uma linha partindo na direção que o sonar está apontando intersecta a parede formando um ângulo próximo de 90 graus com ela. Nessas regiões, as observações do sonar geram arcos que tangenciam as paredes. Efeito semelhante ocorre com a maioria das quinas e cantos, porém o arco formado toca o contorno apenas em um ponto (o canto ou a quina). É interessante notar que, em geral, quanto mais próxima da parede estiver o sonar, menor é o arco formado. Arcos formados que não se encaixam nos padrões descritos são resultados de efeitos secundários da interação dos pulsos sonoros com os elementos do ambiente, e os arcos mais distantes do robô e das paredes indicam que o sonar não detectou nada, e retornou um valor padrão de distância

¹Um mapa de profundidades é o resultado das observações de um sensor de profundidade rotatório coletadas durante uma rotação do sensor, usualmente de 360 graus. Neste caso, o centro do mapa de profundidades é a posição do sensor, e a partir dessa posição um ponto é adicionado na direção de cada observação. A distância entre o ponto e o centro indica a distância observada.

Figura 5.1: Mapa de Profundidades denso (formado por observações realizadas em curtos deslocamentos angulares) obtido com um sonar rotatório e o contorno da sala. O sonar é representado pelo triângulo no centro da figura. Obtido de Leonard e Durrant-Whyte (1992).

máxima.

Segundo Leonard e Durrant-Whyte (1992), ao se analisar o funcionamento de sonares montados em bases rotatórias, pode-se obter um modelo confiável para interpretar corretamente a resposta obtida dos mesmos.

Os pulsos de som emitidos e captados são filtrados pelo próprio sensor, de modo que apenas reflexões especulares sejam consideradas, rejeitando retornos provenientes de reflexões difusas. Paredes, por exemplo, geram reflexões especulares dos pulsos sonoros. Para um pulso ser detectado, ele precisa ser refletido por algum obstáculo e retornar para o detector. Considerando que um pulso seja formado por um único raio muito fino, disparado na direção que o sonar aponta, um obstáculo para ser detectado tem que estar alinhado com o sonar de modo que a reflexão do pulso retorne pela mesma trajetória que realizou para atingir o obstáculo. No caso de uma parede, o pulso deve incidir perpendicularmente sobre ela.

Na realidade, o pulso sonoro emitido não pode ser modelado por um feixe muito fino. Numa aproximação mais fiel à realidade, o pulso é modelado por um cone cujo vértice coincide com o sonar e a reta que passa pelo seu vértice e pelo centro de sua base é paralela à direção que o sonar aponta. O ângulo de

Figura 5.2: Mapa de Profundidades denso com um sonar rotatório obtido a partir de simulação. O sonar é representado pelo triângulo no centro da figura. Obtido de Leonard e Durrant-Whyte (1992)

abertura do cone é β . O pulso vai se expandindo à medida que se afasta do sonar. Pode-se pensar no pulso sonoro como raios muito finos e muito próximos que vão se afastando uns dos outros à medida que se afastam do sonar. Então, voltando ao exemplo da parede, para que o sonar consiga observá-la, é necessário que pelo menos um dos raios contidos no cone sonoro incida perpendicularmente sobre ela.

Pode-se concluir que para o sonar observar uma parede é necessário que o centro do pulso sonoro emitido incida sobre a parede formando um ângulo α de tal modo que $90^\circ - \frac{\beta}{2} \leq \alpha \leq 90^\circ + \frac{\beta}{2}$. A distância entre a parede e o sonar é dada então pela distância percorrida pelo raio que incidiu perpendicularmente na parede, ou seja, a distância da parede até o sonar.

A análise realizada considera o pulso sonoro apenas como um cone de som, porém, na verdade, o pulso real apresenta outros dois cones menores, localizados nos lados do cone principal, separados dele por um ‘cone de silêncio’. A figura 5.3 mostra um gráfico com um trecho da resposta do sonar rotatório em que é observada uma parede. No eixo y estão as profundidades observadas pelo sonar, e no eixo x a orientação do sonar em relação à parede observada. O grau 0° indica que uma reta orientada na direção em que o sonar aponta é perpendicular à parede. A figura 5.4 mostra uma visão alternativa dos mesmos dados, na forma

de um mapa de profundidades. Como se pode notar na figura 5.3, o cone principal do pulso sonoro detecta a parede numa faixa de 50 graus (de -20 a 25 graus), aproximadamente. Seguindo o cone principal estão duas regiões pequenas onde a parede não é detectada. Essas regiões de silêncio são seguidas por pequenas regiões (de -35 a -25 graus e de 30 a 40 graus) que novamente contêm observações da parede, porém mais ruidosas do que as do cone principal.

Figura 5.3: Gráfico *orientação × profundidade* da resposta do sonar para uma parede. Valores acima de 1,20m indicam observações fora do alcance do sensor. Obtido de Leonard e Durrant-Whyte (1992).

A distância angular entre o cone principal e os secundários, assim como o tamanho dos cones secundários, depende das características e da distância do obstáculo ao sonar. O valor do ângulo de abertura β então depende da distância e características do obstáculo. Porém, para simplificar o modelo e evitar o trabalho de levantar os valores de β para diversos tipos de materiais e distâncias, optou-se por trabalhar com um valor fixo para β , de tal modo que o valor será o menor valor encontrado nos experimentos para sua determinação.

Apesar do trabalho corrente considerar apenas observações de paredes, a teoria permite também detectar cantos e quinas. Observando a parte superior esquerda da figura 5.2 pode-se ver que as observações do sonar relativas ao canto produzem um arco. Esse arco é resultado da geometria peculiar de um canto de 90 graus. Os pulsos do sonar refletem em ambas as paredes que definam o

bandas laterais do pulso sonoro. Utilizando as RCDs extraídas de um mapa de profundidades de um ciclo de operação do sonar, e conhecendo a disposição dos objetos no ambiente, é possível associar as RCD aos objetos do ambiente.

5.2 Adaptação do método RCD

Leonard e Durrant-Whyte (1992), e Kuc e Siegel (1987) usam sonares rotatórios capazes de gerar mapas de profundidades densos, com quase 2 observações por grau de rotação. Neste trabalho, por outro lado, são usados oito sonares fixos dispostos num semicírculo conforme ilustra a figura 5.5. Esse sistema de sonares, sem se deslocar, é capaz de gerar apenas um mapa esparso de profundidades, exemplificado pela figura 5.6. Esse tipo de mapa não consegue capturar informação suficiente para fazer uso do método de RCDs diretamente.

A solução adotada neste trabalho segue o mesmo princípio das soluções apresentadas por Tardós (2002) e por Wijk, Jensfelt e Christensen (1998): em vez de considerar observações capturadas variando a orientação do sonar para uma posição fixa, são consideradas observações para uma orientação fixa variando a posição do sonar. Para obter o conjunto de observações desejado, é necessário mover o robô durante a captura das observações.

Em seu trabalho, Tardós (2002) coleta uma massa de dados durante um deslocamento do robô, e os processa utilizando a Transformada de Hough (GONZALEZ; WOODS, 1992) para obter um mapa local do ambiente com os objetos observados. O método adotado garante boa resistência contra erros grosseiros e permite obter a posição de paredes, cantos e quinas, porém não é ideal para um estimador que necessita operar em tempo real, pois é necessário coletar previamente um conjunto de observações para cada aplicação do método, o que implica em corrigir a postura do robô com uma frequência muito menor do que o sonar permitiria.

O método proposto no trabalho corrente tenta evitar grandes períodos sem correção na postura estimada, reduzindo a quantidade mínima de observações que precisam ser coletadas antes que correções possam ser realizadas.

5.3 Restrições adotadas

Neste trabalho são consideradas apenas observações provenientes de superfícies planas perpendiculares ao plano de deslocamento do robô, como paredes, originadas pelo cone principal do pulso sonoro. As demais observações são rejeitadas.

Figura 5.5: Oito sonares formam um semicírculo na região frontal do robô. Os valores indicam a orientação, em graus, dos sonares em relação à frente do robô.

O motivo para essa restrição é facilitar a análise dos resultados e aumentar a confiabilidade das observações não rejeitadas. Pode-se ainda justificar que a perda de informação não é tão grande, pois em geral superfícies planas são responsáveis pela maioria das observações num ambiente interno estruturado.

Cada sonar do anel semicircular é tratado individualmente pelo modelo proposto. Assim, as observações de um sonar não influenciam no tratamento das observações dos demais. Pela figura 5.5 pode-se notar que cada sonar está espaçado dos sonares adjacentes por 20 ou 40 graus, dependendo da sua posição no anel. Considerando o valor do ângulo β como sendo 30 graus (um valor próximo do encontrado em experimentos), pode-se concluir que, para distâncias de até 2 metros (a faixa de operação do sonar), em geral apenas dois sonares irão captar observações de uma mesma superfície plana. Então considerar cada sonar como um sensor independente do outro não implica em grande perda de informação.

As observações só serão consideradas enquanto o robô descreve uma trajetória retilínea. Assim são obtidas seqüências de observações cuja direção e sentido são fixos e a posição, variável. A figura 5.7 apresenta uma simulação de dois mapas de profundidades obtidos a partir de um sonar fixo num robô executando um movimento retilíneo. Dessa figura pode-se notar que as observações do sonar

Figura 5.6: Mapa de Profundidades Esperso obtido com doze sonares espaçados igualmente ao redor de um robô. Observe que mudando um pouco a inclinação do robô, as observações dos sonares podem mudar bastante. Obtido de Leonard e Durrant-Whyte (1992).

resultantes da parede formam um segmento de reta, enquanto que retornos originados de quinas e cantos geram uma pequena curva, similar a um arco, porém com curvatura oposta à dos arcos gerados nos mapas de profundidades de sonares rotatórios.

Para detectar as observações originadas de superfícies planas é considerado que observações coletadas durante um movimento retilíneo sejam colineares. Além de respostas de quinas e cantos não o serem, elas são obtidas apenas durante pequenos deslocamentos, e isso também é usado como forma de se rejeitar observações não desejadas. As observações provenientes dos cones secundários do pulso sonoro e outros efeitos não modelados dos sonares são rejeitadas considerando que essas observações são mais ruidosas, não aderindo tão bem a um segmento de reta.

5.4 Descrição do modelo de observação

Baseado nas idéias desenvolvidas até agora, o modelo de observação do sonar é proposto. Ele determina a observação que o sonar deveria fornecer para uma dada postura do robô. O modelo analisa um conjunto de observações do sonar capturadas até o instante atual e determina se é possível ou não gerar a observação esperada. Se for, o modelo a retorna para o estimador.

Para isso, primeiro é preciso identificar se o sonar está observando algum objeto válido (superfícies planas), e então determinar a postura do objeto em

Figura 5.7: Mapa de Profundidade Linear, obtido de observações de um sonar (cuja posição e orientação são indicadas por triângulos) em diversas posições sobre uma trajetória retilínea. a) Sonar observa uma parede: nessa situação as observações, representadas por pontos, encontram-se alinhadas de modo a formar um segmento de reta. b) Sonar observa um canto: as observações formam uma figura semelhante a um arco. Nessa representação as observações foram manipuladas para ressaltar o formato de arco, em geral mais sutil. Observações do sonar indicadas por um ‘X’ indicam que o sonar não recebeu resposta de uma leitura, o que significa que nenhum obstáculo se encontrava ao alcance do sonar.

relação à referência global (usando o mapa do ambiente). O modelo pode ser dividido em três etapas principais:

Validação das observações A primeira tarefa é determinar o que está sendo observado pelo sonar. Nesta etapa as observações e as respectivas estimativas da postura do robô são usadas para determinar, com o auxílio do mapa, se uma superfície plana está sendo observada; em caso positivo determinar uma estimativa para a postura da parede, em relação ao sonar, baseada na estimativa da postura do robô.

Associação das observações com o mapa Tendo obtido uma estimativa para o elemento observado, o mesmo é associado a algum elemento mapeado. Nesta etapa é realizada uma segunda validação, verificando se a última observação obtida pode ter sido originada pelo segmento mapeado.

Obtenção das observações esperadas Conhecendo a posição global do elemento observado, é obtida a observação esperada desse elemento a partir de uma estimativa da postura do robô.

As duas primeiras etapas são executadas até que algum elemento mapeado seja associado com sucesso às observações. Enquanto isso não ocorre, o modelo

de observação indica ao estimador que não é possível utilizar a observação do sonar para corrigir a postura do robô. Após uma associação ser realizada, as duas etapas entram numa fase de manutenção, enquanto a terceira passa a gerar as observações esperadas (valores $\bar{\mathbf{y}}_2(\mathbf{n})$, conforme mostrado na figura 3.1, necessárias para o estimador operar.

O modelo de observação trabalha com as observações e estimativas mais recentes da postura do robô. O conjunto de observações do sonar j é representado pelo vetor de dimensão k , $\mathbf{z}_j(\mathbf{n}) = [y_j(n), y_j(n-1), \dots, y_j(n-k+1)]$, onde $y_j(i)$ indica a observação do sonar j no instante i . O conjunto de estimativas da postura do robô é representada pela matriz de dimensões $3 \times k$ $\mathbf{X}(\mathbf{n}) = [\bar{\mathbf{x}}(\mathbf{n}|\mathbf{n}-1), \bar{\mathbf{x}}(\mathbf{n}-1|\mathbf{n}-2), \dots, \bar{\mathbf{x}}(\mathbf{n}-k+1|\mathbf{n}-k)]$, onde $\bar{\mathbf{x}}(\mathbf{i}|\mathbf{i}-1)$ indica a postura do robô prevista na iteração i ($\bar{\mathbf{x}}(\mathbf{i}) = [x(i), y(i), \theta(i)]^T$). Cada observação do sonar durante movimento retilíneo é armazenada juntamente com a postura estimada. Os vetores crescem até um certo tamanho, dado por k_{min} . A partir daí, cada nova observação inserida resulta no descarte da mais antiga. Os vetores são descartados quando é detectado um desvio de orientação no deslocamento do robô que indique movimento não retilíneo.

Para facilitar a notação, o índice j referente ao sonar nos vetores será descartado quando não houver risco de confusão. É importante notar como as seqüências $\mathbf{z}(\mathbf{n})$ e $\mathbf{X}(\mathbf{n})$ são formadas. As observações e posturas mais recentes são inseridas nas posições iniciais das seqüências. Então, quanto menor for o índice de um elemento nessas seqüências, mais atual terá sido sua ocorrência.

O número de observações e a distância entre as posturas em que as observações forem realizadas influenciam no resultado da etapa de validação da observação: quanto menor for o número e mais próximas entre si as observações, menos confiável será o resultado da validação; por outro lado, quanto maior o número e mais espaçadas, maior será o tempo sem correção na estimativa da postura e maior o percurso retilíneo que o robô precisará realizar. No entanto, o número de observações não pode ser muito grande, pois quanto maior o for, maior será a probabilidade de que o conjunto contenha observações provenientes de diferentes elementos do ambiente, diminuindo o número de observações validadas.

5.4.1 Operação

À medida que observações são coletadas, aumenta o número de observações, k . Enquanto $k < k_{min}$ a etapa de validação indicará que as observações são inválidas.

Quando $k = k_{min}$, a etapa de validação passa a ser executada, até que seja considerada plausível a hipótese de que as observações advenham de uma mesma superfície plana. Uma vez validadas as observações, um modelo da superfície observada é obtido, o qual é usado na etapa de associação para encontrar elementos mapeados que possam ser o elemento observado. As etapas de validação e associação são executadas até que um elemento mapeado possa ser associado ao conjunto de observações. A partir desse momento, a observação esperada passa a ser calculada.

A partir da iteração em que um elemento mapeado é associado às observações, a etapa de associação pára de buscar uma associação entre as observações e um elemento mapeado sendo a partir daí somente verificado se as observações podem ter sido originadas pelo último elemento mapeado selecionado. O último elemento mapeado é mantido até que, em algum momento, a etapa de validação o rejeite.

5.4.2 Validação das observações

Conforme já explicado, esta etapa só passa a ser executada quando k_{min} observações estão disponíveis. Caso contrário é indicado que as observações não são válidas. Nesta etapa é determinado se as observações correspondem a uma superfície plana através de um teste que se aproveita da propriedade apresentada na figura 5.7: as observações tendem a se alinhar num segmento de reta caso uma superfície plana esteja sendo observada.

Na figura 5.8 é mostrado o robô se deslocando em movimento retilíneo observando uma parede em três momentos distintos, e as observações do sonar são representadas por uma linha traçada de modo a formar um ângulo reto com a parede observada. Entre cada uma das três observações da parede, $y^{(1)}$, $y^{(2)}$ e $y^{(3)}$, o robô percorreu uma certa distância: entre as observações $y^{(2)}$ e $y^{(1)}$, o robô percorreu $d_{2,1}$, e entre $y^{(3)}$ e $y^{(2)}$, o robô percorreu $d_{3,2}$. Considerando essas grandezas e observando os triângulos formados por elas, pode-se estabelecer a seguinte relação, com $y^{(j)} - y^{(i)} = y_{j,i}$:

$$\frac{y_{2,1}}{d_{2,1}} = \frac{y_{3,2}}{d_{3,2}} \quad (5.1)$$

É interessante notar que a relação é válida não importando a inclinação α entre a trajetória do robô e a parede, desde que o sonar esteja numa inclinação adequada para observar a parede. Essa relação pode ser estendida para as k

Figura 5.8: Representação de observações do sonar em diferentes posições ao longo de um movimento retilíneo; $y^{(1)}$, $y^{(2)}$ e $y^{(3)}$ representam as observações de um sonar do robô (sendo $y^{(1)}$ a mais recente), espaçadas entre si por $d_{2,1}$ e $d_{3,2}$, respectivamente.

observações consideradas, resultando em:

$$\frac{y_{2,1}}{d_{2,1}} = \frac{y_{3,2}}{d_{3,2}} = \dots = \frac{y_{k,k-1}}{d_{k,k-1}}$$

e considerando:

$$\frac{y_{j,j-1}}{d_{j,j-1}} = \gamma_{j,j-1},$$

para a realização do teste é formado um conjunto de $k - 1$ elementos $\gamma_{j,j-1}$, com $j > 1$. Como esse conjunto é formado por variáveis que seguem distribuições de probabilidade gaussianas e independentes conhecidas (as observações e o deslocamento do robô), é possível determinar a variância de cada elemento do conjunto. Assumindo que $d_{j,j-1}$ e $d_{i,i-1}$ não variam muito para quaisquer i e j , $i \neq j$, então pode-se dizer que a variância dos $k - 1$ elementos $\gamma_{j,j-1}$ tende a ser próxima. Levando a aproximação um passo adiante, e assumindo que a hipótese de que um único segmento de reta está sendo observado é verdadeira, pode-se considerar que os $k - 1$ elementos $\gamma_{j,j-1}$ são amostras de uma mesma distribuição.

O teste consiste em verificar se é verdadeira a hipótese que a variância das $k - 1$ amostras, dada por s_{γ}^2 , é menor ou igual à variância da distribuição, representada por $\sigma_{obsmedia}^2$. Um teste simples de variância, apresentado por Costa (1977), é usado, cuja formalização é feita a seguir. Apresentam-se duas hipóteses sobre a distribuição de probabilidade de $\gamma_{j,j-1}$, H_0 e H_1 :

$$H_0 : \gamma \sim N(A, \sigma_{obsmedia}^2) \quad (5.2)$$

$$H_1 : \gamma \sim N(A, \sigma_{alternativo}^2), \quad \sigma_{obsmedia}^2 < \sigma_{alternativo}^2.$$

O teste é realizado sobre a estatística χ^2 :

$$\chi_{k-2}^2 = \frac{(k-2)s_\gamma^2}{\sigma_{obsmedia}^2}. \quad (5.3)$$

Para aceitar H_0 , é verificado se χ^2 é menor que um dado limite:

$$\chi_{k-2}^2 < \chi_{k-2,1-\alpha}^2,$$

onde $\chi_{k-2,\alpha}^2$ é um valor tabelado da estatística χ_{k-2}^2 e α indica a confiança considerada no teste.

O teste é realizado sobre todas as observações coletadas. Caso o teste resulte em sucesso, então as observações são validadas. Caso o teste resulte em fracasso, as observações são consideradas inválidas, e caso algum elemento do mapa esteja associado a essas observações, a associação é desfeita.

Determinação de s_γ^2

A média e a variância do conjunto de elementos γ são dadas por:

$$\bar{\gamma} = \frac{\sum_{i=1}^{k-1} \gamma_{i,i-1}}{k-2}$$

e

$$s_\gamma^2 = \frac{\sum_{i=1}^{k-1} (\gamma_{i,i-1} - \bar{\gamma})^2}{k-2}.$$

Determinação de $\sigma_{obsmedia}^2$

Considerando que todos $\gamma_{j,j-1}$ seguem uma mesma *pdf* gaussiana, $\sigma_{obsmedia}^2$ é a variância dessa distribuição. Para determiná-la, cada $\gamma_{j,j-1}$ é tratado como uma variável aleatória, e sua variância é obtida a partir das variáveis que a compõem, $d_{j,j-1}$ e $y_{j,j-1}$. Essas variáveis, por sua vez, seguem *pdfs* que dependem de outras variáveis cujas *pdfs* têm aproximações conhecidas. Assim, antes de $\sigma_{obsmedia}^2$, são determinadas as variâncias a partir de $d_{j,j-1}$ e $y_{j,j-1}$.

Durante as deduções a seguir é assumido que os todos os $d_{j,j-1}$ têm o mesmo valor dado por $\frac{d_{k,1}}{k-1}$. Essa simplificação permite expressar as variâncias por equações mais simples e se justifica pelos seguintes motivos: i) em geral os valores dos $d_{j,j-1}$ são próximos entre si pois as observações dos sonares são realizadas numa frequência constante e o robô não tende a variar a velocidade enquanto se desloca em movimento retilíneo; ii) pode-se observar que $\gamma_{j,j-1}$ varia inversamente com

$d_{j,j-1}$, e a variância de $d_{j,j-1}$ aumenta conforme aumenta $d_{j,j-1}$. Então, quanto maior $d_{j,j-1}$, maior sua variância, porém menor será sua influência sobre $\gamma_{j,j-1}$, o que faz com que a variância de $\gamma_{j,j-1}$ seja menos sensível às variações de $d_{j,j-1}$.

O valor médio de $\gamma_{j,j-1}$ é dado por:

$$\bar{\gamma}_{j,j-1} = \frac{1}{k-1} \sum_{i=2}^k \frac{y_{i,i-1}}{d_{i,i-1}} = \frac{1}{k-1} \sum_{i=2}^k \frac{y_{i,i-1}}{\frac{d_{k,1}}{k-1}} = \frac{y_{k,1}}{d_{k,1}}, \quad (5.4)$$

onde $\bar{\gamma}_{j,j-1}$ indica a estimativa da média de uma distribuição representada pelas $k-1$ amostras $\gamma_{j,j-1}$. Como se pode notar, o valor médio depende apenas das primeiras e últimas observações consideradas.

Pelo modelo, é considerado que durante um segmento de deslocamento não há mudança de orientação, então a única incerteza considerada para $d_{k,1}$ é relativa à distância percorrida no trajeto, sem levar em conta a incerteza relativa à orientação. Eliminar essa fonte de incerteza faz com que o deslocamento seja considerado menos ruidoso, e a validação das observações se torna mais restritiva. A incerteza de $d_{k,1}$ é dada então por $(\sigma_{dx_{k,1}}^2 + \sigma_{dy_{k,1}}^2)$, onde $\sigma_{dx_{k,1}}^2$ e $\sigma_{dy_{k,1}}^2$ indicam as incertezas do deslocamento do robô nos eixos x e y , respectivamente.

A incerteza de $y_{j,j-1}$ é dada pela soma da incerteza de seus componentes, $y^{(j)}$ e $y^{(j-1)}$. As incertezas desses componentes são dadas pela incerteza da própria observação, e pela incerteza do deslocamento do robô entre as posições consideradas. Como o deslocamento não apresenta erro de orientação, apenas a componente do deslocamento na mesma direção da reta representando a observação na figura 5.8 afeta a observação. Por exemplo, para a observação $y^{(2)}$, a componente da incerteza de $d_{2,1}$ que a afeta é dada por $d_{2,1} \cdot \text{sen}(\alpha)$, onde α indica a inclinação entre a parede e o deslocamento do robô, conforme a figura 5.8.

Para determinar α , pode-se fazer uma aproximação considerando como verdadeira a hipótese de que uma superfície plana esteja sendo observada. Então, pode-se desenhar um triângulo retângulo cuja hipotenusa é $d_{k,1}$ e um dos catetos é $|y_{k,1}|$. O ângulo oposto a esse cateto é α .

A incerteza de $y^{(i)}$ é dada por $\sigma_{r_i}^2 = \sigma_{obs}^2 + \sigma_{d_{i,1}}^2 \text{sen}^2(\alpha)$, onde σ_{obs}^2 é a incerteza da observação do sonar e $\sigma_{d_{i,1}}^2$ é a incerteza do deslocamento entre a posição i e a primeira posição. Considerando que a incerteza $\sigma_{d_{j,1}}^2$ pode ser decomposta em

$\sigma_{d_{j,j-1}}^2 + \sigma_{d_{j-1,1}}^2$, a incerteza de $y_{j,j-1}$ pode ser calculada da seguinte forma:

$$\begin{aligned}
 \sigma_{y_{j,j-1}}^2 &= E_y[y_{j,j-1}^2] \\
 &= E_y[(y^{(j)} - y^{(j-1)})^2] \\
 &= E_y[(y^{(j)})^2 + (y^{(j-1)})^2 - 2y^{(j)}y^{(j-1)}] \\
 &= E_y[(y^{(j)})^2] + E_y[(y^{(j-1)})^2] - E_y[2y^{(j)}y^{(j-1)}] \\
 &= \sigma_{obs}^2 + \sigma_{d_{j,1}}^2 \text{sen}^2(\alpha) + \sigma_{obs}^2 + \sigma_{d_{j-1,1}}^2 \text{sen}^2(\alpha) - E_y[2y^{(j)}y^{(j-1)}] \\
 &= 2\sigma_{obs}^2 + \sigma_{d_{j,j-1}}^2 \text{sen}^2(\alpha) + 2\sigma_{d_{j-1,1}}^2 \text{sen}^2(\alpha) - 2E_y[y^{(j)}y^{(j-1)}].
 \end{aligned}$$

Considerando que as observações em instantes distintos não são correlacionadas (apesar de seguirem a mesma distribuição) e levando em conta a decomposição da distância, a expansão de $E_y[y^{(j)}y^{(j-1)}]$ resulta em:

$$E_y[y^{(j)}y^{(j-1)}] = \sigma_{d_{j-1,1}}^2 \text{sen}^2(\alpha),$$

e assim:

$$\sigma_{y_{j,j-1}}^2 = 2\sigma_{obs}^2 + \sigma_{d_{j,j-1}}^2 \text{sen}^2(\alpha).$$

Conhecendo as variâncias de $d_{j,j-1}$ e $y_{j,j-1}$, o próximo passo é a determinação da variância de $\gamma_{j,j-1}$.

Como $\gamma_{j,j-1}$ é uma função não-linear de variáveis aleatórias gaussianas, é adotado o mesmo método usado pelo EKF para linearizar a função para, a partir da função linearizada, determinar a sua variância. O primeiro passo é determinar a matriz de covariância das variáveis que compõem $\gamma_{j,j-1}$.

Considerando o vetor $\omega_{j,j-1}$ contendo as variáveis que compõem $\gamma_{j,j-1}$, como:

$$\omega_{j,j-1} = \left(\frac{d_{k,1}}{k-1} \quad y_{j,j-1} \right)^T,$$

pode-se definir a matriz de covariância $\Sigma\omega_{j,j-1}$ como:

$$\Sigma\omega_{j,j-1} = \begin{pmatrix} \frac{\sigma_{d_{k,1}}^2}{(k-1)^2} & E_y[y_{j,j-1} \frac{d_{k,1}}{k-1}] \\ E_y[y_{j,j-1} \frac{d_{k,1}}{k-1}] & \sigma_{y_{j,j-1}}^2 \end{pmatrix}, \quad (5.5)$$

onde $\sigma_{y_{j,j-1}}^2$ não varia com j , pois como é assumido que $d_{j,j-1} = \frac{d_{k,1}}{k-1}$, então $\sigma_{y_{j,j-1}}^2 = 2\sigma_{obs}^2 + \frac{\sigma_{d_{k,1}}^2}{(k-1)^2} \text{sen}^2(\alpha)$. O valor de $E_y[y_{j,j-1} \frac{d_{k,1}}{k-1}]$ é determinado da seguinte forma:

$$\begin{aligned}
E_y[y_{j,j-1} \frac{d_{k,1}}{k-1}] &= \frac{E_y[(y^{(j)} - y^{(j-1)})d_{k,1}]}{k-1} \\
&= \frac{E_y[y^{(j)}d_{k,1}] - E_y[y^{(j-1)}d_{k,1}]}{k-1} \\
&= \frac{E_y[y^{(j)}(d_{k,j} + d_{j,1})] - E_y[y^{(j-1)}(d_{k,j-1} + d_{j-1,1})]}{k-1} \\
&= \frac{E_y[y^{(j)}d_{j,1}] - E_y[y^{(j-1)}d_{j-1,1}]}{k-1} \\
&= \frac{\sigma_{d_{j,1}}^2 \text{sen}(\alpha) - \sigma_{d_{j-1,1}}^2 \text{sen}(\alpha)}{k-1} \\
&= \frac{\sigma_{d_{j,j-1}}^2 \text{sen}(\alpha)}{k-1} \\
&= \frac{\sigma_{d_{k,1}}^2 \text{sen}(\alpha)}{(k-1)^2}
\end{aligned}$$

A linearização de $\gamma_{j,j-1}$ é realizada sobre o valor médio de $\gamma_{j,j-1}$ dado pela equação 5.4 e que será referenciado por $\gamma(\bar{\omega}_{j,j-1})$. O resultado é:

$$\begin{aligned}
\gamma_{j,j-1} &= \gamma(\omega_{j,j-1}) = \frac{(k-1)y_{j,j-1}}{d_{k,1}}, \\
\gamma(\omega_{j,j-1}) &= \gamma(\bar{\omega}_{j,j-1}) + \mathbf{F}_\gamma(\omega_{j,j-1} - \bar{\omega}_{j,j-1}),
\end{aligned}$$

sendo:

$$\begin{aligned}
\mathbf{F}_\gamma &= \frac{\partial \gamma(\omega_{j,j-1})}{\partial \omega_{j,j-1}} |_{\omega_{j,j-1} = \bar{\omega}_{j,j-1}} \\
&= \left(-\frac{(k-1)y_{j,j-1}}{d_{k,1}^2} \quad \frac{k-1}{d_{k,1}} \right) |_{\omega_{j,j-1} = \bar{\omega}_{j,j-1}} \quad (5.6)
\end{aligned}$$

Calculando a covariância da função linearizada, é obtida a variância desejada:

$$\begin{aligned}
\sigma_\gamma^2 &= E_y[\gamma(\omega_{j,j-1})\gamma(\omega_{j,j-1})^T] \\
&= E_y[(\mathbf{F}_\gamma \omega_{j,j-1})(\mathbf{F}_\gamma \omega_{j,j-1})^T] \\
&= \mathbf{F}_\gamma E_y[\omega_{j,j-1} \omega_{j,j-1}^T] \mathbf{F}_\gamma^T \\
&= \mathbf{F}_\gamma \Sigma \omega_{j,j-1} \mathbf{F}_\gamma^T.
\end{aligned}$$

Como dito no começo da seção, $\sigma_{obsmedia}^2$ é a variância da variável aleatória $\gamma_{j,j-1}$. Então:

$$\sigma_{obsmedia}^2 = \sigma_\gamma^2 = \mathbf{F}_\gamma \Sigma \omega_{j,j-1} \mathbf{F}_\gamma^T.$$

Determinação da parede observada

Quando um conjunto de observações for validado, é necessário determinar uma representação do plano que está sendo observado. Os planos são representados num espaço bidimensional por retas infinitas. Cada reta é representada por dois parâmetros: a sua distância à origem do sistema global de coordenadas (distância da parede à origem), r_{wall} , e a inclinação do segmento que une a reta ao seu ponto mais próximo à origem, θ_{wall} .

Para estimar θ_{wall} , parte-se da orientação do robô durante o deslocamento (invariante, segundo a aproximação considerada), e da estimativa de α . Já r_{wall} é estimado a partir do valor da primeira observação considerada válida, $r^{(1)}$, e da posição do sonar ao realizar essa observação, $(x_{sonar}^{(1)}, y_{sonar}^{(1)})$. A posição do sonar pode ser obtida a partir da postura do robô $(x^{(1)}, y^{(1)}, \theta^{(1)})$ conforme:

$$\begin{pmatrix} x_{sonar}^{(1)} \\ y_{sonar}^{(1)} \end{pmatrix} = \begin{pmatrix} x^{(1)} \\ y^{(1)} \end{pmatrix} + \begin{pmatrix} \cos(\theta^{(1)}) & -\sin(\theta^{(1)}) \\ \sin(\theta^{(1)}) & \cos(\theta^{(1)}) \end{pmatrix} \begin{pmatrix} x'_{sonar} \\ y'_{sonar} \end{pmatrix}, \quad (5.7)$$

onde x'_{sonar} e y'_{sonar} indicam a posição do sonar em relação à frente do robô.

A partir dos dados apresentados, os parâmetros da reta observada são obtidos da seguinte forma:

$$\theta_{wall} = \theta^{(1)} + \alpha \quad (5.8)$$

e

$$r_{wall} = r^{(1)} + x_{sonar}^{(1)} \cos(\theta_{wall}) + y_{sonar}^{(1)} \sin(\theta_{wall}). \quad (5.9)$$

5.4.3 Associação das observações com o mapa

Usando a estimativa do plano sendo observado obtida pelas equações 5.8 e 5.9, consulta-se o mapa para determinar quais retas mapeadas mais se aproximam da estimativa. Essa é uma etapa crítica, pois caso a associação realizada esteja errada, erros grosseiros podem ser gerados pelo modelo de observação, podendo fazer a estimativa da postura do robô divergir do valor real.

Para minimizar erros de associação, os procedimentos aplicados tendem a obter mais falsos negativos do que falsos positivos. Se não for possível determinar, entre um grupo de elementos mapeados, qual deve ser associado às observações, a etapa retorna que nenhum segmento foi associado.

A partir de todos os segmentos mapeados, uma sequência de filtragens é apli-

cada de modo que apenas um segmento não seja eliminado. A primeira filtragem é realizada pela inclinação. São excluídos os segmentos mapeados que não poderiam ser observados pelo sonar, considerando a faixa de inclinações que o cone principal do pulso sonoro alcança. A faixa, de tamanho β , é ampliada levando em conta a incerteza da inclinação do robô, aumentando a faixa em $2\sigma_{\theta(1)}$, onde $\sigma_{\theta(1)}$ é o desvio-padrão de $\theta^{(1)}$ (orientação do robô na primeira posição). A faixa de inclinações então fica com tamanho $\beta + 2\sigma_{\theta(1)}$, centrada na estimativa da inclinação do sonar, que é dada somando à inclinação do robô a inclinação do sonar em relação à frente do robô.

Em seguida, é determinado se o segmento está dentro do alcance do sonar levando em conta o tamanho dos segmentos mapeados. Para cada segmento é considerada uma reta infinita colinear ao mesmo. A partir dessa reta, é determinada uma segunda reta perpendicular a ela e que toca o sonar. O segmento mapeado é descartado caso não haja intersecção com essa segunda reta. Esse teste é realizado para a primeira e a última posição do robô consideradas na etapa de validação.

Finalmente, os segmentos restantes são testados para determinar quais deles poderiam originar as observações. A primeira, a penúltima e a última observações do sonar consideradas válidas na etapa de validação são usadas. É aplicado um teste sobre a diferença entre a observação obtida do sonar $r^{(i)}$ e a observação esperada calculada, denominada de $h(\cdot)$, para cada segmento (o cálculo de $h(\cdot)$ será apresentado na seção 5.4.4). As medidas usadas para um elemento i dos conjuntos de observações e posturas são:

$$\begin{aligned} v^{(i)} &= r^{(i)} - h(\mathbf{x}^{(i)}) \\ (s^{(i)})^2 &= \mathbf{H}\mathbf{P}_{\mathbf{x}}^{(i)}\mathbf{H}^T + \sigma_{obs}^2, \end{aligned} \quad (5.10)$$

onde $\mathbf{H}$ é a derivada da função $h(\cdot)$ em relação à postura, necessária para operação do EKF ($\mathbf{H}$ é determinada na seção 5.4.4) e $(s^{(i)})^2$ é a variância da diferença $v^{(i)}$.

O teste realizado é:

$$\frac{(v^{(i)})^2}{(s^{(i)})^2} < g^2, \quad (5.11)$$

onde g^2 é o valor limite para a aceitação da observação. Esse teste também é conhecido na literatura como *validation gate*, ou porta de validação (LEONARD; DURRANT-WHYTE, 1991), e é usado para rejeitar observações que potencialmente não são corretas, por apresentarem grande divergência do esperado.

Caso mais de um ou nenhum segmento mapeado passe por essa sequência de

filtros, a etapa de associação indica que nenhuma associação foi possível.

Discussão

Para refinar mais a associação em caso de ambigüidade (mais de um elemento com possibilidades de ser associado com as observações) pode-se considerar também as potenciais associações dos demais sonares como um modo de localizar melhor o robô no ambiente.

Outro modo de melhorar a associação seria considerar a última associação realizada pelo próprio sonar. Caso uma associação seja perdida e recuperada num mesmo movimento retilíneo, por exemplo, pode-se tentar inferir se a perda da associação foi causada por uma descontinuidade da superfície sendo observada (porta aberta, estreitamento) ou por um objeto desconhecido (pessoa se movendo, por exemplo), e a partir do motivo mais provável restringir mais os segmentos associáveis.

Essas melhorias não foram consideradas no trabalho corrente e ficam como sugestão para trabalhos futuros.

5.4.4 Obtenção das observações esperadas para a estimativa da postura

Seguindo as etapas de validação e associação, a resposta do modelo de observação, $h(\mathbf{x})$, é obtida a partir da estimativa da postura do robô na iteração atual e do segmento mapeado associado às observações.

É usado o mesmo procedimento adotado por Leonard e Durrant-Whyte (1992) para calcular a observação esperada do sonar. A figura 5.9 mostra um sonar do robô observando uma parede. Pela figura, pode-se determinar a observação esperada da seguinte forma:

$$\bar{y}_{obs} = r_{wall} - (x_{sonar} \cos(\theta_{wall}) + y_{sonar} \sin(\theta_{wall})),$$

onde y_{obs} é a observação esperada para o sonar, x_{sonar} e y_{sonar} indicam a posição do sonar. Recuperando-se o índice temporal n na notação, $\bar{y}_{obs}(n)$, dado pela função $h(\cdot)$, pode ser escrita por:

$$h(\bar{\mathbf{x}}(n)) = r_{wall} - (x_{sonar}(n) \cos(\theta_{wall}) + y_{sonar}(n) \sin(\theta_{wall})) \quad (5.12)$$

$$h(\mathbf{x}(\mathbf{n})) = r_{wall} - (x(n) + (\cos(\theta(n))x'_{sonar} - \sin(\theta(n))y'_{sonar})\cos(\theta_{wall}) + (y(n) + \sin(\theta(n))x'_{sonar} + \cos(\theta(n))y'_{sonar})\sin(\theta_{wall})), \quad (5.13)$$

The diagram shows a robot (Robô) in a 2D environment. A wall (Parede Observada) is represented by a solid line at the top. A coordinate system (X, Y) is established at the bottom left. The robot's position is defined by its distance from the wall (y) and its orientation relative to the wall's normal (θ_{sonar}). The wall's orientation is given by θ_{wall} - 90°. A sonar sensor is shown emitting a beam towards the wall, with its orientation also indicated by θ_{sonar}. The distance from the robot to the wall is labeled y, and the distance from the robot to the sonar sensor is labeled r_{wall}.

Como pode ser visto no apêndice II, seção II.5.1, o Filtro de Kalman Estendido necessita da matriz $\mathbf{H}(\mathbf{n})$, obtida derivando a função descrita em 5.12 em relação à postura, para propagar a incerteza e obter o Ganho de Kalman, sendo $\mathbf{H}(\mathbf{n})$ dado por:

$$\mathbf{H}(\mathbf{n}) = \begin{pmatrix} -\cos(\theta_{wall}) \\ -\sin(\theta_{wall}) \\ x'_{sonar}\sin(\theta(n) - \theta_{wall}) + y'_{sonar}\cos(\theta(n) - \theta_{wall}) \end{pmatrix}^T. \quad (5.14)$$

A seguir o valor dos parâmetros adotados no modelo de observação são apresentados.

Determinação de β e de σ_{obs}^2

Para determinar o valor de β , o robô foi posicionado em frente a uma parede a uma distância de $1,2m$, de modo que sua frente ficasse alinhada a uma reta perpendicular à parede. Essa inclinação foi variada para ambos os lados em passos de $2,5$ graus, e, para cada inclinação, 100 observações foram realizadas. Para cada posição foi calculado o valor médio, a variância e o valor esperado das observações. Para cada direção que o robô foi inclinado foi determinada a inclinação anterior à inclinação em que foi detectado um incremento grande na variância das observações e na discrepância entre a média e a observação esperada.

A diferença angular entre as duas inclinações determinadas resulta no valor adotado para a faixa de atuação do sonar, ou seja, β . O valor obtido para β , com esses experimentos, foi de $27,5$ graus.

O valor de σ_{obs}^2 foi determinado a partir de todas as observações usadas no experimento acima que se encontravam nas inclinações válidas (dentro da faixa delimitada por β). O valor encontrado de σ_{obs}^2 foi de 625 mm^2 .

Quantidade mínima de observações (k_{min})

O número mínimo de observações para operar a validação, k_{min} , depende de dois fatores: i) a distância percorrida entre as observações do sonar, e ii) a quantidade de observações necessárias para os algoritmos operarem.

A quantidade mínima de observações necessárias adotada para os algoritmos de validação operarem são quatro. Apesar de com apenas três observações serem obtidos dois γs (que é o número mínimo para se calcular a variância de um conjunto), um conjunto de amostras com dois elementos praticamente não tem valor estatístico. Com três γs já é possível determinar com algum grau de confiança se o que está sendo observado é ou não uma superfície plana. À medida que aumenta o número de observações usadas, aumenta a confiança com que se rejeita ou aceita um conjunto.

O deslocamento mínimo necessário entre observações influi no algoritmo de validação: quanto maior o deslocamento, maior a distância entre a primeira e última observações consideradas, e melhor a validação, considerando um erro de odometria pequeno para pequenos deslocamentos. Como nos experimentos realizados o robô se deslocou em velocidade constante, o aumento da distância entre observações implica diretamente no aumento do número de observações consideradas.

Por outro lado, à medida que aumenta o deslocamento do robô considerado, superfícies planas menores (como faces de colunas) passam a ser ignoradas. Além disso, um atraso maior será imposto entre o momento que uma sequência de observações válidas começa a ser coletada e o momento em que correções da postura começam a serem geradas.

Através de simulações com seqüências de observações coletadas com sonar real observando uma parede enquanto se desloca em rotas paralelas à mesma, observou-se que as observações dos sonares nas condições consideradas pelo modelo (observando uma superfície plana dentro do ângulo de visão β do sonar) são bem estáveis, oscilando em torno de um valor médio conforme o modelo prevê. Chegou-se à conclusão que a distância percorrida pelo robô entre a primeira e a última observações consideradas é de fato mais importante que o número de observações consideradas para o algoritmo de validação, e foi determinado que a partir de 25 cm já é possível realizar validações que eliminem mais de 95% das observações inválidas.

Assim, o número mínimo de observações é dado pelo maior valor entre quatro e o número de observações capturadas durante um deslocamento de 25 cm.

Durante os experimentos com o robô se deslocando, porém, devido à velocidade do mesmo, as observações de um mesmo sonar foram capturadas em média a cada 20 cm, o que torna dominante a restrição de efetuar pelo menos quatro observações. Na prática foi obtido um deslocamento médio de 80 cm entre as observações coletadas. Essa distância é grande o bastante para que os sonares não detectem as colunas presentes no ambiente onde foram realizados os experimentos.

Limite de rejeição das observações ($\chi^2_{k-2,1-\alpha}$)

Em geral o valor de rejeição de testes estatísticos são escolhidos com faixas de aceitação muito estreitas, com alta rejeição, com o objetivo de aceitar apenas o que se tem grande confiança que obedeça a uma dada hipótese. A primeira validação imposta às observações é construída em cima de um modelo aproximado que desconsidera parte das incertezas modeladas. Considerando isso e analisando os valores obtidos em simulações e experimentos, foi adotado um grau de confiança de 50 %, bem abaixo do que é encontrado na literatura estatística, que em geral é de 80 % ou mais.

Em experimentos esse valor se mostrou adequado para filtrar observações indesejadas e aceitar as observações desejadas.

Limite de rejeição de uma observação (g^2)

O limite de rejeição da equação 5.11 foi também determinado experimentalmente, de modo a aumentar o rigor da primeira validação. O limite adotado foi de 3,2.

5.6 Resumo do Capítulo

Para cada observação coletada de um dos sonares ($y(n)$), determina-se a observação esperada do sonar ($\bar{y}(n)$) a partir do mapa do ambiente (dado por uma coleção de segmentos de retas descritos por suas distâncias à origem, r_{wall} , e suas inclinações, θ_{wall}) e da estimativa predita da postura do robô ($\bar{\mathbf{x}}(\mathbf{n}|\mathbf{n}-1) = [x(n|\mathbf{n}-1), y(n|\mathbf{n}-1), \theta(n|\mathbf{n}-1)]^T$) do seguinte modo:

$$\begin{aligned} \bar{y}_{obs}(n) = & r_{wall} - (x(n) + (\cos(\theta(n|\mathbf{n}-1))x'_{sonar} - \sin(\theta(n|\mathbf{n}-1))y'_{sonar})\cos(\theta_{wall}) + \\ & (y(n) + \sin(\theta(n|\mathbf{n}-1))x'_{sonar} + \cos(\theta(n|\mathbf{n}-1))y'_{sonar})\sin(\theta_{wall})), \end{aligned}$$

onde x'_{sonar} e y'_{sonar} indicam a posição do sonar em relação à frente do robô.

A observação realizada e a observação esperada são usadas então para corrigir o valor da postura predita do robô ($\bar{\mathbf{x}}(\mathbf{n}|\mathbf{n}-1)$), obtendo o valor final da estimativa da postura ($\bar{\mathbf{x}}(\mathbf{n})$).

Para determinar qual e se foi um dos segmentos mapeados o responsável pela observação realizada do sonar é aplicado o procedimento de teste descrito no capítulo. O resultado final é uma indicação que se deve ou não realizar a correção da postura do robô, e em caso positivo o procedimento também fornece os parâmetros r_{wall} e θ_{wall} do elemento observado.

6 Modelo de Observação: Visão Monocular

O sensor de visão monocular é composto por uma câmera montada sobre o robô, apontada na direção frontal do robô, posicionada conforme já mostrado na figura 3.2.

Obter a localização a partir de uma câmera de vídeo é muito interessante, pois a câmera consegue captar quase os mesmos estímulos visuais que os seres humanos, e por ser esse um dos principais sentidos humanos, é esperado que, em ambientes criados pelo homem, seja encontrada grande quantidade de estímulos que podem ser usados para a localização. Apesar disso, obter a localização a partir da visão computacional é um desafio. As principais dificuldades são as seguintes:

- Grande volume de informação: uma câmera de vídeo produz um grande fluxo de dados, sendo necessário grande poder computacional para processar todas as informações fornecidas em tempo real; além do mais, grande parte dos dados em geral não tem utilidade para a localização ou apresentam informação redundante.
- Interferência entre os estímulos visuais: causada também pelo grande volume de informação, interferências geradas por alguns elementos na observação de outros podem ocorrer, como oclusões (um elemento ocultando outro temporariamente) ou alterações nas condições de iluminação do ambiente através de sombras ou reflexão.
- Falta de informação de profundidade: a profundidade, ou distância que os objetos observados se encontram da câmera, não pode ser obtida diretamente da visão monocular; sendo a visão monocular usada para determinar a postura do robô, é necessário determinar sua postura relativa a elementos conhecidos do ambiente e, sem a informação de profundidade, este processo é dificultado.

Uma outra dificuldade num sistema de localização que deve operar numa plataforma computacional de baixo custo é manter baixa a taxa de uso do processador, pois provavelmente outros subsistemas, como navegação e planejamento de rota, por exemplo, compartilham o mesmo processador.

O processamento dos quadros de vídeo, por mais simples que seja, é o gargalo de desempenho do estimador, devido ao grande volume de dados. Como a taxa de geração de quadros das câmeras é alta, fornecendo em geral de 15 a 30 quadros por segundo, mesmo que o estimador seja capaz de manter o tempo de resposta baixo, o processamento das imagens pode acabar consumindo quase toda capacidade de processamento disponível por grandes períodos, interferindo na execução correta dos demais sistemas.

Considerando que o robô se desloca vagarosamente, a $0,3\text{ m/s}$, por exemplo, tratar quadros a uma taxa de 30 quadros/s pode não apresentar uma boa relação entre o ganho obtido na postura do robô e o custo necessário de processamento. Em geral, quanto menor o tempo decorrido entre quadros processados, menor a distância percorrida pelo robô, menor o erro acumulado na postura do robô e menor a importância dada para a correção do sensor de visão durante a correção da postura do robô (lembrando que a importância dada para as correções é tão maior quanto mais precisa ela for em relação à estimativa já existente).

A solução para aumentar a taxa de informação passada da observação para a localização do robô por quadro processado é processar um novo quadro apenas após uma distância mínima (tanto linear como angular) ter sido percorrida. Entretanto, a distância percorrida entre quadros processados deve ficar limitada às necessidades do algoritmo de observar um mesmo conjunto de elementos num número mínimo de quadros consecutivos.

A distância linear mínima entre quadros adotada é representada por $d_{captura}$ e a distância angular mínima é representada por $\theta_{captura}$. Os valores reais para essas distâncias são obtidos experimentalmente.

O conhecimento do modelo de projeção adotado e de conceitos básicos relacionados à formação das projeções na câmera são importantes na definição da abordagem adotada e serão descritos a seguir.

6.1 Modelo de Câmera

A câmera utilizada neste trabalho é do tipo CCD (MA, 2004), um dispositivo de aquisição largamente usado em câmeras de baixo custo. O modelo de câmera

é baseado no modelo de câmera de orifício, ou *pinhole*. Segundo esse modelo, a câmera é constituída por um centro focal C e por um plano de projeção, dispostos conforme exibido na figura 6.1. A reta perpendicular ao plano de projeção e que contém o centro focal é denominada de eixo principal da câmera. As imagens são formadas no plano de projeção, e a projeção de cada elemento do ambiente é formada pelas intersecções entre o plano de projeção e as retas que intersectam o elemento e contém o centro focal C . Essas retas representam a luz refletida (ou emitida) pelo elemento e captada pela câmera. Caso dois pontos distintos do ambiente gerem a mesma reta ligando-os ao centro focal, diz-se que um ponto está obliterando o outro e apenas o mais próximo à câmera será registrado (desconsiderando casos de transparência).

O plano de projeção é um espaço discreto, formado por um reticulado onde cada unidade do reticulado pode assumir um único valor de cada vez. O tamanho do reticulado indica a resolução da câmera, e as células do reticulado são denominadas de pixels. Apesar do plano de projeção em geral não ser quadrado, e sim retangular, os pixels quase sempre podem ser representados por quadrados, e é assumido neste trabalho que eles são quadrados.

A câmera utilizada possibilita a captação de imagens em cores. A representação das cores é feita por três bandas distintas de cores: vermelho, verde e azul. A combinação das intensidades das cores de cada banda fornece a imagem colorida.

É considerado o sistema de coordenadas (X_C, Y_C, Z_C) intrínseco à câmera, alinhado conforme mostra a figura 6.1. Nesse sistema um ponto no espaço representado por $\mathbf{p}_{\text{cam}} = [x_{\text{cam}}, y_{\text{cam}}, z_{\text{cam}}]^T$ tem sua projeção no plano de projeção da câmera $\mathbf{p}_{\text{proj}} = [u, v]^T$ obtida da seguinte forma:

$$\begin{aligned} u &= f_u \frac{y_{\text{cam}}}{x_{\text{cam}}} + u_0 \\ v &= f_v \frac{z_{\text{cam}}}{x_{\text{cam}}} + v_0, \end{aligned} \tag{6.1}$$

onde u e v são as coordenadas do ponto projetado no plano de projeção no sistema de coordenadas (U, V) intrínseco ao plano de projeção. Os valores de u e v são dados em pixels. f_u e f_v indicam a distância focal multiplicada por um fator de conversão pixel/metro para as direções das coordenadas u e v , respectivamente, e u_0 e v_0 indicam o centro do plano de projeção, ou seja, o ponto no plano de projeção intersectado pelo eixo principal de projeção, dado no sistema de coordenadas (U, V) .

Considere um sistema de coordenadas global (X, Y, Z) desvinculado da

Figura 6.1: O modelo é composto pelo Centro de Focal (C) e pelo Plano de Projeção. A reta perpendicular ao plano de projeção e que contém o centro focal é chamada de Eixo Principal da câmera. O sistema de coordenadas intrínseco à câmera tem o centro focal como origem, e o eixo X_C coincide com o eixo principal. O plano de projeção é retangular e possui um sistema de coordenadas formado pelos eixos U e V cuja origem é o canto inferior direito de um observador localizado no centro focal. U e V são paralelos e orientados no mesmo sentido dos eixos Y_C e Z_C , respectivamente. A intersecção do eixo principal com o plano de projeção é o ponto (u_0, v_0) , e as dimensões do plano de projeção são u_{count} de largura por v_{count} de altura.

câmera. A projeção de um ponto qualquer é obtida transformando o ponto do sistema de coordenadas global para o sistema de coordenadas intrínseco à câmera, usando uma transformação de corpo rígido (MA, 2004), e em seguida aplicando as equações de projeção. O ponto no sistema global é representado por $\mathbf{p}_{\text{real}} = [x, y, z]^T$, enquanto o ponto no sistema intrínseco continua sendo $\mathbf{p}_{\text{cam}} = [x_{\text{cam}}, y_{\text{cam}}, z_{\text{cam}}]^T$. As equações de projeção são então as seguintes:

$$\begin{aligned} \mathbf{p}_{\text{cam}} &= \mathbf{R}^T(\mathbf{p}_{\text{real}} - \mathbf{C}) \\ u &= f_u \frac{y_{\text{cam}}}{x_{\text{cam}}} + u_0 \\ v &= f_v \frac{z_{\text{cam}}}{x_{\text{cam}}} + v_0, \end{aligned} \tag{6.2}$$

onde $\mathbf{R}$ e $\mathbf{C}$ são os parâmetros que indicam a rotação e a translação do sistema de coordenadas intrínseco ao centro focal da câmera em relação às coordenadas absolutas, respectivamente, com $\mathbf{R}^T$ indicando a matriz transposta de $\mathbf{R}$.

Os parâmetros do modelo de câmera usados pelo modelo de observação são estimados numa etapa de calibração detalhada no apêndice I.

6.2 Objeto Observado

Como já mencionado, devido à grande quantidade de informação captada pela visão, é necessário grande poder computacional para tratar todos os dados obtidos. Além do mais, considerar todos os elementos presentes numa cena exigiria algoritmos muito complexos, que não necessariamente trariam benefícios na mesma proporção à sua complexidade. Um modo eficiente e muito utilizado na literatura para se filtrar a informação é concentrar a atenção em um subconjunto de características e elementos nos quadros de vídeo, como regiões que assumem determinadas formas geométricas, ou objetos que se destacam, como extintores de incêndio, por exemplo.

A escolha do tipo de elemento a ser observado influencia a confiabilidade e a complexidade da solução. Para os sonares, conforme a figura 5.9 no capítulo 5, foi escolhido observar superfícies planas por uma série de motivos. Para o tratamento da visão monocular, os fatores mais importantes que foram considerados na escolha do que deve ser observado são os seguintes:

- Abundância do elemento ou característica no tipo de ambiente considerado: quanto maior sua presença no ambiente, maior a probabilidade de se observar o elemento, e maior a acurácia que se pode obter; por outro lado o excesso de elementos presentes num quadro pode exigir maior capacidade de processamento.
- Existência de um conjunto de características que identifique as projeções de um objeto em diversos quadros. É importante ser capaz de identificar um mesmo elemento do ambiente em mais de um quadro capturado para aumentar a confiabilidade da associação dos elementos mapeados às observações.
- Facilidade de identificação e extração do elemento a partir dos quadros capturados.
- Invariância contra possíveis interferências, como situações de oclusão e variação de iluminação.

Em trabalhos anteriores uma série de características dos quadros já foram escolhidas para serem observadas. As características mais usadas são variações bruscas de intensidade e cor (variações fotométricas) na imagem em duas direções ortogonais, formando pontos (CHIUSO, 2002), (MOURAGNON, 2006). Esses pontos

são fáceis de extrair (JAIN; KASTURI; SCHUNCK, 1995) e em geral são abundantes em qualquer tipo de ambiente, porém eles não apresentam boa resistência a oclusões, e tendem a sofrer variações, podendo até desaparecer, conforme a mudança da iluminação.

Outros autores optam por usar segmentos de retas formados por variações fotométricas (CHANG; AGGARWAL, 1997). Tais segmentos são em geral menos abundantes que os pontos, apresentam alguma proteção contra oclusões e são menos sensíveis às variações de iluminação. A região da imagem ao redor de cada segmento pode ser usada para caracterizá-los, provendo alguma capacidade adicional de identificar projeções oriundas de um mesmo elemento em diferentes quadros. A extração porém é mais complexa, assim como o tratamento posterior.

Em alguns trabalhos que adotam segmentos, uma restrição adicional é adotada, aceitando apenas segmentos com uma dada orientação. Moon, Miura e Shirai (1999) e Borges (2005) utilizam retas verticais para mapear um ambiente com auxílio de visão. A restrição tem o intuito de aumentar a seletividade e agilizar a extração e processamento. Observando apenas segmentos de retas verticais é possível adotar algoritmos de detecção e extração mais eficientes e robustos, e no tipo de ambiente considerado (ambientes internos) eles são encontrados em boa quantidade e em geral representam variações fotométricas ou topológicas em paredes, como batentes, portas, janelas, colunas, quinas e cantos. A vizinhança desses segmentos verticais fornece informação que permite distingui-los entre si com alguma confiança.

Pelos motivos apresentados acima, neste trabalho são observadas variações fotométricas nos quadros que resultam em segmentos de retas verticais. A figura 6.2 apresenta um exemplo de quadro capturado num ambiente típico com vários elementos capazes de produzir os segmentos verticais desejados.

Além das vantagens na identificação e extração, esse tipo de elemento permite uma representação matemática otimizada tanto das projeções extraídas dos quadros como dos elementos do ambiente que resultam nessas projeções. Em geral, segmentos de reta tridimensionais são representados por seis parâmetros (as coordenadas dos dois pontos extremos, por exemplo) e suas projeções bidimensionais, por quatro. Porém, como o robô se desloca apenas sobre um plano perpendicular aos segmentos observados, e assumindo que não existem dois segmentos colineares diferentes num ambiente, pode-se aproximar os segmentos por retas infinitas. Sendo assim, cada reta, por ter inclinação conhecida, pode ser representada pela intersecção da própria reta com o plano de deslocamento do

Figura 6.2: Corredor contendo elementos geradores de retas verticais. Nessa imagem tanto as janelas como as portas são potenciais fontes de retas verticais.

robô, o que resultaria numa representação com apenas um ponto num plano conhecido (dois parâmetros). Do mesmo modo, a projeção de uma reta vertical seria também uma reta vertical (caso a câmera esteja perfeitamente alinhada), e sua representação é dada pela intersecção da projeção com uma reta do plano de projeção que intersecta o eixo principal e seja a projeção de uma reta paralela ao plano de deslocamento, conforme ilustra a figura 6.3.

Apesar da aproximação dos segmentos de reta por retas infinitas permitir uma redução na representação, ela elimina uma característica dos segmentos que poderia ser aproveitada, que é o seu tamanho. Porém essa não é uma perda considerada como significativa pois o tamanho pode variar bruscamente de quadro para quadro com pequenas mudanças na iluminação. Oclusão também afeta muito o tamanho.

No decorrer do texto, as observações da câmera serão tratadas por **quadros**. As projeções nos quadros em forma de reta vertical originadas de elementos do ambiente serão tratadas por **observações** ou por **projeções**. Quando o termo **reta vertical** for usado estará se referindo aos elementos do ambiente que produzem as projeções em forma de retas verticais. Já os elementos presentes no mapa que representam as retas verticais do ambiente serão tratados por **marcos**.

Figura 6.3: Representação de uma Reta Vertical e sua projeção. Na figura é apresentada a câmera, representada pelo centro focal, C , pelo eixo principal, X_C , e pelo plano de projeção, observando uma porta. A projeção de um dos batentes no plano de projeção é exibida. Em vermelho (linha mais grossa) é exibida a reta imaginária paralela ao plano de projeção e perpendicular ao eixo principal, chamada aqui de horizonte. O ponto usado para representar a reta vertical do batente é ressaltado por um círculo ao redor da intersecção do batente com o plano de deslocamento, enquanto que a representação da projeção da mesma reta é ressaltada por outro círculo sobre a intersecção da projeção do batente com a projeção do horizonte.

6.3 Arquitetura

Apesar das diferenças existentes entre os sensores de visão e sonar, a arquitetura básica de ambos os modelos sensoriais é a mesma: primeiro associam-se seqüências de observações a elementos do ambiente e determina-se a posição relativa desses elementos ao robô. Em seguida associam-se esses elementos a algum referencial conhecido, e então a postura obtida do robô a esse referencial através das observações é usada para corrigir a postura do robô.

Uma grande diferença separa o tratamento sensorial dos sonares e da visão: as observações fornecidas pelos sonares são resultado de no máximo um elemento de interesse do ambiente¹, enquanto que um quadro retornado pela câmera pode

¹O retorno do sonar pode ser decorrência de mais de um elemento do ambiente, porém o mo-

conter projeções de diversas retas verticais. Assim, é necessário determinar as associações entre dois conjuntos de dados, as observações e os marcos, sendo que nem todos os marcos precisam ser associados a alguma observação, e nem todas as observações precisam ser associadas a um marco.

A arquitetura do modelo de visão está esquematizada na figura 6.4. Quando um quadro é fornecido pela câmera, o primeiro passo é detectar e extrair as projeções das retas verticais. Essa etapa pode ser considerada uma etapa de pré-processamento, pois as projeções extraídas são consideradas as observações do modelo de visão.

Figura 6.4: Diagrama de blocos do modelo de observação para a visão monocular. A câmera fornece um quadro de vídeo para o módulo detector e extrator de projeções de retas verticais, passando para o módulo de associação as projeções encontradas. Esse módulo por sua vez determina quais projeções correspondem às retas já sendo observadas, quais deixaram de ser observadas e quais estão sendo observadas e que não estavam antes. As retas verticais que estão sendo observadas no quadro são passadas para o próximo módulo, que as associa a marcos mapeados. As associações válidas então são passadas para o módulo que determina a observação esperada para cada marco sendo observado.

De posse das observações (as projeções das retas verticais), o próximo passo é a associação entre as observações e os marcos. O maior problema que se quer evitar numa associação de dados é que seja aceita a associação entre um marco e uma projeção tal que a projeção não foi gerada pela reta vertical que o marco

delo adotado desconsidera essa possibilidade, eliminando as observações que não se enquadram na resposta do tipo de elemento considerado (paredes planas).

representa (um erro do tipo falso positivo). Uma associação errada que passa despercebida pode causar muito mais danos do que associações corretas perdidas (falso negativo).

Para diminuir a aceitação de falsos positivos, é introduzido o conceito de reta vertical sendo observada, cujo objetivo é estabelecer uma associação entre quadros consecutivos processados, aumentando a capacidade do algoritmo em detectar e eliminar associações do tipo falso positivo. Uma reta vertical observada, ou apenas reta vertical, representa uma reta sendo observada no ambiente, e uma etapa de associação é introduzida no modelo para associar as projeções extraídas do quadro a essas retas verticais da cena 3D.

Então para determinar se um marco está sendo observado, além de associá-lo a uma projeção do quadro atual, ele precisa ser associado a uma reta vertical do ambiente por um determinado número de quadros.

Essa etapa de associação entre projeções e retas verticais recebe as observações retiradas do quadro sendo processado e um conjunto de retas verticais que estão sendo observadas. Cada reta já tem ligada a si uma sequência de projeções que foram associadas a ela nas iterações anteriores. O objetivo dessa etapa é adicionar mais uma projeção a essa lista. Caso não seja possível, a reta é descartada. As projeções que não são associadas a nenhuma reta dão origem a novas retas verticais observadas.

A próxima etapa tenta então associar as retas verticais aos marcos. Apenas marcos associados por um número mínimo de iterações à mesma reta vertical são enviados à próxima etapa, responsável por determinar as observações esperadas, $\bar{y}(n)$ (a resposta do modelo de observação de algoritmo de localização).

O passo final obtém a projeção esperada para os marcos sendo observados fornecidos pelo passo anterior para uma dada postura do robô.

A seguir é fornecida uma descrição detalhada de cada etapa do modelo de observação, incluindo a descrição das características usadas para comparar as retas verticais entre si e com os marcos.

6.4 Extração das projeções a partir dos quadros

Essa etapa é responsável por obter dos quadros de vídeo as projeções das retas verticais e as características que permitem identificar, entre quadros, projeções da mesma reta vertical. As técnicas apresentadas são baseadas no trabalho apre-

sentado por Barra e Costa (2005).

6.4.1 Formato das projeções das retas verticais

Para conseguir extrair as projeções desejadas com sucesso, é necessário primeiro definir como as projeções devem estar nos quadros para serem aceitas como projeções de retas verticais. Em teoria, caso o alinhamento da câmera fosse o ideal, a superfície de deslocamento fosse totalmente plana e o robô não produzisse vibração enquanto se desloca, retas verticais no ambiente produziriam retas verticais como projeções. No mundo real nenhuma dessas suposições é verdadeira, porém cada uma das fontes citadas produz diferentes tipos de desalinhamentos e com magnitudes diferentes.

Foi constatado experimentalmente que variações e descontinuidades na superfície de deslocamento e vibrações devido ao deslocamento não provocam distorções significativas nos quadros capturados. A única distorção significativa ocorre durante rotações bruscas, causando borramento da imagem. Em situações como essa, as imagens produzem nenhuma ou quase nenhuma projeção aproveitável usando as técnicas que serão apresentadas.

Para evitar esse problema, colocou-se como restrição que, enquanto o robô estiver executando rotação de ângulos maiores que 15 graus sem se deslocar, nenhum quadro será processado.

Já quanto à questão do alinhamento da câmera sobre o robô, constatou-se em experimentos que o alinhamento incorreto pode causar variações significativas nos quadros. Constatou-se também que é muito difícil posicionar a câmera sobre o robô de modo que o sistema de coordenadas intrínseco à mesma fique perfeitamente alinhado ao sistema de coordenadas de referência do ambiente. Para facilitar o trabalho de alinhamento, o processo de extração deve permitir um relaxamento parcial desse alinhamento. Sendo X_C , Y_C e Z_C os eixos do sistema de coordenadas intrínseco à câmera na posição ideal, é relaxada a acurácia da inclinação da postura da câmera ao redor dos eixos X_C e Y_C com objetivo de obter melhor acurácia na inclinação ao redor do eixo Z_C . Um desvio na inclinação ao redor do eixo Z_C é crítico pois implica num viés de igual magnitude na estimativa da orientação do robô. Desvios nas demais orientações implicam em distorções nas projeções das retas verticais nos quadros, afetando menos a acurácia do estimador.

Os desvios de inclinação considerados fazem com que as projeções sejam retas

com inclinações próximas a 90 graus, mas não verticais, e o algoritmo de extração deve estar pronto para lidar com isso.

6.4.2 Organização da etapa de extração

A extração das observações é dividida em duas sub-etapas. Na primeira, chamada de pré-processamento, as projeções das retas verticais são detectadas e realçadas na imagem. Na segunda, as projeções são localizadas, extraídas e representadas por um conjunto de parâmetros.

6.4.3 Pré-processamento

O objetivo dessa etapa é obter uma imagem onde apenas os pixels mais prováveis de serem parte de uma projeção de reta vertical estejam presentes. A literatura apresenta diversos métodos que podem ser usados para ressaltar as projeções das retas verticais. Grande parte desses métodos utiliza derivadas da imagem (GONZALEZ; WOODS, 1992).

Considerando a imagem uma função bidimensional, $I(u, v)$, onde a intensidade de cada pixel é o valor que a função assume para o ponto (u, v) , pode-se tomar a derivada de $I(u, v)$ em relação a u ou em relação a v . A combinação das duas derivadas resulta no gradiente da imagem, onde a orientação do vetor indica o sentido de maior crescimento da intensidade num dado pixel e o seu módulo, a intensidade desse crescimento.

Considerando apenas a derivada numa direção, tem-se a indicação da variação dos valores da imagem na dada direção. Para a derivada horizontal, quanto maior o valor da derivada num pixel, maior a diferença de intensidade entre os pixels vizinhos à direita e à esquerda, indicando a presença de uma transição fotométrica.

As derivadas podem ser aproximadas utilizando determinadas matrizes e as convoluindo com a imagem. As matrizes são preenchidas de tal modo a calcularem a diferença entre os pixels vizinhos de um lado do pixel central com os vizinhos do outro lado. Na figura 6.5 são mostrados as derivadas na direção vertical e horizontal, e o módulo do gradiente utilizando uma técnica conhecida por Operador de Sobel (GONZALEZ; WOODS, 1992). O Operador de Sobel utiliza os

filtros nas direções horizontal (Sobel_X) e vertical (Sobel_Y) apresentados por:

$$\text{Sobel}_X = \begin{pmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{pmatrix} \quad e \quad \text{Sobel}_Y = \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}. \quad (6.3)$$

Outros filtros operam de formar similar ao filtro de Sobel, trocando apenas a forma da matriz de convolução do filtro, como o filtro de Prewitt (GONZALEZ; WOODS, 1992), por exemplo.

Figura 6.5: Em cima, à esquerda: imagem original. Em cima, à direita: aplicação do Operador de Sobel vertical. Abaixo, à esquerda: aplicação do Operador de Sobel horizontal. Abaixo, à direita: o módulo do gradiente obtido através da combinação dos Operadores de Sobel horizontal e vertical.

Para detectar segmentos verticais, é interessante ressaltar apenas os pixels cujo gradiente é mais intenso na direção horizontal. Após a realização de testes usando diferentes matrizes de convolução, foi escolhido usar apenas a matriz dada

por:

$$\mathbf{grad}_{\text{vert}} = \begin{pmatrix} 1 & 1 & 0 & -1 & -1 \\ 1 & 1 & 0 & -1 & -1 \\ 1 & 1 & 0 & -1 & -1 \\ 1 & 1 & 0 & -1 & -1 \\ 1 & 1 & 0 & -1 & -1 \\ 1 & 1 & 0 & -1 & -1 \\ 1 & 1 & 0 & -1 & -1 \end{pmatrix}. \quad (6.4)$$

Essa matriz é similar a **Sobel_x** porém mais alongada, tanto na vertical como na horizontal. O alongamento vertical aumenta a seletividade de segmentos com inclinações mais próximas a 90 graus (tendendo a segmentos verticais) e elimina seqüências de pixels muito pequenas. O alongamento horizontal visa aumentar a seletividade por transições entre duas regiões homogêneas, filtrando transições sucessivas muito próximas, que seriam difíceis de distinguir uma das outras.

Antes de realizar a diferenciação da imagem, primeiro aplica-se à imagem uma operação de borramento leve, convoluindo a imagem com o filtro **blur** dado por:

$$\mathbf{blur} = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}. \quad (6.5)$$

Essa operação tem como objetivo diminuir a intensidade de ruídos presentes na imagem.

A convolução da imagem pelo filtro **grad_{vert}** é realizada para cada banda de cor, e como resultado final é selecionado para cada pixel o maior valor obtido entre as três bandas (pontos de máximo local). Como resultado se tem uma imagem em níveis de cinza com os segmentos com inclinação próxima à vertical ressaltados. O resultado final dessa etapa de pré-processamento é mostrado na figura 6.6, no quadro à direita.

6.4.4 Detecção das projeções

A partir do resultado do pré-processamento, as projeções são detectadas e suas posições no quadro, calculadas. Essa etapa necessita localizar seqüências de pixel que estejam alinhados verticalmente na imagem pré-processada.

O algoritmo adotado percorre a imagem procurando por pixels que tenham

Figura 6.6: Em cima, à esquerda: imagem original. Em cima, à direita: aplicação das operações para obter a imagem com as retas verticais ressaltadas. Abaixo, à esquerda: pontos de máximo local detectados.

grande probabilidade de pertencer ou estar próximo a uma borda, ou seja, cujo valor esteja acima de um dado limiar. Ao encontrar um pixel que ultrapasse tal limiar, dado por lim_{start} , o algoritmo procura a partir dele, na direção horizontal, um pixel que represente um ponto de máximo local. Esse pixel é o primeiro de uma nova borda, e a partir dele novos pixels são procurados caminhando para cima e para baixo da imagem (na direção vertical).

Para cada pixel recém-adicionado à borda, são observados os três pixels vizinhos logo acima ou abaixo, dependendo do sentido de crescimento da borda para aquele pixel. O próximo pixel a ser adicionado à borda é aquele, dentre os três, com maior intensidade. Caso não seja possível determinar qual direção seguir, a direção vertical é escolhida. Para o primeiro pixel são observados tanto os três vizinhos acima como os três abaixo, e para cada conjunto é adicionado um pixel. O procedimento é repetido até que se encontre um pixel que já faça parte de alguma borda ou até se deparar com uma vizinhança cujos valores dos pixels estejam abaixo de um limiar de parada, lim_{end} , indicando baixa probabilidade da borda continuar por algum dos vizinhos.

Devido a interferências e ruídos nos quadros, pode ser que as projeções sofram pequenas discontinuidades que poderiam interromper sua extração previamente, obtendo várias pequenas seqüências de pixels em vez de uma seqüência única. Para evitar isso, ao atingir uma vizinhança abaixo do limiar lim_{end} , o algoritmo continua crescendo a borda para até Lh vizinhanças consecutivas abaixo do limiar de rejeição. Caso, após a adição de Lh pixels, não seja encontrada uma vizinhança com um pixel acima do limiar lim_{end} , os Lh pixels adicionados são descartados e o crescimento da borda é finalizado.

O procedimento descrito é repetido para todos os pixels que estão acima do limiar de início e que ainda não fazem parte de alguma borda já detectada. O resultado dessa operação é um conjunto de seqüências de pixels adjacentes que formam bordas que lembram segmentos com inclinação próxima a 90 graus. O resultado pode ser visto na figura 6.6, no quadrante inferior à esquerda.

O próximo passo visa detectar seqüências que sejam de fato segmentos de reta. Um segmento de reta precisa ter a mesma inclinação em todos os seus pontos (com exceção das extremidades). Para selecionar apenas as seqüências que mais se aproximam dessa característica, é verificado se elas apresentam inclinação local similar ao longo de seu corpo. Essa operação é realizada calculando a inclinação em cada pixel da seqüência, considerando os $2 \ range_{deriv}$ pixels mais próximos. Caso a variação entre dois pixels seja maior que um limiar, lim_{deriv} , então a seqüência é separada em duas, e os $2 \ range_{deriv} + 1$ pixels usados para determinar as inclinações dos dois pixels que resultaram a ruptura são descartados.

No final desta etapa tem-se um conjunto de seqüências de pixels representando segmentos quase verticais. O último passo para se obter as representações das projeções desejadas é combinar seqüências de pixels que podem ter sido originadas pela mesma reta vertical e foram separadas por algum motivo, que pode ser oclusão, ruído na imagem ou baixo contraste do quadro. Para esse passo é determinado um modelo de reta para cada segmento. O modelo adotado é dado por:

$$u = m.v + u_{fix}, \quad (6.6)$$

onde (u, v) indicam um ponto no sistema de coordenadas intrínseco ao plano de projeção, e m e u_{fix} são os parâmetros que modelam a reta. Para obter m e u_{fix} é usada a técnica de regressão linear (COSTA, 1977). Considerando uma seqüência de N pixels, e cada pixel i da seqüência é representado por (u_i, v_i) , então os

parâmetros m e u_{fix} são dados por:

$$m = \frac{\sum_{i=1}^N (v_i - \bar{v})u_i}{\sum_{i=1}^N (v_i - \bar{v})^2} \quad e \quad u_{fix} = \bar{u} - m\bar{v}, \quad (6.7)$$

onde N indica o número de pixels que compõem a seqüência e $\bar{v}$ e $\bar{u}$ indicam as médias dos v_i e u_i da seqüência, respectivamente.

Os parâmetros de cada seqüência são comparados entre si, e aquelas seqüências cujas diferenças entre os parâmetros u_{fix} e m fiquem abaixo de limiares dados por u_{prox} e m_{prox} , respectivamente, são consideradas como sendo parte do mesmo segmento e são combinadas numa única seqüência. Caso mais de duas seqüências possam ser combinadas com uma terceira seqüência, porém não combinam entre si, então a terceira seqüência é combinada apenas com a seqüência que o parâmetro u_{fix} mais se aproxima.

Uma vez combinadas as seqüências, são descartadas as que têm menos pixels que um dado limiar, lim_{numpix} , e os parâmetros u_{fix} e m são recalculados para as novas seqüências. Finalmente, as retas verticais que possuírem o parâmetro m dentro de uma faixa delimitada por m_{min} e m_{max} são consideradas aceitas.

6.4.5 Representação das projeções

A partir das seqüências de pixels obtidas nas etapas descritas e dos parâmetros do modelo de reta, para cada seqüência são calculados os parâmetros finais das projeções das retas verticais.

Cada projeção de reta vertical é acomodada num modelo que a considera uma reta vertical, assumindo que o parâmetro m tem valor zero. Essa representação tem a vantagem de ter um único parâmetro, u_{vert} , o que permite obter um modelo de observação para a visão com menor complexidade computacional. O valor de u_{vert} é obtido calculando a média de todos os pixels presentes na seqüência que representa a projeção.

A obtenção da incerteza de u_{vert} é um pouco mais complexa, e merece alguns comentários a respeito. Considerando a câmera em alinhamento perfeito, caso u_{vert} fosse obtido de apenas um pixel, então a incerteza sobre sua posição seria dada por $\sigma_{u_{cam}}^2$, que é a variância da posição dos pixels devido ao próprio sensor (câmera de vídeo). Porém, como u_{vert} é resultado de um certo número pixels, $n_{u_{vert}}$, a variância é reduzida para $\frac{\sigma_{u_{cam}}^2}{n_{u_{vert}}}$, considerando que a detecção dos pixels que formam a reta vertical tenha sido perfeita. Se $n_{u_{vert}}$ for um número em torno de 50 e $\sigma_{u_{cam}}^2$ for um valor pequeno em torno de 0,5 (valores razoáveis), o valor da

variância fica em 0,01, o que é desprezível comparado com a incerteza adicionada pelos fatores desconsiderados: os erros oriundos de desalinhamento da câmera e da calibração da câmera (determinação da distância focal f e tamanho real do plano de projeção).

Devido à presença de tantas fontes de incerteza distintas atuando, optou-se por determinar a incerteza combinada para a posição u_{vert} empiricamente. A variância resultante é denominada $\sigma_{u_{vert}}^2$ e será usada como parâmetro de incerteza das observações do modelo de observação.

Além da posição da projeção da reta vertical no quadro de vídeo, um conjunto de características é calculado para cada projeção. O objetivo desse conjunto é o de ajudar na associação entre projeções oriundas da mesma reta vertical. Esse conjunto considera a distribuição de cores nos arredores de cada projeção, e é chamado de Perfil Médio de Cor, ou *PMC*.

6.4.6 Determinação do Perfil Médio de Cor

A hipótese que orientou o desenvolvimento do Perfil Médio de Cor (*PMC*) é que a distribuição de cores nos arredores das retas verticais contém informação suficiente para ajudar a identificar projeções de uma mesma reta vertical em diferentes quadros do vídeo.

O *PMC* considera uma região da imagem centrada na seqüência de pixels que originou cada projeção identificada (referenciada agora de seqüência central). A região se estende por uma faixa de $n_{lateral}$ pixels para a direita e para a esquerda de cada pixel da seqüência central, ao longo da mesma, conforme mostra a figura 6.7. Caso a seqüência central não seja contínua (devido à fusão de seqüências menores durante a obtenção das projeções), então a região considerada também não será contínua.

A partir dessa região, três vetores, sendo um para cada banda de cor da imagem, são criados. Eles têm o mesmo tamanho da largura da região considerada e são criados do seguinte modo: a posição central de cada vetor (o vetor terá sempre tamanho ímpar) recebe o valor médio da respectiva banda de cor dos pixels contidos na seqüência central. Caminhando no vetor em ambos os sentidos, cada posição dos vetores distante i posições do centro recebe o valor médio da respectiva banda de cor dos pixels que estão posicionados a i pixels ao lado do pixel seqüência central que está à mesma altura, sendo que i pode ser positivo ou negativo, indicando o sentido do afastamento do centro do vetor.

Figura 6.7: Representação do Perfil Médio de Cor. A figura representa um pedaço de uma imagem centrado em uma transição fotométrica, com os pixels destacados com a ajuda da sobreposição de um reticulado. Os pixels em preto representam o máximo local da transição fotométrica. A flecha central aponta um pixel em preto, enquanto as setas laterais mostram o tamanho da região considerada no cálculo do *PMC*.

Essa é a última etapa do processamento de um quadro de vídeo. A partir desse ponto, serão usadas apenas as representações das projeções, formadas por u_{vert} e $\sigma_{u_{vert}}^2$, e pelos três vetores do *PMC*, um para cada banda de cor.

6.5 Associação de projeções às retas verticais

Essa etapa recebe o conjunto de projeções extraídas do quadro corrente e tenta associá-las com retas verticais no ambiente. Se possível, as projeções são associadas a retas que já vinham sendo observadas. Para as projeções em que isso não é possível, é assumido que novas retas estão sendo observadas. Essa etapa também é responsável por eliminar as retas que não foram observadas no quadro atual. No final da etapa, apenas retas observadas no quadro atual são mantidas.

Grimson (1990) classifica o problema de associar n observações a m modelos (as retas verticais) como um problema de busca num espaço de correspondência, com n dimensões, sendo que em cada dimensão pode-se assumir m valores distin-

tos. Segundo ele, a menos que seja possível limitar as possibilidades de associação entre as observações e os modelos, seria necessário buscar pelo espaço inteiro para se encontrar o conjunto correto de associações, ou seja, avaliar $(m + 1)^n$ conjuntos de associações (considerando também a possibilidade de não associar as observações a algum modelo), o que teria um custo computacional proibitivo.

Para restringir o número de possibilidades, Grimson (1990) propõe a utilização de restrições, que eliminam regiões inteiras do espaço de busca, diminuindo o número de possibilidades. O método adotado nessa etapa faz uso de restrições que levam em conta tanto a posição das projeções como seus *PMCs*.

Devido à distância percorrida pelo robô entre os quadros processados e a inexistência inicial de uma estimativa para a posição das retas verticais sendo observadas, não se pode prever onde suas projeções vão aparecer no quadro atual. Esse motivo, e o grande número de projeções e retas observadas envolvidas, impedem a adoção de métodos mais simples para realizar a associação de dados, como o algoritmo do vizinho mais próximo, onde são associadas aos modelos as observações que se encontram mais próximas.

Apesar de não ser possível usar o método do vizinho mais próximo diretamente, é possível determinar janelas no quadro atual, para cada reta vertical sendo observada, que determinam uma região em que a probabilidade de encontrar a projeção relativa à reta vertical é grande.

Assim como as projeções, as retas verticais também possuem um Perfil Médio de Cor (*PMC*). O *PMC* das retas verticais são obtidos a partir dos *PMCs* das últimas m_{projs} projeções associadas à reta em quadros anteriores. O *PMC* da reta vertical é a média aritmética dos *PMCs* das projeções consideradas. A partir dos *PMCs* das retas verticais e das projeções do quadro atual, calcula-se uma medida de similaridade, representada por C_{ij} , que é usada como a segunda restrição no espaço de busca. Os índices i e j indicam a reta vertical e a projeção que compõe a associação, respectivamente.

Mesmo após aplicar essas restrições (janelas de busca e *PMCs*), em geral diversos conjuntos de associações entre retas verticais e projeções ainda são possíveis. Para escolher apenas uma dentre as possibilidades restantes, é necessário um método para comparar os conjuntos entre si, de modo a determinar o mais provável. Nessa etapa são usados os *PMCs* como forma de determinar as melhores associações.

A seguir as restrições adotadas são detalhadas, e o algoritmo para determinar

o conjunto de associações é apresentado.

6.5.1 Restrição por janela de busca espacial

Ao realizar a primeira associação de uma reta vertical a uma projeção, dispõe-se apenas da informação da última projeção associada à reta vertical, e não é possível obter uma estimativa sobre sua posição, porém, arbitrando valores máximos e mínimos para a distância da reta vertical à câmera é possível delimitar uma janela de busca no quadro atual baseado na posição da projeção no quadro anterior.

Uma vez que duas ou mais projeções estão associadas a uma reta vertical, é possível obter uma estimativa para a posição da reta vertical, com o auxílio da estimativa do deslocamento do robô. A estimativa está sujeita às incertezas tanto do deslocamento do robô como das projeções sendo pouco precisa; porém, a partir dela, é possível obter uma janela de busca espacial igual ou mais restritiva que a janela inicial.

A seguir são descritos os procedimentos adotados para obter a janelas nas duas situações.

Restrição com uma projeção

Como visto pelo modelo de projeção da câmera, a projeção de um ponto do ambiente é obtida na intersecção do plano de projeção da câmera com uma reta que contém o próprio ponto e o centro focal da câmera. Seguindo o caminho contrário, a partir de uma projeção no plano de projeção pode-se obter apenas a reta que contém o centro focal e o ponto que gerou a projeção. Para determinar a posição exata desse ponto sobre a reta seria necessário conhecer sua distância ao centro focal.

Na ausência dessa informação, são assumidos então valores arbitrários para a coordenada x da reta no sistema de coordenadas intrínseco à câmera, x_{cam} , de tal modo a obter os dois extremos da janela de busca espacial.

Quanto mais perto da câmera um objeto está, mais sua projeção tende a se deslocar em função do deslocamento da câmera, como pode ser visto pela equação 6.1, onde é mostrado que a projeção, u , é inversamente proporcional a x_{cam} . Considerando isso, para determinar o afastamento mínimo da projeção da reta vertical em relação à projeção da mesma no quadro anterior é escolhido um valor grande para a coordenada x_{cam} da reta vertical, dado por x_{max} . Para o afastamento máximo, é escolhido um valor pequeno, dado por x_{min} .

A partir de cada x_{cam} escolhido e a partir da projeção da reta vertical no quadro anterior, u_{ant} , é obtido o valor para y_{cam} da reta usando a equação 6.1. Tendo x_{cam} e y_{cam} da reta no referencial da câmera na postura em que foi capturado o quadro da iteração anterior, uma transformação de corpo rígido é aplicada para obter as coordenadas da reta no referencial atual da câmera, do seguinte modo:

$$\mathbf{p}_{curr} = \mathbf{R}_{move}^T (\mathbf{p}_{ant} - \mathbf{C}_{move}),$$

onde $\mathbf{p}_{ant}$ representa as coordenadas x_{cam} e y_{cam} da reta vertical nas coordenadas da câmera na postura do quadro anterior, representadas por $x_{cam,ant}$ e $y_{cam,ant}$, $\mathbf{p}_{curr}$ representa as coordenadas $x_{cam,curr}$ e $y_{cam,curr}$ da reta vertical nas coordenadas da câmera na postura atual, e $\mathbf{C}_{move}$ e $\mathbf{R}_{move}$ representam o deslocamento do robô, expresso na forma de matrizes de translação e rotação, respectivamente. Finalmente, aplicando $\mathbf{p}_{curr}$ na equação de projeção 6.1 é obtido o valor de u para o quadro atual para o dado $x_{cam,curr}$.

O menor valor de u é atribuído a u_{min} e o maior valor a u_{max} . A janela onde projeções correspondentes devem ser procuradas no quadro atual, para a projeção identificada no quadro anterior, é definida por u_{min} e u_{max} .

Além de considerar diferentes valores para a profundidade da reta observada, é necessário considerar também a incerteza do deslocamento usado para obter os limites da janela. Foi observado em experimentos que a influência da incerteza da translação pode ser desprezada por ser muito pequena (para o robô considerado se deslocando no tipo de ambiente considerado) e pela posição da projeção ser pouco sensível à mesma. Já a incerteza da rotação pode afetar consideravelmente a posição dos limites da janela e não pode ser desprezada. Para considerar a incerteza da rotação, os limites da janela (u_{min} e u_{max}) são calculados em duas situações diferentes: na primeira situação é adicionado ao parâmetro de rotação do deslocamento o desvio padrão da incerteza da rotação, e na segunda situação o desvio padrão da incerteza da rotação é subtraído da rotação do deslocamento. É escolhido como u_{min} o menor valor obtido dentre os dois u_{min} obtidos, e para u_{max} o maior valor dentre os u_{max} , resultando na maior janela possível.

Restrição com mais de uma projeção

Uma vez que duas ou mais projeções estão associadas a uma reta vertical, é possível obter uma estimativa para a posição da reta vertical, $(\bar{x}_{vert}, \bar{y}_{vert})$. O procedimento é o seguinte:

- Determina-se uma estimativa para a posição da reta vertical no ambiente e para sua incerteza a partir da primeira e da última projeção associadas à reta vertical e da estimativa do deslocamento do robô entre as posturas em que foram capturadas as projeções; esta posição da reta vertical é determinada no sistema de coordenadas da câmera na postura mais recente usada;
- Usando a estimativa e a incerteza da posição da reta vertical e do deslocamento do robô, é determinada a janela de busca espacial.

Esse método de obtenção de janelas de busca, além de produzir uma janela de busca menor, tem como vantagem a realização de uma validação extra nas projeções já associadas à reta vertical. Caso alguma das projeções usadas para obter a estimativa da reta vertical tenha sido associada erroneamente, é menos provável que a janela de busca contenha uma projeção que possa ser associada à reta vertical, eliminando uma reta vertical ‘contaminada’.

A escolha da primeira e da última projeções associadas para determinar a estimativa da reta vertical se justifica pois, devido a características da geometria projetiva e de limitações de resolução da câmera (JAIN; KASTURI; SCHUNCK, 1995), deslocamentos de um pixel no quadro capturado podem representar deslocamentos de alguns centímetros até alguns metros no ambiente, dependendo da posição relativa da câmera e do elemento observado.

Deslocamento do robô considerado

A estimativa e incerteza do deslocamento usadas na obtenção das janelas de busca não podem ser derivadas da estimativa da postura do robô para evitar descon continuidades no deslocamento, que podem ser geradas por correções da postura aplicadas pelos modelos de observação.

Por isso, para cada nova reta vertical sendo observada é mantido um Filtro de Kalman Estendido simples, que possui apenas o modelo de dinâmica, para estimar o deslocamento robô entre a primeira projeção associada e a última. Na verdade o filtro é compartilhado por todas as retas criadas numa dada iteração, diminuindo o custo computacional.

Discussão

A segunda parte do algoritmo de janela de busca descrito (a obtenção de janelas de busca a partir de duas projeções) tem uma complexidade computacional bem

maior do que a primeira parte de algoritmo (a obtenção de janelas de busca a partir de apenas a primeira projeção). Caso o processamento esteja se tornando um gargalo para o uso do sistema em tempo real, um dos pontos que podem ser suprimidos é esse, criando janelas de busca com apenas uma projeção (sempre a última associada a uma dada reta vertical).

Equações relacionando duas projeções associadas à mesma reta vertical

As estimativas das retas verticais são obtidas aplicando a equação de projeção 6.1 às duas projeções, u_1 e u_2 , sendo que o índice 1 indica elementos obtidos na situação mais antiga (quando a primeira projeção associada à reta foi coletada) e o índice 2 indica elementos obtidos da situação mais recente (quando a última projeção associada à reta foi coletada). A posição da reta no sistema de coordenadas intrínseco à câmera nos instantes 1 e 2 são representadas por $[x_1, y_1]^T$ e $[x_2, y_2]^T$, respectivamente. Considerando a situação 1 como referência ‘global’, a relação entre $[x_1, y_1]^T$ e $[x_2, y_2]^T$ pode ser dada pela equação de transformação de corpo rígido 6.2.

Considerando o deslocamento bidimensional do robô sobre o plano XY e com possibilidade de rotação apenas sobre o eixo Z , pode-se expandir as matrizes de rotação, $\mathbf{R}$, e translação, $\mathbf{C}$ usadas na equação 6.2 do seguinte modo:

$$\mathbf{R} = \begin{pmatrix} \cos(\theta_{mov}) & -\sin(\theta_{mov}) \\ \sin(\theta_{mov}) & \cos(\theta_{mov}) \end{pmatrix}$$

$$\mathbf{C} = \begin{pmatrix} x_{mov} \\ y_{mov} \end{pmatrix},$$

onde $[x_{mov}, y_{mov}, \theta_{mov}]^T$ representa o deslocamento do robô entre os instantes 1 e 2.

De posse de todos os elementos, pode-se escrever o sistema de equações que

permite a determinação da posição da reta no instante 2:

$$u_1' = u_1 - u_0 = f_u \frac{y_1}{x_1} \quad (6.8)$$

$$u_2' = u_2 - u_0 = f_u \frac{y_2}{x_2} \quad (6.9)$$

$$\begin{pmatrix} x_1 \\ y_1 \end{pmatrix} = \begin{pmatrix} \cos(\theta_{mov}) & -\sin(\theta_{mov}) \\ \sin(\theta_{mov}) & \cos(\theta_{mov}) \end{pmatrix} \begin{pmatrix} x_2 \\ y_2 \end{pmatrix} + \begin{pmatrix} x_{mov} \\ y_{mov} \end{pmatrix}$$

$$\begin{pmatrix} x_1 \\ y_1 \end{pmatrix} = \begin{pmatrix} \cos(\theta_{mov})x_2 - \sin(\theta_{mov})y_2 + x_{mov} \\ \sin(\theta_{mov})x_2 + \cos(\theta_{mov})y_2 + y_{mov} \end{pmatrix}. \quad (6.10)$$

Resolvendo 6.9 para y_2 e substituindo em 6.10:

$$\begin{pmatrix} x_1 \\ y_1 \end{pmatrix} = \begin{pmatrix} x_2(\cos(\theta_{mov}) - \frac{u_2'}{f_u'} \sin(\theta_{mov})) + x_{mov} \\ x_2(\sin(\theta_{mov}) + \frac{u_2'}{f_u'} \cos(\theta_{mov})) + y_{mov} \end{pmatrix}$$

$$= \begin{pmatrix} x_2.C + A \\ x_2.D + B \end{pmatrix}, \quad (6.11)$$

onde A , B , C e D são variáveis auxiliares.

Partindo de 6.8 e substituindo x_1 e y_1 definidos em 6.11:

$$\frac{f_u}{u_1'} = \frac{x_1}{y_1} = \frac{x_2.C + A}{x_2.D + B}$$

$$\frac{f_u}{u_1'}(x_2.D + B) = x_2.C + A$$

$$x_2(\frac{f_u}{u_1'}D - C) = A - \frac{f_u}{u_1'}B$$

$$x_2 = g(\mathbf{x}_{in}) = \frac{A - \frac{f_u}{u_1'}B}{\frac{f_u}{u_1'}D - C}, \quad (6.12)$$

sendo que:

$$\mathbf{x}_{in} = \begin{pmatrix} u_1 \\ u_2 \\ x_{mov} \\ y_{mov} \\ \theta_{mov} \end{pmatrix}. \quad (6.13)$$

Tendo x_2 , pode-se calcular y_2 a partir de x_2 com a equação 6.9.

A equação 6.12 tem um problema. Devido a seu caráter não-linear, ela apresenta grande sensibilidade a variações das variáveis de entrada, em especial de

θ_{mov} , podendo gerar resultados fisicamente absurdos, como uma reta com $x_2 < 0$, indicando que a reta estaria localizada atrás do plano de projeção (o que é impossível, pois para ser captada pelo mesmo, ela precisa estar à sua frente).

Essa questão precisa ser considerada na obtenção da estimativa e incerteza da posição da reta vertical.

Obtenção da estimativa e covariância da posição das retas verticais

Como as variáveis que compõem o vetor $\mathbf{x}_{in}$ são aleatórias (são descritas por *pdfs* representadas pela média e covariância), o resultado que será obtido aplicando as equações 6.12 e 6.9 também será composto por variáveis aleatórias.

Então, para determinar x_2 e y_2 é usada a Transformação Unscented, ou UT (JULIER; UHLMANN; DURRANT-WHYTE, 1995) (a mesma usada no UKF). A UT determina a transformação de uma variável aleatória, descrita por seus dois primeiros momentos (a média e a covariância), por uma função não linear. Para obter o resultado são geradas amostras a partir da variável aleatória de entrada. Essas amostras são conhecidas por *pontos sigma* e são aplicados à função. A partir dos *pontos sigma* transformados são obtidos os dois primeiros momentos que caracterizam a variável aleatória transformada. Maiores detalhes sobre o funcionamento e aplicação da UT são dados no apêndice II, na seção II.5.2.

O problema de obter resultados fisicamente absurdos citados na obtenção da estimativa da posição da reta vertical aparece de forma acentuada na aplicação da UT, pois como a equação 6.12 é muito sensível à variação das variáveis de entrada, a aplicação dos *pontos sigma* nela tem maior probabilidade de gerar resultados absurdos.

Para reduzir o efeito da não-linearidade da equação 6.12, Julier (2002) mostra que se pode diminuir o distanciamento dos *pontos sigma* em relação à estimativa média (a estimativa que é obtida a partir dos valores de entrada) alterando os parâmetros usados para obter os *pontos sigma*.

Os pontos *sigma points* usados pelo UKF para representar $\mathbf{x}_{in}$ são calculados conforme as equações a seguir (para maiores detalhes do funcionamento do UKF, consultar a seção II.5.2 do apêndice II):

$$\begin{aligned}\mathcal{X}^{(0)}(\mathbf{n}) &= \bar{\mathbf{x}}_{in} \\ \mathcal{X}^{(i)}(\mathbf{n}) &= \bar{\mathbf{x}}_{in} + (\sqrt{(N + \delta) \cdot \mathbf{P}_{in}})^{(i)}, \quad i = \{1, \dots, N\} \\ \mathcal{X}^{(i+N)}(\mathbf{n}) &= \bar{\mathbf{x}}_{in} - (\sqrt{(N + \delta) \cdot \mathbf{P}_{in}})^{(i)}, \quad i = \{1, \dots, N\},\end{aligned}$$

onde $\mathcal{X}(\mathbf{n})$ é uma matriz cujas colunas são os *pontos sigma*, $\mathbf{P}_{\text{in}}$ é a matriz de covariância de $\mathbf{x}_{\text{in}}$, e $(\sqrt{(N+\delta)}.\mathbf{P}_{\text{in}})^{(i)}$ indica a coluna i da matriz $\sqrt{(N+\delta)}.\mathbf{P}_{\text{in}}$. A matriz $\sqrt{(N+\delta)}.\mathbf{P}_{\text{in}}$ é o fator de Cholesky (TEUKOLSKY; VETTERLING; FLANNERY, 1992) da matriz $(N+\delta).\mathbf{P}_{\text{in}}$, de tal modo que $\sqrt{(N+\delta)}.\mathbf{P}_{\text{in}}.\sqrt{(N+\delta)}.\mathbf{P}_{\text{in}}^T = (N+\delta).\mathbf{P}_{\text{in}}$. δ é dado por $\delta = \alpha^2(N + \kappa) - N$, onde α é o parâmetro de escala usado para controlar a dispersão os *sigma points* e κ é um parâmetro secundário de escala (JULIER; UHLMANN; DURRANT-WHYTE, 1995) não usado neste trabalho.

Quanto menor α mais próximos os *pontos sigma* ficarão. O valor adotado para α é indicado por α_{prox} .

Ainda assim corre-se o risco de algum *ponto sigma* originar uma posição para a reta vertical com $x_2 < 0$. Para lidar com essa possibilidade, são checados os resultados obtidos para a aplicação de todos os *pontos sigma*. Quando é detectado que $x_2 < x_{\min}$, então o *ponto sigma* é substituído por $[x_{\min}, \frac{f_u}{u_2}x_{\min}]^T$. O parâmetro $x_{\min}$ usado aqui é o mesmo adotado para gerar a janela de busca com apenas uma projeção, descrito anteriormente.

Os *pontos sigma* são obtidos a partir da média e da covariância de $\mathbf{x}_{\text{in}}$. A média é dada pelos valores das estimativas das projeções e do deslocamento. A covariância é dada por:

$$\mathbf{P}_{\text{in}} = \begin{pmatrix} \begin{pmatrix} \sigma_{u_{\text{vert}}}^2 & 0 \\ 0 & \sigma_{u_{\text{vert}}}^2 \end{pmatrix} & \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \\ \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{pmatrix} & \mathbf{Q}_{\text{mov}} \end{pmatrix},$$

onde $\mathbf{Q}_{\text{mov}}$ é a covariância do deslocamento entre os instantes em que foram capturadas as projeções u_1 e u_2 , dado por x_{mov} , y_{mov} e θ_{mov} .

A UT fornece como resposta uma estimativa da posição da reta vertical, $\bar{\mathbf{p}}_2 = [\bar{x}_2, \bar{y}_2]^T$, e sua covariância, $\mathbf{Cov}_{\mathbf{p}_2}$.

A UT necessita do cálculo da raiz quadrada (fator de Cholesky) de $\mathbf{P}_{\text{in}}$ para determinar os *pontos sigma*. Essa é uma operação dispendiosa se precisar ser calculada para cada UT aplicada. Porém como o movimento não é correlacionado com as projeções u_1 e u_2 (segundo o modelo adotado) a matriz $\mathbf{Cov}_{\mathbf{p}_2}$ assume a forma de uma matriz bloco diagonal em que cada bloco é uma matriz de covariância (positivo-definida). Então, o cálculo da raiz quadrada de $\mathbf{P}_{\text{in}}$ da

seguinte forma:

$$\sqrt{\mathbf{P}_{\text{in}}} = \begin{pmatrix} \begin{pmatrix} \sigma_{u_{\text{vert}}} & 0 \\ 0 & \sigma_{u_{\text{vert}}} \end{pmatrix} & \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \\ \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{pmatrix} & \sqrt{\mathbf{Q}_{\text{mov}}} \end{pmatrix},$$

sendo que a matriz $\sqrt{\mathbf{Q}_{\text{mov}}}$ só precisa ser calculada uma vez para as retas verticais que compartilham o mesmo EKF para estimar o deslocamento.

Obtenção da janela de busca a partir da estimativa da reta vertical

Usando a estimativa e covariância da reta vertical no instante anterior, $\mathbf{p}_2$ e $\mathbf{Cov}_{\mathbf{p}_2}$, a estimativa e covariância do deslocamento do robô entre o quadro anterior e o quadro atual, $\mathbf{p}_{\text{mov}2} = [x_{\text{mov}2}, y_{\text{mov}2}, \theta_{\text{mov}2}]^T$ e $\mathbf{Cov}_{\text{mov}2}$, é determinada a estimativa e covariância da projeção da reta vertical no quadro atual, $\bar{u}_{\text{centro}}$, e sua variância, $\sigma_{u_{\text{centro}}}^2$, aplicando novamente a UT. A variável de entrada é descrita por sua estimativa ($\bar{\mathbf{x}}_{\text{in}2}$) e covariância ($\mathbf{P}_{\text{in}2}$) da seguinte forma:

$$\bar{\mathbf{x}}_{\text{in}2} = \begin{pmatrix} x_2 \\ y_2 \\ x_{\text{mov}2} \\ y_{\text{mov}2} \\ \theta_{\text{mov}2} \end{pmatrix} \quad e \quad \mathbf{P}_{\text{in}2} = \begin{pmatrix} \mathbf{Cov}_{\mathbf{p}_2} & \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \\ \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{pmatrix} & \mathbf{Q}_{\text{mov}2} \end{pmatrix}.$$

A função que determina a projeção a partir da estimativa da posição da reta vertical e do deslocamento do robô é obtida combinando uma transformação de corpo rígido para determinar a posição da reta vertical no instante atual, $\mathbf{p}_{\text{curr}} = [x_{\text{curr}}, y_{\text{curr}}]^T$, a partir da posição estimada da reta no instante anterior, $\mathbf{p}_2 = [x_2, y_2]^T$:

$$\begin{pmatrix} x_{\text{curr}} \\ y_{\text{curr}} \end{pmatrix} = \begin{pmatrix} \cos(\theta_{\text{mov}2}) & \sin(\theta_{\text{mov}2}) \\ -\sin(\theta_{\text{mov}2}) & \cos(\theta_{\text{mov}2}) \end{pmatrix} \left(\begin{pmatrix} x_2 \\ y_2 \end{pmatrix} - \begin{pmatrix} x_{\text{mov}2} \\ y_{\text{mov}2} \end{pmatrix} \right),$$

com a equação de projeção da reta vertical no instante atual:

$$u_{\text{centro}} = f_u \frac{y_{\text{curr}}}{x_{\text{curr}}} + u_0.$$

A equação resultante é a seguinte:

$$u_{\text{centro}} = f_u \frac{\cos(\theta_{\text{mov}2})(y_2 - y_{\text{mov}2}) - \sin(\theta_{\text{mov}2})(x_2 - x_{\text{mov}2})}{\cos(\theta_{\text{mov}2})(x_2 - x_{\text{mov}2}) + \sin(\theta_{\text{mov}2})(y_2 - y_{\text{mov}2})} + u_0. \quad (6.14)$$

Aplicando a UT sobre a função descrita pela equação 6.14 obtém-se $\bar{u}_{centro}$ e $\sigma_{u_{centro}}^2$. A partir desses valores a janela de busca é construída encontrando-se os valores de u_{min} e u_{max} da seguinte forma:

$$\begin{aligned} u_{min} &= u_{centro} - c_{janela} \sigma_{centro}^2 \\ u_{max} &= u_{centro} + c_{janela} \sigma_{centro}^2, \end{aligned}$$

sendo a janela delimitada por u_{min} e u_{max} .

O fator de c_{janela} é determinado experimentalmente e visa também corrigir as aproximações realizadas na UT modificada usada para obter a distribuição da posição da reta vertical, que em geral tendem a subestimar a incerteza real associada à posição da reta.

6.5.2 Restrição por Perfil Médio de Cor

Pode-se usar os *PMCs* das projeções do quadro atual e os calculados para as retas verticais para determinar se uma projeção pode ter se originado de uma dada reta. Por exemplo, para uma reta formada pelo batente de uma porta, provavelmente as cores de um dos seus lados será marrom (cor de madeira), e o outro lado vai ser da cor da parede. O *PMC* dessa reta vai ser bem diferente do *PMC* de uma reta formada pela terminação de um quadro de avisos de feltro verde.

O objetivo é restringir ainda mais a possibilidade de pares entre retas verticais e projeções antes de começar a se procurar pelo conjunto de associações com maior probabilidade de ser válido.

Um método muito usado na literatura para determinar se duas regiões em quadros distintos representam um mesmo elemento do ambiente é conhecido como correlação cruzada normalizada (JAIN; KASTURI; SCHUNCK, 1995). A correlação cruzada é obtida do cálculo da diferença quadrática entre as regiões. Representando as regiões nas imagens através das funções f e g , a diferença quadrática entre elas é dada por:

$$\sum_i^N (f(i) - g(i))^2.$$

Expandido os termos, tem-se:

$$\sum_i^N (f(i))^2 + \sum_i^N (g(i))^2 - 2 \sum_i^N f(i)g(i).$$

O termo $\sum_i^N f(i)g(i)$ é a correlação cruzada entre as regiões f e g , e quanto maior seu valor, mais similares serão as regiões. Para evitar que os valores absolutos de f e g influam no valor obtido pela correlação cruzada, a mesma é normalizada, resultando então na medida de similaridade adotada:

$$C_{ij} = \frac{\sum_{k=1}^N (\mathcal{C}_i(k) - \bar{\mathcal{C}}_i) \cdot (\mathcal{C}_j(k) - \bar{\mathcal{C}}_j)}{(N-1) \cdot \text{Var}(\mathcal{C}_i) \cdot \text{Var}(\mathcal{C}_j)}, \quad (6.15)$$

onde N é o tamanho dos PMCs, C_{ij} indica o valor da medida entre a reta vertical i e a projeção j do quadro atual, $\mathcal{C}_i$ e $\mathcal{C}_j$ são os PMCs respectivos, $\bar{\mathcal{C}}_i$ e $\bar{\mathcal{C}}_j$ são as médias dos mesmos, e o operador $\text{Var}(\cdot)$ indica a variância. Esse cálculo é realizado para cada vetor de cor que compõe os PMCs, e aquele que fornecer o menor valor para C_{ij} é escolhido para representar a associação.

Todos os pares cujo valor de C_{ij} esteja abaixo de um dado limiar, indicado por C_{min} , são descartados.

6.5.3 Determinação do conjunto de associações válidas

Sendo n o número de projeções do quadro atual, e m o número de retas verticais sendo rastreadas, o número total de associações possíveis entre uma projeção e uma reta vertical é nm . As restrições aplicadas até agora reduziram essa quantidade, porém ainda podem existir muitos conjuntos de associações possíveis partindo dos pares que sobreviveram às restrições. O próximo passo é determinar o melhor conjunto de associações dentre os possíveis. Para comparar as associações entre si, a medida de similaridade baseada nos *PMCs*, C_{ij} , é usada, e uma última restrição precisa ser obedecida: dentro de um conjunto de associações cada projeção e cada reta vertical podem compor no máximo uma associação.

Para formalizar, pode-se dizer que o objetivo é maximizar a função f_{max} dada por:

$$f_{max} = \sum_{i=1}^m \sum_{j=1}^n \delta_{ij} \cdot C_{ij} \quad (6.16)$$

variando apenas o valor de δ_{ij} , que pode assumir o valor 1 caso i e j sejam escolhidas como pares de possível correspondência e caso C_{ij} tenha sido aprovado nas etapas anteriores, ou zero caso contrário. δ_{ij} deve respeitar ainda:

- $0 \leq \sum_{i=1}^m \delta_{ij} \leq 1$ e
- $0 \leq \sum_{j=1}^n \delta_{ij} \leq 1$,

ou seja, cada projeção e cada reta vertical podem compor no máximo uma asso-

ciação.

Por exemplo, considere que $m = 2$ e $n = 3$. Os pares (i, j) validados são: $(1,1)$, $(1,2)$, $(1,3)$, $(2,1)$, $(2,2)$ e $(2,3)$. Caso o par $(1,1)$ seja escolhido, tem-se que $\delta_{i=1,j=1} = 1$, $\delta_{i=1,j \neq 1} = 0$ e $\delta_{i \neq 1,j=1} = 0$, restando os pares $(2,2)$ e $(2,3)$ para se continuar escolhendo.

Esse tipo de problema é conhecido na literatura por associação de dados de multi-sensores e multi-alvos (BAR-SHALOM; LI, 1995), e diversas técnicas já foram apresentadas. A técnica conhecida como JPDA (da sigla em inglês de Associação de Dados usando Probabilidade Conjunta) (BAR-SHALOM; LI, 1995) compara todas as possibilidades de associação em cada iteração baseadas em uma probabilidade atribuída a cada potencial associação. Já a técnica conhecida por MHT (do inglês de Teste de Múltiplas Hipóteses) (REID, 1979), tem como filosofia manter hipóteses mais prováveis dos conjuntos de associações durante um certo número de iterações para se determinar qual dos conjuntos produz resultados mais prováveis.

Ambos os métodos são computacionalmente dispendiosos e, mesmo existindo soluções para diminuir a complexidade computacional dos métodos, (ROECKER; PHILLIS, 1993), (ROECKER, 1994), em situações onde não é possível restringir previamente as associações possíveis entre alvos e observações eles ainda são computacionalmente complexos. Esses métodos foram desenvolvidos para associar observações sensoriais a objetos usando primariamente a trajetória que descrevem, porém podem ser adaptados para usar outras formas de comparação entre estimativa e observação, desde que seja possível obter uma métrica confiável para determinar individualmente a probabilidade de uma associação possível ser de fato correta.

Apesar da superioridade desses métodos, foi escolhido usar um método baseado em um algoritmo muito mais simples, conhecido por it Vizinho mais próximo (BAR-SHALOM; LI, 1995), onde a cada iteração do algoritmo é escolhida como válida a associação com maior probabilidade de ser correta. Apesar de globalmente esse método nem sempre apontar para a solução correta, foi determinado empiricamente que para o problema corrente uma variante dessa abordagem, baseada nas considerações a seguir, fornece bons resultados a um baixo custo computacional.

6.5.4 Considerações sobre a medida de similaridade

A medida de similaridade adotada apresenta uma deficiência importante. Nem sempre que a associação com maior C_{ij} para uma reta vertical é a associação correta. Pode ser que uma outra associação com C_{ij} menor seja de fato a associação válida.

Isso implica que mesmo que seja possível maximizar f_{max} conforme a equação 6.16, o resultado obtido pode não ser o mais próximo da solução correta.

Essa deficiência pode ser gerada por ruído ou obstruções nos quadros do vídeo, porém uma outra fonte muito comum é presença de estruturas no ambiente que levam à formação de projeções com PMC s similares e próximas umas das outras, resultando na formação de associações de retas verticais com mais de uma dessas projeções.

Figura 6.8: Exemplo de ambigüidade na detecção das projeções de retas verticais. As janelas na parede da direita, em especial as duas colunas divisórias internas, geram projeções com PMC s muito similares.

A figura 6.8 exemplifica uma situação onde uma estrutura em forma de janelas apresenta padrões de listras verticais que acabam gerando as projeções ambíguas. Esse tipo de padrão aparece em estruturas como seqüências de janelas em corredores, alguns tipos de cortinas, portões em grade, ou mesmo em elementos de decoração.

Não há como evitar a escolha de uma associação errada em todos os casos que situações desse tipo ocorrerem.

O melhor modo para escolher a associação correta entre as demais é começar

selecionando para o conjunto de associações válidas as associações cujas retas verticais têm um menor número de associações potenciais a projeções ambíguas. Assim, começando as associações por essas retas, a probabilidade de conseguir um conjunto correto de associações é maior.

6.5.5 Algoritmo para determinar as associações válidas entre retas verticais e projeções

O algoritmo adotado tem como principal característica o baixo custo computacional em comparação com abordagens como JPDA, MHT e buscas em árvore (RUSSELL; NORVIG, 1995). Assim como o algoritmo do *Vizinho mais próximo*, ele não possui nenhum tipo de retrocesso² como em geral algoritmos mais complexos de busca e associação de dados apresentam.

O algoritmo adotado seleciona como válida a cada iteração uma associação, dando preferência àquelas com valor alto da medida de similaridade C_{ij} e que ao mesmo tempo a reta vertical presente na associação não esteja envolvida em ambigüidades similares à exemplificada na figura 6.8. O algoritmo encerra quando mais nenhuma associação, obedecendo à restrição que cada projeção e cada reta vertical podem compor no máximo uma associação, esteja disponível.

O algoritmo é descrito em detalhes a seguir:

Pré-processamento:

1. Os pares representando as possíveis associações entre retas verticais e projeções extraídas do quadro atual são divididos em grupos segundo a reta vertical contida em cada associação.
2. Cada grupo é ordenado segundo o valor de C_{ij} .

Executar enquanto existirem associações nos grupos:

1. Adotando o índice k para representar a ordem da associação dentro de seu grupo de tal modo que quanto menor o valor de k maior o valor de $C_{ij}^{(k)}$ e que $1 \leq k \leq kMax_i$, calcula-se para cada grupo i o seguinte valor:

$$\begin{cases} \text{se } kMax_i > 1, \text{ então } d_i = C_{ij}^{(1)}(C_{ij}^{(1)} - C_{ij}^{(2)}) \\ \text{se } kMax_i = 1, \text{ então } d_i = C_{ij}^{(1)}(C_{ij}^{(1)} - C_{min}), \end{cases}$$

²o retrocesso num algoritmo permite dar ‘passos para trás’ na construção da solução, permitindo refazer decisões tomadas em passos anteriores visando encontrar um outro caminho que leve a uma solução melhor.

onde C_{min} é o limite usado para restringir associações com medidas de similaridade muito pequenas. Valores altos de d_i indicam associações com alta similaridade e pequena ambigüidade entre as associações possíveis para a reta vertical.

2. A associação escolhida como válida é a com maior medida de similaridade do grupo que possui maior valor de d_i .
3. O grupo é eliminado, e os pares nos demais grupos que fazem referência à projeção que foi aceita como associação válida também são eliminados.

É possível implementar o algoritmo de modo que ele tenha complexidade da ordem de $O(m^2 + n.m.\log(n))$. Como em geral n é próximo de m , a ordem pode ser simplificada para $O(m^2.\log(m))$. Essa complexidade é bem menor do que a apresentada por algoritmos que precisam analisar todos os conjuntos possíveis, que seria de $O(m!)$, caso n seja próximo de m . Mesmo com algoritmos que consigam reduzir a complexidade do algoritmo de busca completa, eles em geral apresentam complexidade maior que $O(m^2.\log(m))$, tendo em geral como limite superior $O(m!)$.

Barra e Costa (2005) apresentam resultados experimentais que demonstram a eficácia dessa abordagem.

6.6 Modelo de Marcos

São tratados por marcos as retas verticais cuja posição é conhecida. A intenção é representar no mapa apenas as transições fotométricas verticais mais estáveis e que mais se destacam. Cada marco é representado por um ponto no plano de deslocamento do robô indicando sua posição no ambiente, e por características que ajudem a identificá-lo através da distribuição de cores ao seu redor. Diferentemente das retas verticais observadas e das projeções extraídas dos quadros, os dados sobre a distribuição de cores na vizinhança dos marcos é muito menos informativa, devido à grande variação de iluminação e configuração do ambiente, que pode adicionar ou remover oclusões parciais, impossibilitando o uso de características detalhadas como os *PMCs*.

A informação que é mantida sobre a distribuição de cores é denominada Médias de Cores, ou *MC*. O *MC* armazena, para cada banda de cor, a intensidade média da cor à esquerda e à direita da reta vertical, do ponto de vista da câmera. O tamanho da região considerada para o cálculo das médias é o

mesmo tamanho considerado para obtenção dos *PMCs* das projeções. Os atributos que armazenam os valores médios são denominados R_{dir} , R_{esq} , G_{dir} , G_{esq} , B_{dir} e B_{esq} , sendo que R , G e B indicam a banda de cor a que cada atributo se refere (vermelho, verde e azul, respectivamente) e os subscritos *dir* e *esq* indicam o lado da reta (direito e esquerdo, respectivamente).

O modelo de marcos primeiro determina quais marcos estão sendo observados associando retas verticais observadas a eles. Em seguida os marcos selecionados como sendo observados são usados para corrigir a postura do robô, juntamente com a projeção mais recente associada à reta vertical associada ao marco.

Entre ser associado a uma reta vertical e ser usado para corrigir a postura do robô, um marco passa primeiro por um período de validação, batizado de quarentena. Durante a quarentena é exigido que um marco seja sempre associado com a mesma reta vertical por um número mínimo de iterações consecutivas, dado por $N_{marco,assoc}$. Além disso, o marco só pode sair da quarentena e ser considerado válido caso a reta vertical à qual está associado tenha um número mínimo de projeções associadas, dado por $N_{proj,assoc}$, ou seja, foi observada num número mínimo de quadros consecutivos.

Através da quarentena cria-se uma conexão entre as duas etapas de associação: associação de projeções a retas e associação de retas a marcos. Isso é importante pois cada etapa foi desenhada usando abordagens e características distintas para determinar os conjuntos de associação, e a união das etapas aumenta a robustez do modelo de visão, pois a probabilidade de que falsos positivos passem despercebidos por ambas as etapas é menor do que caso só existisse uma delas.

Além do mais, o simples fato de um mesmo marco ser associado à mesma reta em quadros diferentes é um forte indício que ele não é um falso positivo.

6.6.1 Associação entre marcos e retas verticais

O primeiro passo é determinar quais marcos estão no alcance da câmera. Para isso é calculada a projeção esperada de cada marco, $u_{vert,marco}$, considerando a estimativa mais atualizada da postura do robô. Os marcos considerados são aqueles que estão à frente da câmera, que não estão encobertos por nenhum elemento mapeado (o segmento que conecta o marco com o centro focal da câmera não intersecta nenhum elemento do mapa) e cuja projeção obedeça a:

$$-u_{extra} < u_{vert,marco} < u_{count} + u_{extra},$$

sendo que u_{count} é o tamanho em pixels do plano de projeção e u_{extra} é um fator de relaxamento para não desconsiderar marcos cujas projeções sejam excluídas devido à inacurácia da postura do robô.

O primeiro conjunto de associações potenciais é formado por associações realizadas entre as projeções esperadas dos marcos, $u_{vert,marco}$, e as projeções mais recentes das retas verticais, $u_{vert,reta}$, tendo como única restrição à sua formação uma medida de similaridade baseada nas características da distribuição de cor de cada elemento, similar à medida adotada na etapa anterior.

A partir do conjunto inicial de associações potenciais, o algoritmo determina uma série de conjuntos de associações entre marcos e retas verticais. Cada novo conjunto representa uma potencial resposta da etapa de associação. O último passo é selecionar um dos conjuntos como o resultado do algoritmo, buscando entre eles o conjunto mais provável de conter um maior número de associações válidas e o mínimo de associações inválidas.

Enquanto na associação entre retas verticais e projeções do quadro atual o conjunto de associações válidas é gerado de forma incremental e monotônica, no algoritmo de associação entre retas verticais e marcos primeiro são gerados uma série de conjuntos, e depois um deles é selecionado. Considerando que os conjuntos gerados formem um espaço de busca, pode-se dizer que o método adotado está mais próximo dos algoritmos de busca tradicionais.

Apesar da maior complexidade, é baixa a probabilidade do algoritmo de associação entre retas verticais e marcos apresentar tempo de execução muito mais elevado que a associação entre projeções e retas verticais, principalmente por dois motivos: i) o número de marcos disponíveis para formar associações em geral é menor do que o número de projeções extraídas num quadro, resultando em um número menor de associações possíveis em relação à associação entre projeções e retas; e ii) o critério de restrição na formação dos conjuntos não permite a geração de muitos grupos. O segundo motivo impede que o algoritmo atinja o limite superior de complexidade, que é exponencial.

Nesse caso optou-se por não usar diretamente JPDA ou alguma variante pois nesses algoritmos, apesar de se considerar a probabilidade de todo o conjunto de associações para se determinar o conjunto aceito, é usado para obter a probabilidade global apenas as probabilidades individuais de cada associação ser correta, e como será visto adiante para o problema abordado aqui, a relação entre as associações que compõe o conjunto de associações válidas é tão importante quanto as relações individuais.

6.6.2 Formação e comparação entre os conjuntos de projeção por disposição espacial

Antes de apresentar o algoritmo de associação entre retas verticais e marcos, é introduzido o conceito de disposição espacial, que inspira a geração inicial dos conjuntos de associações e posteriormente é usado na comparação entre esses conjuntos.

Para explicar o sentido dado à expressão disposição espacial, convém pensar primeiro na ordem em que as projeções dos elementos de um conjunto aparecem no plano de projeção. Por exemplo, caminhando num sentido no plano de projeção, é esperado cruzar primeiro com a projeção $u_{vert,marco}$ do marco A , em seguida com a do marco B , e então com a do marco C . É esperado que as projeções extraídas do quadro referentes a esses marcos sejam encontradas na mesma ordem.

Porém não é só a ordem o importante, mas sim a distância relativa entre as projeções. Por exemplo, para que as associações entre o marco A e a reta vertical X e entre o marco B e a reta vertical Y possam ser validadas, é necessário que as distâncias AB (distância entre as projeções A e B , incluindo o sinal de direção, ou seja, $AB = -BA$) e XY sejam similares.

Uma outra forma de se encarar essa propriedade é observando a distância entre a projeção esperada do marco e a projeção do quadro atual associada à reta vertical de uma mesma associação (AX e BY no exemplo acima). Essa distância, representada por u_{shift} , deve ser similar para todas as associações que compõem o conjunto de associações que for aceito como válido.

Estendendo essa comparação para todas as associações de um conjunto, o método pode ser visto como uma medida do alinhamento entre o conjunto formado pelas projeções das retas verticais e o conjunto formado pelas projeções esperadas dos marcos. Para exemplificar, supõe-se que cada conjunto de projeções é representado como um tira de papel, e cada projeção um traço vertical na tira. O objetivo é tentar alinhar as duas tiras de papel, uma referente aos marcos e a outra, às retas verticais, de modo que os traços de ambas as tiras coincidam, e medir de alguma forma o quanto faltou para se obter um alinhamento perfeito.

É usado a variância dos u_{shift} de um conjunto de associações, $Var(u_{shift})$, para determinar o quão longe se está de um alinhamento perfeito: quanto menor a variância, maior a probabilidade do conjunto ser formado por associações válidas. Um critério para comparar conjuntos de associações usando apenas $Var(u_{shift})$, considerando um conjunto de associações entre retas verticais e marcos indexado

por k , contendo n_k associações, poderia ser dado por:

$$D_{shift}^{(k)} = \frac{1}{Var(u_{shift}^{(k)})}, \quad (6.17)$$

sendo que quanto maior $D_{shift}^{(k)}$ maior a probabilidade de o conjunto ser composto por associações válidas.

Deve ser ressaltado, porém, que o exemplo das tiras de papel é apenas ilustrativo, pois devido à forma que as projeções são geradas (segundo a equação 6.1), conjuntos de projeções dos mesmos elementos obtidos a partir de diferentes posições de câmera não podem ser alinhados perfeitamente apenas somando ou subtraindo um valor de correção às projeções de um dos conjuntos. Isso é apenas uma aproximação, que é tão melhor quanto mais próximos os elementos observados do ambiente estiverem entre si ou então mais próximas forem as posições de câmera em que os conjuntos foram obtidos.

Isso quer dizer que a importância que deve ser dada a $D_{shift}^{(k)}$ depende dos valores dos u_{shift} do conjunto e da distância no ambiente entre os elementos observados, e essas informações devem ser levadas em consideração ao compor $D_{shift}^{(k)}$ com outros critérios de comparação para obter um método mais robusto para determinar qual o conjunto de associações deve ser aprovado.

Dessa forma, deve-se somar à $Var(u_{shift}^{(k)})$ na equação 6.17 um termo de amortecimento que, quanto maior for, menor será o valor de $D_{shift}^{(k)}$ e menor sua sensibilidade às variações de $Var(u_{shift}^{(k)})$. Esse termo de amortecimento é composto por dois fatores: o valor máximo de u_{shift} do conjunto considerado e a distância máxima entre os elementos que estão sendo observados.

O valor máximo do u_{shift} é fácil obter, sendo chamado de $u_{shift,max}$. A distância máxima entre os marcos observados também não seria difícil de obter, porém nem sempre grandes distâncias entre os elementos no ambiente causam grandes variações de u_{shift} , pois a posição deles em relação à câmera também é importante. Um indicador que se mostrou mais sensível é a distância máxima entre as projeções estimadas dos marcos sendo observados, dada por $u_{max,marco} - u_{min,marco}$. Aplicando o amortecimento 6.17, obtém-se o seguinte:

$$E_{shift}^{(k)} = \frac{1}{Var(u_{shift}^{(k)}) + \frac{u_{shift,max}(u_{max,marco} - u_{min,marco})}{u_{shift,att,lim}}}, \quad (6.18)$$

lembrando que $E_{shift}^{(k)}$ é um termo para ser usado em combinação com outros critérios de comparação. Esse fator foi definido empiricamente, onde se chegou à conclusão que o melhor modo de balancear a importância dada a $Var(u_{shift}^{(k)})$ é

realizando uma soma ponderada com a multiplicação de seus fatores limitantes ($u_{shift,max}$ e $u_{max,marco}^{(k)} - u_{min,marco}^{(k)}$), onde $u_{shift_{att,lim}}$ é um fator de ponderação determinado empiricamente.

A adoção da variância de u_{shift} para comparar os potenciais conjuntos de associações pode favorecer a seleção de conjuntos menores, pois qualquer conjunto de tamanho N terá variância maior que o conjunto formado por todas as suas associações menos a associação cujo u_{shift} mais se afasta da média. Para balancear esse efeito, o número de elementos no conjunto é adicionado ao critério de comparação como um fator extra. Assim, quanto mais elementos num conjunto, maior a possibilidade do conjunto ser validado. Empiricamente percebeu-se que usar a raiz quadrada do número de elementos do conjunto produz resultados superiores.

Multiplicando 6.17 por $\sqrt{n_k}$, obtém-se finalmente:

$$F_{shift}^{(k)} = \frac{\sqrt{n_k}}{Var(u_{shift}^{(k)}) + \frac{u_{shift,max}(u_{max,marco}^{(k)} - u_{min,marco}^{(k)})}{u_{shift_{att,lim}}}}. \quad (6.19)$$

A variância de u_{shift} comprovou-se útil para selecionar as associações válidas quando a estimativa e a postura real do robô estão próximas ou os marcos estão localizados numa mesma região do espaço, porém é necessário usar um segundo critério para ajudar a determinar os conjuntos de associações nas demais situações. As características de cor, os *MCs* do marcos e os *PMCs* das retas verticais, são usadas para compor o critério de comparação adotado, que será descrito na próxima seção.

6.6.3 Descrição do algoritmo de associação

O algoritmo recebe como entrada um conjunto de retas verticais sendo observadas e determina um conjunto de associações entre elas e os marcos mapeados.

O algoritmo de associação é descrito a seguir.

1. **Determinação das associações entre marcos e retas verticais.** São formadas apenas associações entre marcos e retas verticais que respeitam uma restrição baseada nas características fotométricas. Essa restrição leva em conta os *MCs* dos marcos e os *PMCs* das retas verticais. A partir delas é obtidas uma medida de similaridade para cada combinação de marco e reta vertical, representada por C'_{ij} , onde os índices i e j indicam o marco e a reta vertical que compõem a potencial associação.

Para obter C'_{ij} , primeiro os $PMCs$ das retas verticais são convertidos em MCs . Para isso, R_{dir} , R_{esq} , G_{dir} , G_{esq} , B_{dir} e B_{esq} são obtidos calculando a média de cada metade do vetor da respectiva banda de cor.

Em seguida, os seguintes valores são obtidos (são mostrados apenas os valores para MCs dos marcos, i , mas o cálculo para os $PMCs$ das retas verticais é idêntico):

$$\begin{aligned} \left(R_{diff}^{(i)}, G_{diff}^{(i)}, B_{diff}^{(i)} \right) &= \left(R_{dir}^{(i)} - R_{esq}^{(i)}, G_{dir}^{(i)} - G_{esq}^{(i)}, B_{dir}^{(i)} - B_{esq}^{(i)} \right) \\ M^{(i)} &= \frac{R_{dir}^{(i)} + R_{esq}^{(i)} + G_{dir}^{(i)} + G_{esq}^{(i)} + B_{dir}^{(i)} + B_{esq}^{(i)}}{6}, \end{aligned}$$

onde $R_{diff}^{(i)}$, $G_{diff}^{(i)}$ e $B_{diff}^{(i)}$ indicam a variação de cor encontrada na transição, e $M^{(i)}$ é a intensidade média dos pixels ao redor da reta vertical (considerando que cada pixel pode assumir valores entre 0 e 255).

Usando $\left(R_{diff}^{(i)}, G_{diff}^{(i)}, B_{diff}^{(i)} \right)$ para os marcos e para as projeções, obtém-se:

$$diff_{banda} = \frac{\left| R_{diff}^{(i)} - R_{diff}^{(j)} \right| + \left| G_{diff}^{(i)} - G_{diff}^{(j)} \right| + \left| B_{diff}^{(i)} - B_{diff}^{(j)} \right|}{3},$$

que é a diferença entre as transições de cor da projeção e do marco.

Usando $M^{(i)}$ para os marcos e para as projeções, obtém-se:

$$diff_{média} = \left| M^{(i)} - M^{(j)} \right|, \quad (6.20)$$

que é a diferença entre as intensidades médias da projeção e do marco.

O valor de C'_{ij} é 0 caso $(30 - diff_{banda}) < 0$ ou $(100 - diff_{média}) < 0$. Caso contrário, o valor é dado por:

$$C'_{ij} = \frac{0,7(30 - diff_{banda}) + 0,3(100 - diff_{média})}{0,7 \cdot 30 + 0,3 \cdot 100},$$

onde C'_{ij} tem seu valor limitado entre 0 e 1. As associações só são aceitas caso $C'_{ij} > 0$.

A equação para C'_{ij} foi obtida empiricamente, de modo a balancear a importância tanto da intensidade média da imagem ao redor das retas verticais quanto da variação de cor responsável pela projeção da reta vertical. Desse modo C'_{ij} é muito mais sensível à variação de cor do que a intensidade média (devido aos multiplicadores 0,7 e 0,3), pois é esperado variações da intensidade no ambiente entre o instante em que o mapa foi feito e o que está sendo usado pelo modelo de observação. Porém, caso a intensidade varie muito, ela pode afetar o grau de confiança depositado na associação que C'_{ij}

representa, apesar de precisar de uma variação muito maior de intensidade do que de variação de cores para uma associação ser rejeitada.

2. **Formação dos conjuntos de associação.** A partir das associações formadas, u_{shift} é calculado como visto na seção anterior. Em seguida as associações são ordenadas segundo u_{shift} e armazenadas num vetor. Uma janela móvel percorre o vetor, sendo que a janela abrange um número variável de associações de modo que a diferença entre os u_{shift} da primeira e da última associações dentro da janela seja menor que um limiar, indicado por du_{shift} . Uma extremidade da janela avança uma associação por vez no vetor enquanto a outra extremidade é recalculada segundo a regra acima. A janela continua avançando até que o seu fim atinja o último elemento do vetor.

Para cada seqüência de associações delimitada em cada avanço da janela, conjuntos de associações são formados respeitando a seguinte regra: cada marco e reta vertical pode fazer parte de apenas uma associação por conjunto formado. Para minimizar o número de conjuntos que devem ser gerados e analisados, apenas os conjuntos com mais associações são mantidos. Essa limitação tem como objetivo diminuir a complexidade computacional do algoritmo. Porém é uma restrição muito forte que torna o algoritmo muito dependente da escolha do du_{shift} apropriado, que pode ser problemático de obter, pois ele dependeria, entre outras coisas, da disparidade entre a postura real e a estimada do robô (quanto maior for a disparidade, maior o u_{shift} esperado, maior a variância dos u_{shift} nos conjuntos e maior deve ser du_{shift}).

Foi observado em experimentos que, através da utilização de mais de uma janela com valores distintos de du_{shift} , pode-se obter resultados similares a usar todos os conjuntos possíveis, porém sem aumentar a complexidade computacional do algoritmo. Para cada janela de tamanho diferente, o algoritmo descrito nesse item é repetido.

Usando dados simulados e coletados no ambiente de testes usados nos experimentos apresentados no capítulo 7, optou-se por usar 4 tamanhos diferentes de janela, representados por $du_{shift,1}$, $du_{shift,2}$, $du_{shift,3}$ e $du_{shift,4}$. O valor desses parâmetros é especificado no fim deste capítulo na seção 6.8.

3. **Comparação entre os conjuntos de associações.** São calculados os valores de comparação para cada conjunto de associações obtido. Cada conjunto é indexado pelo índice k . A medida de similaridade leva em conta, para cada conjunto k , além do fator $F_{shift}^{(k)}$, a média dos C'_{ij} das associações

do conjunto $(\bar{C}_{ij}^{(k)})'$. A medida de similaridade associada a um conjunto de associações é dada pela seguinte expressão:

$$\begin{aligned} Sc_k &= \bar{C}_{ij}^{(k)'} + uShift_{mult} \cdot F_{shift}^{(k)} \\ &= \bar{C}_{ij}^{(k)'} + uShift_{mult} \cdot \frac{\sqrt{n_k}}{Var(u_{shift}^{(k)}) + \frac{u_{shift,max}(u_{max,marco}^{(k)} - u_{min,marco}^{(k)})}{uShift_{att,lim}}}, \end{aligned}$$

sendo que quanto maior Sc_k maior a probabilidade de o conjunto ser válido, e $uShift_{mult}$ é uma constante que compatibiliza e determina a importância entre os valores de $\bar{C}_{ij}^{(k)'}$ e $F_{shift}^{(k)}$.

4. Determinação do conjunto de associações aceitas como válidas.

Finalmente, dentre os conjuntos formados, é escolhido aquele com o maior valor de Sc_k .

Os parâmetros que determinam Sc_k são apresentados no fim desse capítulo na seção 6.8, juntamente com os demais parâmetros usados no capítulo e deixados em aberto.

Discussão

A capacidade de Sc_k determinar o conjunto de associações corretas depende da quantidade de marcos mapeados e da qualidade das retas verticais que o ambiente pode fornecer. Por exemplo, se as retas verticais presentes permitirem a coleta de marcos muito próximos uns dos outros e que sejam originados por elementos do ambiente próximos entre si, u_{shift} é uma ótima ferramenta para determinar o conjunto correto de associações. Por outro lado, se elementos em profundidades distantes geram projeções próximas umas das outras (ao observar um salão com diversas colunas pontilhando seu interior, por exemplo), $\bar{C}_{ij}^{(k)'}$ pode ser mais importante que $Var(u_{shift}^{(k)})$.

Outro fator que deve ser levado em consideração é a disparidade esperada entre a postura real e a postura estimada do robô. Quanto maior a disparidade, maior o u_{shift} das associações corretas, e potencialmente maior a variação entre esses u_{shift} . Nesses casos novamente $\bar{C}_{ij}^{(k)'}$ deve receber maior peso do que $Var(u_{shift}^{(k)})$.

Apesar das considerações feitas acima, a formação dos conjuntos de associações usando C'_{ij} para limitar as associações e a proximidade dos valores de u_{shift} para gerar os conjuntos foi o suficiente, em experimentos, para restringir os possíveis conjuntos de associações de modo que os critérios de comparação adota-

dos fossem capazes de selecionar conjuntos contendo apenas associações válidas, filtrando conjuntos com associações erradas com grande eficácia. Porém a filtragem excessiva observada de associações válidas motiva a investigação de melhorias na técnica de associação entre marcos e retas verticais.

Uma solução para minimizar esse tipo de limitação no uso do u_{shift} seria agrupar as retas verticais em dois ou três subconjuntos menores, sendo que cada subconjunto conteria retas verticais cujas projeções se localizam em determinada região do último quadro capturado. Por exemplo, um grupo é formado com as retas verticais que possuem projeções à direita do centro do quadro, e outro grupo com as retas que possuem projeções à esquerda do centro do quadro. Para cada grupo seria então executado o algoritmo acima até obter-se possíveis conjuntos de associações válidas para cada grupo de retas verticais. Novos conjuntos seriam então obtidos a partir desses, combinando suas associações de modo que em cada conjunto resultante, cada marco e cada reta vertical faça parte de no máximo uma associação.

Esse solução não resolveria totalmente a limitação do uso dos u_{shift} , porém a minimizaria para a maioria das situações encontradas em ambientes internos estruturados (onde as projeções próximas tendem a serem originadas de elementos fisicamente próximos também).

Estuda-se também a possibilidade de uma segunda melhoria, que visa eliminar as limitações dos u_{shift} . Essa melhoria propõe combinar a abordagem atual com a construção de janelas de busca no quadro atual. Partindo das projeções estimadas dos marcos e variando levemente a postura estimada do robô sobre a estimativa inicial, usa-se os modelos bidimensionais que representam a posição dos marcos no ambiente para criar pequenas janelas de busca no quadro atual. À medida que se variam os parâmetros da postura do robô em intensidade crescente, as janelas relativas a cada marco vão crescendo, incorporando novas projeções presentes no quadro atual. As projeções candidatas a formarem um conjunto de associações com os marcos são aquelas que, após um crescimento das janelas, ficaram próximas à borda de alguma das janelas. Assim um novo conjunto de associações é formado sempre que um número mínimo de associações puder ser extraído após um crescimento e, uma vez um conjunto formado, pode-se aplicar a seus elementos restrições quanto à característica de cor.

6.6.4 Obtenção das observações esperadas a partir dos marcos

Após as etapas de associação, a resposta do modelo de observação, $h(\mathbf{x})$, é obtida a partir da estimativa da postura do robô na iteração atual e dos marcos aceitos pela etapa de associação entre marcos e retas verticais. A dimensão da observação varia de iteração para iteração, e é igual ao número de marcos observados disponíveis, n_{marcos} . Cada posição dos vetores de observação e da resposta do modelo de observação ($h(\mathbf{x})$) corresponde a um marco, sendo que a resposta de $h(\mathbf{x})$ para um marco é o $u_{vert,marco}$ já calculado para o marco, e a observação é a projeção associada ao marco na fase de associação, $u_{vert,reta}$.

A variância de cada observação é dada por $\sigma_{u_{vert}}^2$, e as observações são consideradas independentes entre si. A matriz de covariância das observações é dada então por uma matriz identidade multiplicada por $\sigma_{u_{vert}}$.

6.6.5 Jacobiano do Modelo de Marcos

Como pode ser visto no apêndice II.5.1, o Filtro de Kalman Estendido necessita da matriz $\mathbf{H}(\mathbf{n})$, obtida derivando a função $h(\mathbf{x}(\mathbf{n}))$ em função de $\mathbf{x}(\mathbf{n})$. A função $h(\mathbf{x}(\mathbf{n}))$ retorna um vetor coluna cujo tamanho é igual a n_{marcos} , e pode ser dividida em n_{marcos} funções $h^{(i)}(\mathbf{x}(\mathbf{n}))$, sendo que cada função $h^{(i)}(\mathbf{x}(\mathbf{n}))$ é responsável por determinar a projeção do marco i .

A função $h^{(i)}(\mathbf{x}(\mathbf{n}))$ para o modelo de marcos é uma composição das funções de projeção dada na equação 6.1 e da transformação de corpo rígido dada pela equação 6.2 para cada marco observado.

Partindo da equação 6.14 e substituindo $[x_2, y_2]^T$ pelas coordenadas do marco i , $[x_{marco}^{(i)}, y_{marco}^{(i)}]^T$, e $[x_{mov2}, y_{mov2}, \theta_{mov2}]^T$ pela postura do robô, $[x_{robo}, y_{robo}, \theta_{robo}]^T$, obtém-se $h^{(i)}(\mathbf{x}(\mathbf{n}))$:

$$h^{(i)}(\mathbf{x}(\mathbf{n})) = f_u \frac{\cos(\theta_{robo})(y_{marco}^{(i)} - y_{robo}) - \sin(\theta_{robo})(x_{marco}^{(i)} - x_{robo})}{\cos(\theta_{robo})(x_{marco}^{(i)} - x_{robo}) + \sin(\theta_{robo})(y_{marco}^{(i)} - y_{robo})} + u_0, \quad (6.21)$$

onde $\mathbf{x}(\mathbf{n}) = [x_{robo}, y_{robo}, \theta_{robo}]^T$.

A partir da equação 6.21, pode-se obter $\mathbf{H}(\mathbf{n})$, onde cada linha representa as derivadas parciais de $h^{(i)}(\mathbf{x}(\mathbf{n}))$ pelos componentes de $\mathbf{x}(\mathbf{n})$. Dado o marco i

representado por $[x_{marco}^{(i)}, y_{marco}^{(i)}]^T$, a derivada de $h(\mathbf{x}(\mathbf{n}))$ por $\mathbf{x}(\mathbf{n})$ é dada por:

$$\begin{aligned}\frac{\delta h(\mathbf{x}(\mathbf{n}))}{\delta x_{robo}} &= -f_u \frac{\text{sen}(\theta_{robo})Z + \cos(\theta_{robo})V}{Z^2} \\ \frac{\delta h(\mathbf{x}(\mathbf{n}))}{\delta y_{robo}} &= f_u \frac{\cos(\theta_{robo})Z - \text{sen}(\theta_{robo})V}{Z^2} \\ \frac{\delta h(\mathbf{x}(\mathbf{n}))}{\delta \theta_{robo}} &= f_u \left(\frac{V^2}{Z^2} + 1 \right) \\ V &= \cos(\theta_{robo})(y_{marco}^{(i)} - y_{robo}) - \text{sen}(\theta_{robo})(x_{marco}^{(i)} - x_{robo}) \\ Z &= \cos(\theta_{robo})(x_{marco}^{(i)} - x_{robo}) + \text{sen}(\theta_{robo})(y_{marco}^{(i)} - y_{robo}).\end{aligned}$$

V e Z são variáveis auxiliares para simplificar a visualização das derivadas.

6.7 Implementação - distância entre quadros processados

A escolha da distância percorrida entre quadros capturados para a correção da postura do robô deve balancear corretamente a necessidade de se corrigir a postura que vem sendo atualizada apenas com observações dos odômetros e o tempo de processamento que se pode utilizar sem prejudicar o funcionamento dos demais sistemas compartilhando o mesmo processador. Além desses fatores deve-se considerar também as condições de operação do modelo de observação: quanto maior a distância entre dois quadros capturados maior a chance do modelo de observação associar marcos a projeções erroneamente, além do que, se a distância entre dois quadros for muito grande (em especial a distância angular) aumenta-se a chance de cada marco ser visto apenas num número pequeno de quadros, diminuindo muito a eficácia do modelo.

Como nos experimentos realizados não existe nenhuma outra aplicação compartilhando o processador, apenas os critérios relativos à eficácia e desempenho do modelo de observação foram considerados. A distância mínima linear ($d_{captura}$) e angular ($\theta_{captura}$) entre quadros foi escolhida de tal modo que, entre correções na postura realizadas pelo modelo de observação da visão monocular, a incerteza acumulada na postura pelos odômetros pudesse ser visualmente perceptível nos experimentos. Para o robô se deslocando a uma velocidade média de $0,2 \text{ m/s}$, a distância linear escolhida (em metros) foi de:

$$d_{captura} = 0,35.$$

Considerando que antes de uma correção ser disponibilizada pelo modelo de

visão são necessários processar no mínimo três quadros, então, no momento em que ELViS começa a operar, uma correção do modelo de visão poderá estar disponível após cerca de 90cm serem percorridos.

Essa distância também é interessante para analisar a operação do modelo de observação. Distâncias muito menores não trariam ganhos proporcionais na correção da estimativa devido ao pequeno deslocamento médio em pixels das projeções de uma mesma reta vertical em quadros consecutivos, enquanto distâncias muito maiores torna pequeno o tempo de vida das retas verticais observadas, não permitindo gerar correções com grande qualidade. Além dos 35cm, foram realizados testes com distâncias de 20cm, 30cm, 40cm, e 60cm, e apenas com 60cm detectou-se uma queda no número de retas observadas por mais de três quadros consecutivos devido a maior perda de associações corretas nos algoritmos de associação de retas verticais a projeções e de retas verticais a marcos.

Na prática, a distância angular mínima não foi muito usada, pois o robô tende ou a rotacionar muito pouco enquanto se desloca; ou então a rotacionar grandes ângulos quando parado. No primeiro caso os quadros são capturados pela distância percorrida, não necessitando do gatilho angular. No segundo caso a captura de quadros fica bloqueada pela trava que impede a captura de quadros em rotações maiores que 15 graus para impedir o processamento de quadros distorcidos devido ao rápido deslocamento do robô. Além do que, ao rotacionar mais de 40 graus, nenhuma das retas verticais que estavam sendo observadas antes da rotação continuarão a serem observadas. Por esses motivos $\theta_{captura}$ não é usado.

6.8 Valores dos parâmetros usados no Modelo de Observação de Visão Monocular

Durante a descrição dos diversos algoritmos apresentados neste capítulo, uma série de parâmetros foi usada. A seguir os mesmos são exibidos numa tabela juntamente com o valor adotado para cada um deles, e na sequência é apresentada uma breve descrição de cada parâmetro e a explicação de como seu valor foi obtido.

lim_{start} , lim_{end} Usados na extração das projeções dos quadros capturados, são limiares dos valores dos pixels usados durante o afinamento das bordas ressaltadas na convolução do quadro com o filtro apresentado na equação 6.4. Valores obtidos em experimentos realizados durante a fase de imple-

Parâmetro	Valor	Unidade
lim_{start}	150	intensidade do pixel
lim_{end}	80	intensidade do pixel
Lh	3	pixels
$range_{deriv}$	2	pixels
lim_{deriv}	2	pixels
u_{prox}	1	pixels
m_{prox}	0.1	–
lim_{numpix}	20	pixels
m_{min}	-0.07	–
m_{max}	0.07	–
$n_{lateral}$	4	pixels
α_{prox}	0.01	–
x_{min}	150	mm
x_{max}	25000	mm
c_{janela}	2	pixels
C_{min}	60	–
m_{projs}	3	–
u_{extra}	64	pixels
$du_{shift,1}$	3	–
$du_{shift,2}$	6	–
$du_{shift,3}$	12	–
$du_{shift,4}$	24	–
$uShift_{att,lim}$	7,5	–
$uShift_{mult}$	100	–
$N_{proj,assoc}$	2	–
$N_{marco,assoc}$	2	–

Tabela 6.1: Parâmetros

mentação do modelo de observação.

Lh Usado na extração das projeções dos quadros capturados, é o valor adotado para o número de pixels que devem ser observados antes de interromper a geração de uma sequência a partir da imagem pré-processada.

$range_{deriv}$ Usado na extração das projeções dos quadros capturados, este parâmetro define o número de pixels de uma cadeia que são considerados para determinar a inclinação do segmento em torno de cada pixel com objetivo de determinar se a cadeia pode ser considerada ou não um segmento de reta, considerando que todos os pixels de um segmento precisam apresentar inclinação similar. Quanto menor o valor deste parâmetro, mais rigoroso será o algoritmo para aceitar uma cadeia de pixels como uma reta vertical. Quanto maior ele for, mais ondulações locais são aceitas nas cadeias de pixels que representam os segmentos. O valor do parâmetro foi escolhido

empiricamente.

lim_{deriv} Usado na extração das projeções dos quadros capturados, este parâmetro indica o quanto a inclinação local dos pixels de uma cadeia podem variar para que a cadeia possa ser considerada um segmento de reta. O valor do parâmetro foi escolhido empiricamente.

u_{prox}, m_{prox} Usados na extração das projeções dos quadros capturados, são limites que indicam que, se as diferenças entre os valores dos parâmetros u e m , respectivamente, de dois segmentos ficarem abaixo de u_{prox} e m_{prox} , então os segmentos fazem parte da mesma projeção de reta vertical. Os valores desses parâmetros foram escolhidos empiricamente de modo a se aproximar ao seguinte comportamento: projeções de retas verticais distantes em até dois pixels não sejam mescladas, mas projeções distantes em apenas um pixel o sejam, considerando variações de inclinação de até 5 graus.

lim_{numpix} Usado na extração das projeções dos quadros capturados, determina o número mínimo de pixels necessários para que um conjunto de cadeias de pixels que representa uma reta vertical seja aceito. Quanto menor o valor, mais distantes podem ser localizar da câmera as retas verticais (gerando portanto projeções menores), porém mais ruído será aceito também como reta vertical. Foi determinado empiricamente que projeções ocupando a partir de 12,5% da altura do quadro poderiam ser consideradas como retas verticais produzindo muito pouco ruído.

m_{min}, m_{max} Usados na extração das projeções dos quadros capturados, delimitam a faixa de inclinações que um conjunto de cadeias de pixels que representa uma reta vertical pode apresentar para ser considerado uma reta vertical. Foi observado em experimentos que as projeções das retas verticais apresentam inclinações que podem desviar em até 5 graus dos 90 graus, devido a variação da intensidade da projeção da reta vertical ao longo de seu comprimento e principalmente a erros de alinhamento da câmera e trepidação. Foi escolhido a faixa de modo a considerar desvios de até 4 graus de 90. Foi observado em experimentos que esse valor é o suficiente para aceitar mais de 95% das projeções corretas que chegam até essa etapa.

$n_{lateral}$ Usado para obter os *PMCs* das projeções. O tamanho dos vetores que compõem o *PMC* são dados por $1 + 2 \cdot n_{lateral}$. O tamanho do vetor foi escolhido de modo a conter informação tanto da transição fotométrica que representa a reta vertical, como de um pequeno pedaço das superfícies que

seguem a transição de ambos os lados. O vetor não pode ser muito grande para não prejudicar o tempo de execução do algoritmo nem a sua corretude, pois quanto maior o vetor, maior a chance de ele conter informação sobre alguma interferência temporária, que pode estar presente num quadro de um modo, e no seguinte de outro.

α_{prox} Determina o espalhamento dos *pontos sigma* na UT usada para determinar a janela de busca espacial. Quanto menor α , mais próximos os *pontos sigma* ficarão e menor o risco de inconsistências serem geradas. Porém α muito pequeno afeta a capacidade da UT em estimar corretamente a covariância das variáveis. Como o desejado é apenas criar uma janela de busca e não obter a covariância precisa das variáveis, foi escolhido um valor de α pequeno o suficiente para evitar a maioria das inconsistências.

x_{min}, x_{max} Para os valores dos afastamentos máximos e mínimos arbitrados nas projeções para determinar a janela do seu deslocamento foi levado em conta a resolução e o ângulo de visão da câmera. Elementos muito distantes não conseguem gerar contraste o suficiente para formar projeções que passem pelas restrições das etapas de extração. Elementos muito próximos por sua vez, se não estão no caminho do robô (que logo seriam perdidos pela câmera pois o robô precisaria realizar grande mudança de orientação para desviar do obstáculo), não são observados pela câmera devido ao ângulo de visão da câmera.

c_{janela} Este fator determina o tamanho da janela de busca obtida quando mais de uma projeção estiver associada à reta vertical. Foi adotado um valor maior que um, que tende a ser menos restritivo e ajuda a compensar as aproximações realizadas na UT usada para obter a estimativa e covariância da posição da reta vertical no sistema de coordenadas da câmera no instante em que o último quadro antes do atual foi captura.

C_{min} Usado como limiar de corte para rejeitar associações entre projeções do quadro atual e do quadro anterior usando a medida de similaridade C_{ij} baseada nos *PMCs*. Associações com valor de C_{ij} menor que C_{min} são rejeitadas. Sua determinação foi empírica, levando em conta a necessidade de se selecionar apenas pares com boa probabilidade de serem uma associação correta, ou seja, pares entre projeções visualmente parecidas.

m_{projs} Limita o número de projeções usadas para obter o *PMC* das retas verticais. Quanto mais projeções são usadas, menor o efeito do ruído no *PMC*.

Porém quanto mais projeções forem usadas, maior a probabilidade de se distorcer o *PMC* com observações tomadas sob condição de iluminação um pouco diferente, seja por alterações na iluminação ambiente, seja devido a refletividade das superfícies envolvidas.

u_{extra} Usado para determinar as projeções esperadas dos marcos que estariam sendo observados caso a estimativa da postura do robô seja a correta. Indica o quanto se permite que a projeção esperada de um marco potencialmente observado fique distante dos limites do plano de projeção. Foi escolhido aumentar o plano de projeção em 50%, sendo um aumento de 25% de cada lado, pois é um valor que consegue acomodar erros consideráveis na posição do robô, ou seja, todos os marcos observados na postura real terão suas projeções consideradas. Erros de orientação são mais complicados de acomodar, pois mesmo aumentando o plano de projeção ao infinito, apenas erros de orientação de 50 graus são acomodados completamente.

$du_{shift,1}, du_{shift,2}, du_{shift,3}, du_{shift,4}$ Usados durante a associação de retas verticais a marcos, du_{shift} indica a diferença-limite entre os u_{shift} da primeira e da última associações de uma sequência de associações ordenada por u_{shift} , que é aceita para o próximo passo do algoritmo de associação.

$uShift_{mult}, uSshift_{att,lim}$ Usados durante a associação de retas verticais a marcos para determinar o critério de comparação entre conjuntos de associação, os moderadores $uShift_{mult}$, e $uSshift_{att,lim}$ foram determinados em uma série de testes enquanto se determinava a melhor forma de combinar os fatores na medida de similaridade.

A função de $uShift_{mult}$ é balancear a contribuição do critério criado a partir dos u_{shift} com a contribuição obtida a partir das características de cor. Já $uSshift_{att,lim}$ tem por função controlar a influência dos moderadores de $Var(u_{shift}^{(k)})$ (que são $u_{shift,max}$ e $u_{max,marco}^{(k)} - u_{min,marco}^{(k)}$).

$N_{proj,assoc}$ Usado durante a validação das associações realizadas, determina o número de vezes consecutivas que uma reta vertical deve ser associada a projeções. Quanto maior o valor de $N_{proj,assoc}$, maior a probabilidade da reta observada ser realmente uma reta vertical, e não algum ruído. Um número maior que dois para este parâmetro pode ajudar a eliminar retas verticais muito ambíguas que acabam recebendo associações ora de projeções corretas, ora de projeções de uma reta vertical próxima com características similares. Durante os testes realizados, porém, constatou-se que o algo-

ritmo de associação de retas verticais a projeções não necessita que este parâmetro seja maior do que dois, para os ambientes testados.

$N_{marco,assoc}$ Usado durante a validação das associações realizadas, determina o número de vezes consecutivas que um marco deve ser associado a uma reta vertical para que o marco possa ser usado na correção da estimativa. Quanto maior o valor de $N_{marco,assoc}$, menor o número de associações erradas serão usadas para corrigir a estimativa da postura, porém um número menor de vezes as associações corretas serão usadas para corrigir a postura. Foi constatado que duas associações consecutivas é o suficiente para filtrar as associações erradas. Com mais do que duas não foi detectado ganho nos testes realizados.

6.9 Resumo do Capítulo

Para cada quadro fornecido pela visão, são extraídas projeções no formato de retas verticais de elementos do ambiente e, usando o mapa do ambiente, determinado quais projeções extraídas correspondem a quais elementos mapeados. As associações indicam quais elementos do ambiente estão próximos ao robô, além de fornecer informação sobre a posição relativa dos elementos ao robô. Quando é possível associar um número mínimo de projeções à elementos mapeados por um número mínimo de iterações é possível determinar, a partir da postura estimada do robô, a posição estimada dos elementos sendo observados e, a partir das posições estimadas e das posições mapeadas dos elementos observados, a postura do robô é corrigida.

Para determinar as associações entre as projeções e os elementos mapeados é usado um processo de correspondência em duas etapas. Na primeira, projeções são associadas a entidades abstratas, referenciadas por retas verticais observadas. A cada nova iteração as projeções extraídas são associadas sempre que possível a retas verticais observadas já existentes. Caso não seja possível tal associação, uma nova reta vertical é criada, enquanto que as retas verticais observadas que não são associadas a nenhuma projeção são eliminadas.

Na segunda etapa da correspondência, as retas verticais observadas são associadas aos elementos mapeados, referenciados por marcos. As associações válidas a cada iteração são aquelas realizadas por duas ou mais iterações consecutiva entre um mesmo elemento mapeado e uma mesma reta vertical observada por duas ou mais iterações.

7 Implementação e Resultados Experimentais

Uma série de experimentos foram realizados para determinar a eficácia dos modelos de observação e dos estimadores selecionados que compõem o sistema ELViS. O objetivo primário dos testes é verificar a capacidade dos modelos de observação de sonar e visão em melhorar a estimativa da postura do robô, o ganho em acurácia em relação a um estimador *dead-reckoning* (que utiliza apenas os odômetros), e se a combinação de ambos os modelos de observação fornece alguma vantagem em relação ao uso dos modelos isolados.

Como objetivo secundário deseja-se determinar qual dos dois estimadores, o UKF ou o EKF, fornecem os melhores resultados e se a diferença entre eles é significativa, além de também avaliar o tempo de execução do ELViS.

Para ajudar a atingir esses objetivos foi desenvolvido um ambiente de testes que permite executar simulações usando dados reais ou artificiais para todas as combinações de estimadores e modelos sensoriais necessários aos experimentos.

Os dados artificiais são gerados a partir da postura real do robô e adicionando ruído gaussiano às observações obtidas. Simulações com dados artificiais são importantes para agilizar testes de algoritmos que compõem os modelos de observação e testes dos próprios modelos, permitindo analisar o desempenho de cada modelo e estimador em situações mais controladas.

A grande vantagem dos dados artificiais é a possibilidade de se comparar os resultados obtidos com os valores reais. O ponto fraco é o fato das observações serem geradas considerando a mesma modelagem de sensores utilizada nas construções dos modelos de observação e o mesmo modelo de perturbação (ruído gaussiano), o que faz com que os resultados obtidos com esses dados, apesar de servirem como indicadores, não possam ser usados para validar os modelos e estimadores.

Os dados reais são obtidos previamente, fazendo o robô se deslocar por um

determinado trajeto. Durante o percurso as observações dos sensores são armazenadas num arquivo, juntamente com o instante em que foram coletadas, e esse arquivo é usado no ambiente de testes para reproduzir a trajetória percorrida. Para simular o ambiente em que se espera que o estimador opere, durante a coleta dos dados usados neste capítulo uma pessoa ficou circulando, indo e voltando ao longo do corredor pelo seu centro. Durante os experimentos tomou-se o cuidado para que pelo menos dois quadros de vídeo com a pessoa fossem tratados pelo estimador. Como esperado, a pessoa não gerou retas verticais espúrias, sendo que o único prejuízo de sua presença foi a perda de algumas retas verticais oclusas por ela.

A posição real do robô é coletada através de marcações realizadas manualmente no chão durante o deslocamento do robô. Apesar de ser necessário validar os resultados usando dados reais, a capacidade de se determinar a acurácia dos resultados está limitada à acurácia da estimativa da postura inicial do robô, das posições do robô durante o percurso e da posição dos elementos do ambiente (mapa). Outra limitação em se trabalhar com dados reais é que é apenas capturada a posição real do robô, e não sua orientação.

Nos dados reais considerados, é esperada uma incerteza das medições de até 40 milímetros entre a posição medida do centro do robô e a sua posição real. Foi assumido que a incerteza associada aos dados é uma composição de duas incertezas não correlacionadas: a incerteza das marcações realizadas para determinar as posições do robô (tanto da criação das marcas quanto da medição de suas posições), cujo valor é de 25 milímetros, e da incerteza da posição de referências auxiliares, criadas para auxiliar a realização das medições, cujo valor é de 15 milímetros.

A incerteza dos dados pode ser interpretada do seguinte modo: a incerteza das marcações auxiliares contribuem para a incerteza do viés das medições (é o mesmo para todos os dados), enquanto que a incerteza das marcações podem ser interpretados com um espalhamento gaussiano sobre o valor real (varia de dado para dado).

7.1 Implementação do ambiente de testes

ELViS e o ambiente de testes foram implementados usando duas linguagens: Java e C++. Em C++ foram desenvolvidos a comunicação com o robô e programas de calibração e captura de imagens. Em Java foi implementado o modelo de

observação da visão, o modelo de observação dos sonares, o modelo de dinâmica, os estimadores EKF e UKF, a integração dos blocos e o ambiente de testes.

A escolha por implementar a maior parte do código em Java foi feita devido à maior agilidade que a linguagem oferece, provendo uma série de funcionalidades e facilidades que permitem o programador se dedicar mais à implementação dos algoritmos. As APIs gráficas disponíveis em Java também foram um motivador para a adoção da linguagem pois facilitou o desenvolvimento de um ambiente de testes gráfico possibilitando maior agilidade na análise dos resultados.

Foram implementadas tanto as versões originais do EKF e do UKF como as versões robustas de ambos, o EKF modificado mostrado nas equações II.20 e o SR-UKF (ver o apêndice II). Nos experimentos a seguir são mostrados resultados apenas para as versões robustas, porém foram realizados testes para comprovar que as versões de um mesmo estimador produzem resultados muito similares. Vale notar que nos experimentos realizados com as versões não estabilizadas não foi observada nenhuma situação que os levassem a divergir.

A figura 7.1 mostra um imagem do ambiente de testes sendo utilizado durante uma simulação de alguns estimadores.

Figura 7.1: Ambiente de testes durante uma simulação. Na figura pode-se ver uma simulação sendo executada, onde são comparados os resultados do UKF usando nenhum sensor (linha em vermelho) e usando a visão combinada com os sonares (linha em azul) com o deslocamento que realmente foi realizado (linha em preto). Na ponta de cada uma das linhas indicando o percurso percorrido até então encontra-se uma representação da postura real do robô, juntamente com uma elipse indicando a incerteza da estimativa fornecida pelo estimador. Na parte mais abaixo da figura encontra-se uma barra de progresso indicando a iteração da simulação sendo executada. A caixa azul indica os erros da posição na iteração corrente em relação a postura real para cada conjunto de sensores sendo simulados. Na parte superior encontram-se uma série de controles do simulador.

7.2 Experimentos

Neste trabalho são reportados resultados de experimentos tanto com dados artificiais como com dados reais. Cada conjunto de dados (reais e artificiais) origina análises diferentes.

Grande parte dos experimentos foram realizados nos corredores do prédio de engenharia elétrica da POLI-USP, segundo andar, bloco C. Esse ambiente foi escolhido por se enquadrar nas restrições de ambiente consideradas e por ser amplo o suficiente para que fosse possível notar a influência da inacurácia dos odômetros na estimativa da postura do robô.

O conjunto de dados reais que é analisado no texto foi coletado no ambiente citado. A figura 7.2 exibe o mapa com a parte do corredor utilizada na coleta dos dados, juntamente com a trajetória percorrida pelo robô durante a coleta dos dados. Vale ressaltar que foram mapeados apenas os elementos de interesse para os modelos de sonar e visão, ignorando elementos que esses modelos não vão considerar, como latas de lixo e bancos, por exemplo.

O ambiente mostrado na figura 7.2 tem aproximadamente 16 metros de comprimento por 4 metros de largura. O robô percorreu uma trajetória de aproximadamente 21 metros. A trajetória do robô foi obtida interpolando 45 posições mensuradas enquanto o mesmo se deslocava.

Usando o UKF foi obtida a estimativa da trajetória do robô usando apenas as observações dos odômetros. A figura 7.3 mostra a trajetória real e a obtida pelo ELViS usando dados reais e simulados. Para os dados reais, a trajetória comandada foi composta por três segmentos de reta; a figura 7.3 mostra em preto a trajetória de fato percorrida pelo robô ao considerar suas estimativas utilizando apenas os odômetros, as quais estão representadas na trajetória em vermelho.

Já para os dados simulados é observado o efeito contrário: a trajetória ‘real’ percorrida pelo robô é a mesma que trajetória comandada, mostrada em preto. A trajetória estimada, em vermelho, foi sensível aos erros de odometria, conforme o esperado.

Cada um dos segmentos que compõem a trajetória será tratado no restante dos experimentos como uma etapa da mesma. O trajeto é composto então por três etapas, a primeira se afastando da origem numa direção, a segunda na direção perpendicular à primeira, e a terceira paralela à primeira, porém em sentido contrário.

Figura 7.2: Mapa do ambiente de testes e trajetória real do robô. A figura retrata o mapa de um pedaço de um corredor do prédio de engenharia elétrica da POLI-USP, segundo andar, bloco C. Em preto é mostrado o contorno do ambiente. Os círculos vermelhos indicam as retas verticais mapeadas, os marcos. O triângulo representa o robô, enquanto o rastro preto é o caminho percorrido por ele.

A figura 7.4 exibe o erro entre a trajetória estimada e a trajetória real. O erro é exibido na forma da distância instantânea entre a estimativa da posição e a posição real do robô em cada iteração do estimador. Pode-se reparar que o erro tende a diminuir a partir da iteração 520. Essa iteração indica aproximadamente o início da terceira etapa da trajetória, quando o robô passa a caminhar em direção à posição inicial. O erro vai diminuindo até atingir o valor mínimo (iteração 800), próximo ao ponto em que a trajetória fica mais próxima da posição inicial, e a partir daí volta a crescer. Esse efeito é comum em trajetórias em circuito fechado ou semi-fechado.

O uso de trajetórias côncavas ou em circuitos fechados, como é o caso adotado, é interessante pois permite usar um espaço menor para percorrer distâncias mais amplas, porém pode prejudicar a análise do erro acumulado, em especial por

Figura 7.3: Trajetória da estimativa da posição com odômetros usando dados reais e simulados. A figura mostra a estimativa da posição do robô durante a trajetória. À direita está a estimativa obtida usando dados reais enquanto à esquerda está o resultado de uma simulação com dados artificiais.

sensores de movimento, pois o erro inserido por esses sensores tende sempre a crescer, e o efeito do circuito fechado tende a cancelar parte do erro acumulado. Já a análise de sensores de postura fica bem menos comprometida, pois essa correlação entre a distância da posição atual à posição inicial e a magnitude do erro se deve ao cancelamento parcial da incerteza (supostamente gaussiana) introduzida pelos odômetros durante os trajetos se afastando e se aproximando em direção à posição inicial. Com a introdução de correções independentes na postura, a estimativa é deslocada a cada correção, quebrando a correlação entre a distância da posição inicial e o erro acumulado.

Figura 7.4: Gráfico com o erro da estimativa do UKF usando apenas odômetros, com dados reais. A linha ‘erro d’ exibe a distância da estimativa à posição real do robô em cada iteração. A linha ‘erro x’ representa o erro na coordenada x, enquanto a linha ‘erro y’ representa o erro na coordenada y.

7.2.1 Dados Reais: Sonar

Figura 7.5: Trajetória da estimativa da posição com sonar e odômetros com dados reais, para EKF e UKF.

A figura 7.5 exibe o resultado da estimativa da trajetória do robô usando os sonares e os odômetros para os dados reais. São apresentados os resultados usando o EKF (em azul) e o UKF (em vermelho) como estimador. O gráfico 7.6 exibe o erro dos estimadores na forma da distância da posição estimada em relação à posição real. Analisando o gráfico de erro pode-se ver que o erro ficou próximo a 50 milímetros durante o primeiro segmento de deslocamento em linha reta. A partir do segundo deslocamento o erro sofreu uma variação abrupta atingindo 200 milímetros. Apesar de, pela figura 7.5, parecer que o erro das estimativas é muito pequeno na terceira etapa, na verdade o que ocorre é que a estimativa está defasada da posição real devido a um erro introduzido no final da primeira etapa.

O erro introduzido no final da primeira etapa é causado por uma região no ambiente que gerou muito ruído para os sonares frontais do robô. Essa região tem a forma de um ‘U’, sendo que um dos cantos do ‘U’ é composto pelo encontro de uma coluna com uma parede, e o outro canto por uma porta de vidro articulada semi-aberta, formando um ângulo de aproximadamente 90 graus. Essa região pareceu produzir múltiplas reflexões para as emissões dos sonares, fazendo as observações parecerem vir de mais longe, como mostra a figura 7.7.

Foi constatado que os sonares frontais estão sujeitos a esse tipo de situação com maior frequência que os sonares laterais, pois como o robô tende a se des-

Figura 7.6: Gráfico com o erro da estimativa da posição com sonar e odômetros com dados reais, para EKF e UKF.

Figura 7.7: A figura mostra a região em forma de 'U' que fornece observações enganosas ao sonar. O triângulo preto representa a posição real do robô, as linhas verdes representam observações dos sonares que detectaram algo e as vermelhas observações que não detectaram nada. A elipse azul representa a posição estimada já corrigida pelas observações dos sonares frontais.

locar muito em linha reta ou em trajetórias com inclinações suaves, os sonares frontais observam a mesma região por um período grande de tempo. Se a região que eles observam for muito ruidosa ou propiciar o tipo de problema encontrado

no ambiente de testes, eles vão gerar correções muito distorcidas. Já os sonares laterais não apresentam esse problema, pois mesmo que observem uma região problemática num dado momento, essa região será observada por um curto período de tempo e as validações do modelo de sonar vão perceber a anomalia e ignorá-la.

Figura 7.8: Trajetória da estimativa da posição com sonar e odômetros com dados reais, para EKF e UKF, ignorando os 4 sonares frontais.

Partindo dessa evidência, foi realizada mais uma execução do estimador com os dados reais, porém eliminando as medidas oriundas dos 4 sonares frontais. Os resultados estão nas figuras 7.8 e 7.9. Como se pode notar comparando os gráficos 7.9 e 7.6 os resultados são melhores, pois a região ruidosa não foi observada pelos sonares frontais (inoperantes, aqui).

Um detalhe que é importante ressaltar, porém, é que um fato não percebido nesse experimento devido à pequena distância percorrida é que, enquanto o robô se desloca em linha reta seguindo uma parede por um longo trajeto, o erro da posição do robô na direção paralela à parede aumenta, pois os sonares conseguem corrigir apenas a posição na direção perpendicular às paredes e outros elementos observados. Nesse experimento uma evidência de que isso estava ocorrendo pôde ser obtida através da inspeção da matriz de covariância da estimativa da postura, que apontava o aumento da incerteza da componente x da posição do robô ao acompanhar a parede na terceira etapa do trajeto.

Concluindo a análise, pode-se notar que os resultados obtidos pelo UKF e pelo EKF são muito próximos, e a diferença entre eles está dentro dos limites de incerteza das medições dos valores reais, apresentados anteriormente na introdução

Figura 7.9: Gráfico com o erro da estimativa da posição com sonar e odômetros com dados reais, para EKF e UKF, ignorando os 4 sonares frontais.

do capítulo.

As oscilações observadas nas figuras 7.6 e 7.9 devem-se às correções esporádicas efetuadas pelo estimador com os dados obtidos pelos sensores.

7.2.2 Dados Reais: Visão

A figura 7.10 exibe o resultado da estimativa da trajetória do robô usando o sensor de visão e os odômetros para os dados reais. São apresentados os resultados para o EKF e o UKF. O gráfico 7.11 exibe o erro dos estimadores na forma da distância da posição estimada em relação à posição real.

De novo pode-se notar que a diferença da estimativa entre o UKF e o EKF está bem abaixo da precisão considerada para os valores reais.

Pode-se notar também que a visão monocular, em média, fornece uma estimativa um pouco melhor que os sonares, sendo que em alguns momentos os sonares demonstraram obter uma estimativa melhor que a visão, em especial no fim do trajeto, onde a estimativa da visão parece rapidamente divergir da posição real.

Figura 7.10: Trajetória da estimativa da posição com visão e odômetros com dados reais, para EKF e UKF.

Esse efeito pode ocorrer quando o número de marcos observados fica muito reduzido, produzindo correções não tão adequadas. No caso do experimento, ocorreu uma combinação de dois efeitos. O estimador não foi capaz de corrigir sua postura com os marcos que estavam sendo observados no momento, e o robô escorregou um pouco para o lado. Isso levou a estimativa da postura a divergir da trajetória real, como se pode observar na figura 7.10. A visão monocular precisa observar um número mínimo de marcos para conseguir associá-los às retas verticais sendo observadas. Quando o número de marcos observados cai, chega-se num ponto em que o estimador não consegue mais corrigir a postura com grande acurácia. O gráfico 7.12 mostra o número de projeções extraídas, de marcos observados, de retas verticais sendo acompanhadas e de marcos sendo usados para corrigir a postura, por iteração.

Desse gráfico pode-se notar que, para cada reta vertical usada para corrigir a postura, é necessário observar em média 3 marcos, no ambiente do experimento. Ou seja, a cada marco utilizado para corrigir a postura, 2 são desperdiçados. Isso mostra que o algoritmo de associação entre retas verticais e marcos é muito restritivo. Por outro lado, foi realizado um experimento usando imagens de ambiente

Figura 7.11: Gráfico com o erro da estimativa da posição com visão e odômetros com dados reais, para EKF e UKF.

similar onde foi constatado que até 98% dos marcos usados para corrigir a postura foram associados corretamente.

Figura 7.12: Gráfico mostrando o número de marcos observados e projeções extraídas por iteração.

7.2.3 Comparação do UKF e EKF

Antes de continuar a análise dos dados reais, é comparado o desempenho dos estimadores, o EKF e o UKF utilizando dados simulados. Pelos gráficos 7.9 e 7.11, e considerando a incerteza associada à trajetória real, não se pode chegar a nenhuma conclusão. Mesmo ignorando a incerteza da trajetória real a diferença entre os dois estimadores é muito pequena em relação ao erro das estimativas.

Em outros experimentos com dados reais constatou-se a mesma dificuldade em apontar qual estimador teve melhor desempenho. Para comparar os estimadores são então usados dados artificiais.

Os dados artificiais têm como vantagem a capacidade de permitir realizar diversas simulações de um certo percurso, e a partir dessas várias realizações pode-se obter a média da estimativa da trajetória.

Por outro lado, os dados artificiais não contam com efeitos não esperados pelos modelos, como observações de sonares indicando os elementos mais próximos ou distantes do que o modelo prevê, por exemplo. Essa deficiência dos dados artificiais por um lado leva os resultados a superestimarem a acurácia do ELViS, porém, por outro lado, faz com que esses efeitos não modelados não interfiram na comparação dos estimadores.

O último ponto a se ressaltar no uso dos dados artificiais é o formato da distribuição das incertezas consideradas. Os dados artificiais consideram incertezas gaussianas independentes para cada observação (sonares, projeções de retas verticais e odômetros), porém na realidade a distribuição dessas incertezas não é conhecida, sendo a adoção delas como gaussiana apenas uma aproximação. Como ambos os estimadores têm seu melhor desempenho operando sobre variáveis com distribuições gaussianas, é esperado que a acurácia obtida nos experimentos com dados artificiais seja ainda mais superestimada.

Foram realizadas simulações usando a trajetória artificial mostrada na figura 7.3 para o EKF e o UKF usando apenas os 4 sonares laterais, apenas a visão monocular e combinando os 4 sonares laterais e a visão monocular. Para cada combinação de sensores e estimador a simulação foi repetida 50 vezes, executando sempre a mesma trajetória, porém variando o erro introduzido nas observações. A partir de cada conjunto de 50 realizações é possível determinar, para cada passo da trajetória, o *erro médio quadrático*, ou erro RMS¹.

¹O erro RMS é a raiz quadrada da média do quadrado das diferenças entre a variável e a sua estimativa: $\sqrt{\sum_i ((\bar{x}^{(i)}) - (x^{(i)}))^2}$

Os gráficos 7.13, 7.14 e 7.15 mostram as curvas de RMS, passo a passo, tanto do EKF como do UKF para os 4 sonares laterais, para a visão monocular e para a combinação dos 4 sonares laterais e da visão, respectivamente.

Figura 7.13: Gráfico com o erro RMS da estimativa da posição com 4 sonares laterais e odômetros com dados artificiais, para EKF e UKF.

Pode-se ver que nos dados simulados também a diferença entre o UKF e o EKF é pequena. Para os sonares a diferença é quase imperceptível, tendendo um pouco em favor do EKF, porém ficando em média abaixo de 1 milímetro (com picos de 2 milímetros), enquanto que o erro em relação à trajetória real ficou entre 20 e 120 milímetros.

Para a visão a diferença é maior, porém agora a favor do UKF. Em duas regiões da trajetória foi detectada uma diferença mais significativa, chegando a até 30 milímetros. Essas regiões ocorrem logo após uma região de crescimento continuado do erro das estimativas em relação à trajetória real. Após essas regiões ocorre um aumento súbito da diferença das estimativas do EKF e do UKF, que permanece elevada por 100 iterações e vai decaindo até ficarem próximas novamente.

A queda súbita da acurácia do EKF em relação ao UKF ocorre após períodos no quais a visão monocular não conseguiu corrigir a postura do robô, gerando

Figura 7.14: Gráfico com o erro RMS da estimativa da posição com visão monocular e odômetros com dados artificiais, para EKF e UKF.

um aumento na incerteza considerada sobre a postura estimada. Isso leva a crer que o UKF tem maior facilidade para tratar situações de maior incerteza do que o EKF, para o modelo de visão.

Apesar da diferença ser pequena, o UKF exibe melhores resultados que o EKF. Levando em conta o melhor desempenho nos testes de acurácia e os motivos secundários listados a seguir, o estimador adotado no ELViS é o UKF. Além da análise acima, os seguintes motivos também fazem a adoção do UKF ser mais atraente:

- O UKF é mais flexível que o EKF, permitindo uma série de adaptações e parametrizações que o permitem modelar melhor características particulares das *pdfs* consideradas. Além de se adaptar mais facilmente às distribuições, o UKF permite controlar o espalhamento das amostras usadas na transformação, podendo adaptar mais facilmente o estimador às não-linearidades de uma dada função.
- A construção do UKF não depende das funções não-lineares adotadas, podendo-se alterar a função sem necessitar alterar o estimador (a menos

Figura 7.15: Gráfico com o erro RMS da estimativa da posição com os 4 sonares laterais, visão monocular e odômetros com dados artificiais, para EKF e UKF.

de uma possível recalibração dos parâmetros de escala). Já o EKF depende das matrizes formadas pela derivação das funções não-lineares pelo vetor de estado. Assim, toda vez que for necessário alterar os modelos de observação e dinâmica, é necessário recalcular as derivadas.

- Estudos comparativos (JULIER; UHLMANN, 1997) indicam que a aproximação da transformação não-linear usada pelo UKF estima melhor a covariância da distribuição transformada, sem nunca a subestimar (considerar a distribuição mais concentrada do que realmente é). Subestimar a covariância faz com que o estimador atribua peso maior do que o devido a uma dada observação, o que não é desejável.

A partir desse ponto em diante serão exibidos apenas resultados usando o UKF.

7.2.4 Dados Reais: com uso de Visão e Sonares

O próximo experimento realizado combina as observações dos sonares e da visão monocular. Levando em conta as observações realizadas nos testes com sonares apenas, são ignorados os 4 sonares frontais do robô.

A combinação de observações de duas fontes sensoriais leva a uma redução da incerteza da estimativa da postura obtida, porém isso não significa necessariamente num aumento de acurácia da estimativa. Por exemplo, dois sensores são usados para determinar a estimativa do tamanho de um parafuso. O primeiro sensor detecta 10 centímetros, e o segundo 12 centímetros. Se a incerteza dos sensores for similar, a combinação resultará numa estimativa próxima a 11 centímetros. Caso o valor real seja 9 centímetros, pode-se ver que a combinação resultou numa estimativa pior que a apresentada pelo primeiro sensor.

Figura 7.16: Trajetória da estimativa da posição com visão, os 4 sonares laterais e odômetros com dados reais usando o UKF.

A principal vantagem que foi observada no uso da combinação é que a robustez dos resultados aumenta, e não a acurácia, minimizando erros grosseiros que um dos sensores pode vir a produzir. Considerando essa característica, é apresentado no gráfico 7.17 a curva com o erro da estimativa da combinação dos sonares e da visão juntamente com as curvas das estimativas usando apenas a visão e a

usando apenas os sonares. Na figura 7.16 são apresentadas as trajetórias relativas às curvas.

Figura 7.17: Gráfico com o erro da estimativa da posição com visão, os 4 sonares laterais e odômetros com dados reais para o UKF.

Pode-se notar que a curva de erro da combinação sensorial está na maior parte do trajeto entre ou abaixo das curvas obtidas apenas com a visão e apenas com os sonares. Uma outra característica notável é que a curva de erro da combinação parece acompanhar a curva obtida usando apenas a visão monocular. Para entender melhor esse efeito, é interessante observar os gráficos 7.18 e 7.19, que exibem as curvas de erro decompostas nas coordenadas x e y , respectivamente.

Desses gráficos pode-se ver que a estimativa da combinação sensorial está mais correlacionada à estimativa da visão na coordenada X , enquanto que na coordenada Y o erro, apesar de começar acompanhando mais os sonares, a contribuição de ambas as fontes sensoriais é mais equilibrada.

O comportamento visto na coordenada X vem do fato que as superfícies do ambiente usados pelo sonar para corrigir a postura do robô são paralelas a essa coordenada (durante a segunda etapa da trajetória quase nenhuma correção foi realizada), e o robô só é capaz de realizar correções na direção perpendicular à superfície observada. Isso significa que a estimativa da postura na coordenada X

Figura 7.18: Gráfico com a coordenada X do erro da estimativa da posição com visão, os 4 sonares laterais e odômetros com dados reais para o UKF.

é muito mais influenciada pelas correções da visão.

O erro na coordenada Y, a partir da iteração 400, também tende a seguir o erro da estimativa apenas com a visão, porém em menor intensidade. Esse efeito se deve ao fato que a incerteza da correção gerada pela visão é menor do que a gerada pelos sonares na segunda e terceira etapas da trajetória. Esse tópico será abordado novamente mais abaixo.

Caso os sonares frontais também estivessem sendo usados, a relação entre as curvas de erro seria similar, com alterações pontuais próximas às iterações 400 e 500, onde os sonares teriam a chance de captar paredes frontais pouco antes de realizar as duas curvas do trajeto.

Para mostrar a capacidade da combinação sensorial atenuar erros de um dos sensores usando o outro é mostrado um experimento usando os dados reais no qual a orientação da câmera sobre o robô foi deslocada em cinco graus. Cada grau de deslocamento faz as projeções ficarem em torno de 8 pixels deslocadas da posição real. Assim tem-se o sensor de visão gerando estimativas constantemente com erro, enquanto o sonar está gerando correções normalmente. Esse não é o

Figura 7.19: Gráfico com a coordenada Y do erro da estimativa da posição com visão, os 4 sonares laterais e odômetros com dados reais para o UKF.

melhor cenário para testar a capacidade da combinação sensorial em filtrar erros grosseiros, que só acontecem de vez em quando, mas serve para se observar melhor como as observações dos sensores são combinadas.

O gráfico 7.20 mostra as curvas de erro para a visão apenas, os 4 sonares laterais apenas, e para os sensores combinados. A figura 7.21 mostra as estimativas da trajetória para cada combinação de sensores.

Pode-se notar que a estimativa da trajetória ficou entre as estimativas dos sensores isolados.

É importante notar que a combinação sensorial é capaz de mesclar correções dos sensores na presença de alguma interferência em um deles, porém há um limite para a intensidade de distorção dentro do qual o estimador consegue obter a estimativa corretamente. Quando um dos sensores passa a distorcer muito as observações, constatou-se que em alguns casos o outro sensor em vez de corrigir a postura na direção correta pode tentar modificá-la para um valor que a afaste do real, porém que o modelo de observação considera que seja consistente com os dados observados. Por exemplo, se a orientação do robô estiver muito distorcida,

Figura 7.20: Gráfico com o erro da estimativa da posição com visão, os 4 sonares laterais e odômetros com dados reais para o UKF. Nesse experimento foi considerado um erro na calibração da orientação da câmera de cinco graus.

o modelo de observação dos sonares pode tentar associar as observações a paredes que apresentem inclinações perpendiculares à parede que realmente produziu a observação.

Já para o modelo de visão, se a orientação estimada do robô estiver afastada mais que 40 graus (que é o ângulo de visão da câmera) da orientação real do robô, dificilmente algum dos marcos que estão sendo observados de fato farão parte do conjunto de marcos que o modelo seleciona para dar início ao processo de associação.

7.2.5 Sensibilidade à posição inicial

Para determinar o limite de erro na postura do robô a partir do qual o estimador deixa de fornecer resultados confiáveis foram realizados experimentos variando a estimativa da postura inicial do robô. Em todos os resultados apresentados até agora a posição inicial foi considerada sempre conhecida, gerando trajetórias cuja curva de erro iniciava sempre do zero.

Figura 7.21: Trajetória da estimativa da posição com visão, os 4 sonares laterais e odômetros com dados reais usando o UKF. Nesse experimento foi considerado um erro na calibração da orientação da câmera de cinco graus.

As curvas de sensibilidade em relação à estimativa da postura inicial para a visão, para os 4 sonares laterais e para a combinação dos dois são exibidas nos gráficos 7.22, 7.23 e 7.24 respectivamente. Cada um mostra uma sequência de curvas onde, em cada uma, o erro aplicado à estimativa da postura inicial é aumentado um pouco em relação à curva anterior. A cada curva a estimativa da postura inicial foi afastada da postura real em 100 milímetros no sentido crescente da coordenada y e rotacionada em 4 graus no sentido horário. Foram geradas curvas até nenhum dos dois sensores ser capaz de corrigir a estimativa da postura do robô durante a primeira etapa do trajeto.

É interessante notar como, nessas condições, a combinação sensorial realmente gera um resultado melhor do que aqueles dos sensores isolados. Isso mostra que, em casos onde o estimador produz uma estimativa com um erro muito maior do que os sensores normalmente produziriam (por erros grosseiros dos sensores ou por reposicionamento do robô, por exemplo), a combinação dos sensores de visão e sonar tem potencial de se recuperar num menor número de iterações do que os sensores de forma isolada.

Figura 7.22: Curvas de erro da trajetória do estimador usando os 4 sonares laterais. As curvas mostram a sensibilidade do estimador à variação da postura inicial. Cada curva foi gerada com a postura estimada inicial distante dos valores indicados na legenda da postura inicial real, $\sqrt{x^2 + y^2}$ à esquerda, e θ , em graus, à direita.

7.2.6 Tempo de Execução

O tempo de execução é uma característica importante dos algoritmos acima, pois é desejado que a solução, além de fornecer uma boa estimativa da postura, possa ser usada em equipamentos de custo menor, implicando em menor capacidade de processamento disponível.

Na tabela 7.1 estão listados os tempos de execução médio de uma iteração em que as seguintes combinações de estimadores e sensores foram executadas:

- EKF + 4 sonares laterais
- UKF + 4 sonares laterais
- EKF + visão monocular
- UKF + visão monocular

Figura 7.23: Curvas de erro da trajetória do estimador usando visão monocular. As curvas mostram a sensibilidade do estimador à variação da postura inicial. Cada curva foi gerada com a postura estimada inicial distante os valores indicados na legenda da postura inicial real, $\sqrt{x^2 + y^2}$ à esquerda, e θ , em graus, à direita.

Como explicado no capítulo 3, ELViS utiliza a técnica SCAAT para realizar a combinação sensorial, e nessa técnica as observações de cada sensor são tratadas em uma iteração distinta, mesmo quando as observações são coletadas no mesmo instante. Então o tempo de execução combinado da visão monocular e dos sonares é a soma dos tempos de execução dos tratamentos em separado.

Sensores / Estimador	Tempo em milisegundos	Tempo normalizado
EKF + 4 sonares laterais	11	0,37
UKF + 4 sonares laterais	9	0,30
EKF + visão monocular	91	3,03
UKF + visão monocular	88	2,93

Tabela 7.1: Tempos de Execução

O processador utilizado durante os testes foi um Athlon XP 2000+ (equivalente a um Pentium 4 com frequência de 1,67GHz), com 256KB de cache L2 e 512MB de RAM. Para tentar diminuir a dependência dos resultados em função do

Figura 7.24: Curvas de erro da trajetória do estimador usando visão monocular e os 4 sonares laterais. As curvas mostram a sensibilidade do estimador à variação da postura inicial. Cada curva foi gerada com a postura estimada inicial distante os valores indicados na legenda da postura inicial real, $\sqrt{x^2 + y^2}$ à esquerda, e θ , em graus, à direita.

hardware e linguagem de programação utilizados, uma terceira coluna na tabela exibe os tempos de execução normalizados. A normalização foi realizada com base no tempo de execução de uma operação específica que pode ser repetida facilmente em qualquer ambiente e linguagem de programação.

A operação escolhida é o borramento de uma imagem usando uma matriz com 3 colunas e 3 linhas, chamada de filtro. O borramento é realizado convoluindo a matriz que representa a imagem com o filtro. A matriz da imagem usada para obter o fator de normalização tem 320 colunas e 240 linhas (do mesmo tamanho das imagens geradas pela câmera). O tempo de execução médio obtido para essa operação foi de 30 *ms*.

Foi escolhida uma operação de processamento de imagens como base de tempo pois foi constatado que o maior gasto em tempo de execução ocorre durante o processamento dos quadros.

É importante notar que, apesar dos tempos por iteração mostrados na tabela

representarem o tempo necessário para processar os dados de um dado sensor, não é em toda iteração do estimador que os sensores geram observações. A tabela 7.2 mostra o número médio de dados que são processados por sensor durante os testes com dados reais realizados. Vale notar que os dados relativos aos sonares indicam o número de observações que apenas um sonar gerou. Para obter todas as observações dos oito sonares é necessário multiplicar o número por 8.

Sensor	Num. de Observações	Dist. média entre observações (mm)
Sonar	95	215
Visão monocular	55	374
Odômetros	1139	18

Tabela 7.2: Observações coletadas por sensor

Desses resultados pode-se destacar que o tempo de execução está bem reduzido, e pode ser ainda mais reduzido apenas portando a implementação totalmente para linguagem C ou C++. Usando a experiência adquirida em outras situações similares pode-se dizer que apenas por portar para C++ o código, o tempo de execução seria reduzido para pelo menos 50% do original, porém podendo chegar a até 10% para algumas operações específicas.

Considerando que o tempo de execução deixa de ser um limite crítico para o algoritmo, tem-se espaço para introduzir no futuro estimadores mais robustos, como o Filtro de Partículas, ou então realizar validações mais complexas entre os marcos e as retas verticais, visando uma perda menor de associações válidas. Ou ainda pode-se usar imagens com resolução maior e câmeras com maior abertura para poder observar regiões maiores do ambiente, ou então reduzir a distância entre quadros processados pelo sistema.

Todas essas considerações devem ser tomadas considerando o processador que estará disponível para o uso do algoritmo e a carga extra que será aplicada sobre ele (outros sistemas compartilhando processamento). Observa-se que o processador usado nos experimentos ainda não pode ser considerado acessível para aplicações de baixo custo, porém espera-se que num futuro próximo (em torno de cinco anos) os processadores de baixo custo estarão operando nessa faixa de processamento.

7.2.7 Comparação com Trabalhos Similares

Comparando os resultados obtidos com outros trabalhos pode-se notar que os resultados em termos de acurácia são similares. Por exemplo, Schroeter (2005)

combinou um anel de 24 sonares e visão omni-direcional e obteve um erro médio de posição de 24 centímetros, enquanto Thrun (2001) obteve erros de estimativa em torno de 10 centímetros durante testes de desempenho de um algoritmo de localização usando Filtro de Partículas e um anel de 48 sonares. Outros autores que usaram laser em vez de sonar como sensor de profundidade conseguiram obter resultados com acurácia maior. Considerando a incerteza associada à medição da postura real, pode-se dizer que os resultados aqui alcançados estão próximos aos reportados na literatura.

Durante a execução dos experimentos e testes do estimador sentiu-se dificuldades em produzir comparações quantitativas entre métodos apresentados na literatura e o apresentado neste trabalho. A dificuldade tem origem na grande variedade de plataformas robóticas e combinações de sensores usados na literatura, diferentes objetivos de cada trabalho, além da falta de um protocolo padrão de experimentos que permitam comparar diretamente os resultados de diferentes trabalhos. A solução seria reimplementar alguns dos trabalhos apresentados que utilizassem os mesmos sensores usados neste trabalho e tentar reproduzir seus experimentos.

Foi decidido que não seriam implementados testes comparativos com outros trabalhos pois nenhum dos trabalhos encontrados possuía a mesma combinação de sensores e mesmo objetivo do trabalho corrente (localização). Além disso, o objetivo principal desse trabalho não é obter o estimador mais acurado, mas sim um estimador capaz de determinar a postura do robô com boa acurácia e confiabilidade nas condições predeterminadas.

7.3 Análise Geral

A partir dos experimentos aqui reportados, a primeira constatação que pode ser feita é que o sensor de visão produz em geral resultados melhores que os sonares. Isso foi constatado em alguns outros experimentos envolvendo outros conjuntos de dados coletados em ambientes controlados. Porém a visão necessita de um número razoável de marcos dentro do campo visual para operar. Já os sonares, para os ambientes testados e nos ambientes considerados em geral, tem grande probabilidade de ter ao seu alcance pelo menos uma parede com superfície desobstruída o suficiente para gerar pelo menos algumas correções para a posição do robô.

A combinação dos dois sensores permite combinar a maior acurácia da visão

monocular com a maior constância dos sonares. Para momentos em que nenhum dos dois sensores consegue corrigir a postura do robô, foi observado que a estimativa obtida apenas pelos odômetros em deslocamentos de até 8 metros produzem estimativas com erro dentro de um limite seguro no qual a combinação da visão com os sonares consegue produzir a estimativa da postura com boa acurácia em poucas iterações.

Foi observado que em alguns casos as observações dos sonares são assimiladas em menor grau, comparando-se à assimilação das correções originadas pela visão monocular. Isso ocorre porque a incerteza das correções obtidas através da visão é menor do que a incerteza obtida pelos sonares. Essas incertezas são determinadas pelas transformações que o estimador (EKF ou UKF) realiza, mas têm origem na incerteza das observações dos sensores. Isso pode gerar um certo atraso entre novas observações dos sonares serem obtidas e suas correções surtirem efeito na postura do robô quando são usados tanto os sonares quanto a visão.

Quando a correção de uma observação é integrada à estimativa da postura, sua incerteza sofre uma redução que é tão maior quanto menor for a incerteza da correção. Então, uma estimativa recentemente corrigida por observações da visão pode ficar temporariamente imune a correções do sonar. Esse pode ser um efeito prejudicial em casos em que a visão não consiga fornecer estimativas por um certo período. Para minimizar esse efeito foram realizados alguns experimentos onde as incertezas atribuídas aos odômetros foram modificadas artificialmente. Descobriu-se que elevando a incerteza atribuída aos odômetros pode-se fazer o estimador ficar mais susceptível às correções das observações. Isso pode ser benéfico no caso da estimativa da localização do robô estar errada por algum motivo (travamento de uma roda, escorregamento, colisão, reposicionamento, etc.). A contrapartida, porém, é que a estimativa fica mais susceptível a erros grosseiros gerados pelos sensores e modelos de observação.

O mesmo tipo de manipulação pode ser realizado também nas incertezas dos sonares e da visão, com o objetivo de aumentar ou diminuir artificialmente sua contribuição à estimativa.

8 Conclusão e Trabalhos Futuros

Foi apresentado neste trabalho um estimador para a postura de um robô que se desloca por um ambiente conhecido. O estimador apresentado faz uso de três conjuntos de sensores: os odômetros, que monitoram o deslocamento do robô a partir de observações das ações realizadas, um conjunto de oito sonares atuando como sensores de profundidade e uma câmera de vídeo atuando como visão monocular.

Foi mostrado nos experimentos que, apesar dos odômetros fornecerem uma boa estimativa da postura durante deslocamentos pequenos, durante deslocamentos maiores é necessário uma fonte sensorial adicional para manter a estimativa próxima da postura real. Outros experimentos mostraram que tanto os sonares como a visão monocular são capazes de cumprir essa função, e que o uso das duas simultaneamente ajuda a obtenção de resultados mais confiáveis, porém não necessariamente mais acurados.

Além do estimador, foram apresentados os algoritmos usados para interpretar as observações da visão e dos sonares. Ambos os algoritmos apresentados priorizam a rejeição de observações que poderiam levar a atualizações erradas na estimativa da postura, podendo com isso sacrificar algumas observações que poderiam levar a atualizações corretas.

Os experimentos demonstraram também que o tempo de execução do estimador não impede seu uso em situações onde é necessário estimar a postura em tempo real, ou seja, num tempo pequeno o suficiente para que o robô possa usar essa informação para preparar uma reação sem que isso o faça parar e esperar a determinação da postura.

8.1 Discussões

No trabalho foi mostrada a viabilidade de se usar sonares esparsos e um número limitado de quadros capturados de uma câmera de vídeo para determinar a localização do robô num ambiente conhecido. Além disso foi mostrado que é possível usar apenas um dos conjuntos de sensores (os sonares ou a câmera) para obter resultado similar, em troca da confiabilidade extra que o uso de ambos os sensores fornece.

Esses são resultados interessantes, pois demonstram a capacidade de se determinar a localização do robô usando relativamente pouca informação em comparação com outras técnicas apresentadas na literatura.

Essa capacidade, aliada ao fato de que o estimador consegue se recuperar de erros de mais de meio metro e de 30 graus na postura (demonstrado pelo experimento da variação da estimativa da postura inicial do robô), abre espaço para uma redução ainda maior na quantidade de informação que necessita ser tratada para determinar a postura do robô a fim de obter um estimador com tempo de execução ainda menor ao custo de alguma redução da acurácia.

Indo nessa direção ao outro extremo, pode-se concluir dos experimentos que o estimador atendeu às expectativas e obteve uma boa estimativa da postura do robô num ambiente interno semi-estático (com uma pessoa se movendo). Apesar do tempo de execução estar dentro do esperado, ele pode ser diminuído ainda mais implementando todo o estimador em C ou C++, permitindo o aumento de complexidade dos modelos de observação, em especial o de visão, inserindo tratamentos mais sofisticados nas associações de dados ou mesmo observando outros tipos de elementos com o objetivo de obter a postura com maior acurácia.

Vale lembrar que a capacidade de se reduzir o tempo de execução ou então de aumentar a complexidade dos modelos e do estimador está vinculada à capacidade do equipamento disponível e à velocidade de deslocamento do robô. Quanto mais rápido o robô se desloca, mais freqüentemente se darão os ciclos de operação do modelo de observação de visão.

8.1.1 Estimadores

Analisando os resultados obtidos com o EKF e o UKF pôde-se concluir que os dois estimadores produzem praticamente os mesmos resultados para a aplicação corrente. Esse resultado não era esperado, pois considerava-se que o UKF seria

significativamente melhor. Porém, este resultado também não chegou a causar surpresa devido a resultados divergentes encontrados na literatura.

8.1.2 Sonares

Os resultados obtidos com os sonares foram satisfatórios, porém, com base nos experimentos, pôde-se constatar que, pelo padrão de deslocamento adotado (trajetórias basicamente retilíneas intercaladas com rotações sobre o centro de massa) o modelo de observação pode fazer com que as observações dos sonares frontais acumulem erro, e não o filtrem, como era esperado. Para evitar esse problema com os sonares frontais, optou-se por ignorar as observações fornecidas pelos quatro sonares frontais e trabalhar apenas com as observações dos quatro sonares laterais.

O problema de adotar apenas os sonares laterais é que, enquanto o robô percorre uma trajetória retilínea, não será possível corrigir sua posição na direção paralela ao deslocamento, e, como foi observado nos experimentos, o erro passa a se acumular nessa direção à medida que o robô se desloca, até o mesmo mudar de direção.

Porém, ao ignorar os sonares frontais, não se está desperdiçando muita informação, pois eles só vão detectar algo que os sonares laterais não detectaram no caso da aproximação de um obstáculo frontal. O número de leituras que o sonar seria capaz de obter antes do robô ter de desviar seria apenas suficiente para poucas correções na postura (somente uma, observada na trajetória percorrida nos experimentos).

Apesar de não essenciais para a estimativa da postura, ao desenvolver um sistema de navegação completo para um robô, os sonares frontais não são dispensáveis pois são fundamentais para o mecanismo de detecção de colisão, importante em qualquer sistema de controle de um robô móvel. Esse tipo de detecção não precisa necessariamente de uma estimativa exata do obstáculo, basta saber que o mesmo se aproxima e que deve-se desviar.

8.1.3 Visão monocular

Os resultados obtidos para a visão monocular também ficaram dentro do esperado. Esperava-se um desempenho similar aos sonares, e o que se constatou foi um desempenho um pouco melhor, porém muito dependente da quantidade de marcos existentes no ambiente.

A maior dificuldade em se trabalhar com o modelo de observação de visão monocular foi em determinar os marcos que iriam ser mapeados e obter seus vetores de características. Um número razoável de marcos sendo observados é necessário para o bom funcionamento da visão monocular. A determinação dos marcos implica na escolha de retas verticais cujas projeções sejam razoavelmente invariantes em relação à posição da qual são observadas e a pequenas variações de iluminação, além de ser interessante escolher marcos cujas projeções não sejam muito parecidas àsquelas dos outros marcos próximos.

Notou-se ainda que a acurácia da determinação da sua posição no ambiente não é crítica para o bom funcionamento do sensor, uma vez que um pixel de imagem pode representar uma área de alguns centímetros quadrados até algumas dezenas de centímetros quadrados do ambiente. No caso dos sonares a acurácia dos resultados obtidos varia quase que linearmente com a acurácia da posição dos elementos mapeados. Essa análise mostra que o sensor de visão é menos sensível que o sensor de sonar a erros de mapeamento, porém isso indica que, no caso de uma futura extensão do algoritmo de localização para um algoritmo de localização e mapeamento, o modelo de visão adotado irá gerar mapas menos precisos do que o modelo de sonares.

Notou-se também que a característica usada para determinar as associações entre marcos e retas verticais sendo observadas é muito restritiva, rejeitando excessivamente associações corretas, em especial quando a disparidade da postura real em relação à estimada é grande na direção do deslocamento retilíneo do robô. Esse tópico foi abordado na seção 6.6.3 do capítulo 6, onde também foram apresentadas propostas para diminuir essa perda em trabalhos futuros.

8.2 Trabalhos Futuros

O trabalho apresentado tem três linhas distintas que dão espaço a melhorias e modificações a serem realizadas em trabalhos futuros: o estimador responsável por combinar as observações dos sensores, o tratamento dos sonares e o tratamento da visão monocular.

Além das três linhas principais, um trabalho futuro possível de ser produzido a partir desse é a adaptação do algoritmo de localização para um algoritmo de localização e mapeamento (SLAM, da sigla em inglês). Para tal, os modelos de observação precisariam ser revistos, porém a maior parte do trabalho deve se concentrar na alteração do estimador, de modo que ele passe a estimar tanto a

postura do robô quanto dos elementos do ambiente.

Outro ponto importante que vale a pena dedicar algum esforço é na alteração do modelo de dinâmica, de tal modo que o mesmo passe a funcionar usando como entrada apenas os sinais de controle enviados ao robô, como a tensão aplicada nos motores, por exemplo. Desse modo as observações dos odômetros poderiam dar origem a um terceiro modelo de observação. Para isso ser possível, seria necessário um modelo mais completo e fiel da dinâmica do robô, assim como adicionar ao vetor aleatório sendo estimado outras variáveis além da postura, como as velocidades linear e angular do robô, por exemplo.

8.2.1 Estimadores

Uma evolução no estimador seria a substituição do UKF por um Filtro de Partículas (descrito sucintamente no apêndice II). Thrun (2001) reportou que algumas variações de Filtros de Partículas usadas por ele já podem ser usadas em tempo real com o hardware atual. Os Filtros de Partículas permitiriam a recuperação da postura do robô em caso de desconhecimento da postura inicial, além de potencialmente permitir obter estimativas mais acuradas.

É importante notar, porém, que a adoção de um Filtro de Partículas deve implicar na revisão dos modelos de observação, pois o Filtro de Partículas irá operar com dezenas, centenas ou mesmo milhares de partículas, e cada partícula necessita que seja mantida uma instância independente do modelo de observação, o que pode tornar seu uso inviável devido ao grande volume de processamento que será gerado.

8.2.2 Sonares

Uma melhoria muito interessante que poderia ser implementada é o uso da informação dos demais sonares, e não só de cada sonar independentemente, como realizado no trabalho, durante a etapa de associação entre a observação e o elemento sendo observado. Usando as observações mais recentes dos demais sonares pode-se obter um mapa local dos arredores do robô, ajudando a determinar qual superfície tem maior probabilidade de estar sendo observada, eliminando algumas ambigüidades entre superfícies paralelas próximas.

Além das observações dos demais sonares, pode-se usar também os marcos sendo observados pelo modelo de visão. Uma vez que se sabe que um conjunto de marcos está sendo observado e que esses marcos pertencem a determinadas

superfícies, pode-se usar essa informação para reduzir as ambigüidades. É importante notar que essa alteração criaria uma dependência entre os dois modelos sensoriais. Apesar das correções continuarem a ser geradas de forma independente, o uso dos marcos para determinar a superfície observada pelo sonar pode fazer com que um erro grosseiro que eventualmente tenha afetado o modelo de observação de visão contamine também o modelo de observação do sonar.

8.2.3 Visão monocular

O modelo de visão apresenta um método robusto e de baixa complexidade computacional para determinar a postura do robô a partir da observação de retas verticais. Porém, para aumentar a robustez, um número considerável de marcos que poderiam ser usados é desperdiçado. Conforme discutido na seção 6.6.3 do capítulo 6, para aumentar o aproveitamento poder-se-ia adotar, na etapa de associação de marcos às retas verticais, janelas espaciais de busca, assim como é feito na associação das retas verticais sendo observadas com as projeções. Desta forma, a partir das estimativas das projeções dos marcos, poder-se-ia determinar com maior correteude as associações potenciais.

Outra melhoria que pode ser adotada diz respeito à determinação dos parâmetros de calibração da câmera. Visando uma aplicação comercial, é possível contruir um robô com uma câmera acoplada cuja posição relativa ao robô e orientação fiquem fixas, e provavelmente com acurácia no alinhamento maior do que a obtida na calibração realizada neste trabalho. Porém a distância focal da câmera pode variar com o uso, não se podendo garantir que a calibração vai manter o mesmo valor ao longo de sua utilização.

Para minimizar esse problema, Chiuso (2002) considera o foco como uma variável aleatória e o adiciona ao estado sendo estimado. Assim, além de se estimar a postura do robô também se estimaria o foco da câmera.

Como já dito, uma das maiores complicações no uso do modelo de observação adotado é a necessidade de se determinar as estruturas e padrões do ambiente que serão usados como marcos. Novamente pensando em uma solução comercial, não bastaria ter a planta do ambiente de onde o robô se deslocaria, seria necessário ir até o local, determinar os marcos manualmente, além de capturar suas imagens para gerar as Médias de Cores (MC).

Uma melhora seria a adoção de um sensor alternativo que trabalharia sem usar marcos. O sensor usaria uma técnica conhecida como Estrutura a partir

do Movimento (*Structure from Motion*, ou SfM, em inglês) onde seriam usadas apenas as retas verticais observadas para estimar a postura do robô. A técnica atua de modo similar à visão estereoscópica, porém a disparidade entre as imagens é dada pelo deslocamento do robô entre duas imagens capturadas. Parte da base para esse algoritmo já está pronta, dentro da etapa de associação de retas verticais a projeções, onde as posições das retas são estimadas para determinar janelas mais restritas de busca.

Essa técnica não poderia substituir um modelo que usa elementos do ambiente, pois, por não trabalhar com referências externas, ela sofre do mesmo problema que os odômetros. Porém, ela pode ser adotada como técnica auxiliar, permitindo diminuir o número de marcos mapeados no ambiente, ou mesmo eliminar esse modelo e usar apenas os sonares como sensores de postura absolutos.

Pode-se ainda aumentar a quantidade e a qualidade da informação disponível para o modelo de observação de visão operar, seja adotando câmeras com resolução maior ou mesmo usando um sistema de visão estereoscópica. Uma outra solução, porém em uma configuração pouco comum, seria usar duas câmeras paralelas à direção de deslocamento do robô, uma apontando num sentido e a outra no sentido oposto. Nessa configuração poder-se-ia realmente dobrar a quantidade de projeções usadas para a localização.

Uma última modificação que pode ser realizada com o objetivo de aumentar a quantidade de projeções disponíveis para o modelo tratar seria o uso de outros tipos de elementos além de retas verticais. Por exemplo, pode-se trabalhar também com retas horizontais no ambiente, que no plano de projeção geram retas que se intersectam num mesmo ponto (o ponto de fuga) ou então objetos que se destacam do ambiente por sua forma, textura ou cor. Nesse caso extintores de incêndio ou latas de lixo poderiam ser usados.

Bibliografia

AKBARALLY, H.; KLEEMAN, L. 3d robot sensing from sonar and vision. In: *Proceedings of the IEEE International Conference On Robotics and Automation*. Minneapolis, Minnesota, EUA: IEEE Computer Society Press, 1996. p. 686–691.

ANDRADE-CETTO, J.; VIDAL-CALLEJA, T.; SANFELIU, A. Unscented transformation of vehicle states in slam. In: *Proceedings of the IEEE International Conference on Robotics and Automation*. Barcelona, Espanha: IEEE Computer Society Press, 2005. p. 323–328.

ARRAS, K. O.; TOMATIS, N.; SIEGWART, R. Multisensor on-the-fly localization using laser and vision. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. Takamatsu, Japão: IEEE Computer Society Press, 2000. p. 131–143.

BALTZAKIS, H.; ARGYROS, A.; TRAHANIAS, P. Fusion of laser and visual data for robot motion planning and collision avoidance. *Machine Vision and Applications*, Springer-Verlag New York, Inc., v. 15, n. 2, p. 92–100, December 2003.

BAR-SHALOM, Y.; LI, X. R. *Multitarget-Multisensor Tracking: Principles and Techniques*. Storrs, CT: YBS Publishing, 1995.

BARRA, R. J. G.; COSTA, A. H. R. Correspondência rápida de retas verticais entre imagens para localização de robôs móveis. In: *Proceedings of the VII Simpósio Brasileiro de Automação Inteligente*. São Luís, MA, Brasil: Sociedade Brasileira de Automática, 2005. ISBN 85-85048-55-7.

BELLOTTI, N.; HU, H. Vision and laser data fusion for tracking people with a mobile robot. In: *Proceedings of the IEEE International Conference on Robotics and Biomimetics*. Kunming, China: IEEE Computer Society Press, 2006. p. 7–12.

BERGMAN, N. *Recursive Bayesian estimation: Navigation and tracking applications*. Tese (Doutorado) — Department of Electrical Engineering, Linköping University, Sweden, 1999.

BERTOZZI, M. et al. Low-level pedestrian detection by means of visible and far infra-red tetra-vision. In: *Proceedings of the IEEE Symposium on Intelligent Vehicles*. Tokyo, Japão: IEEE Computer Society Press, 2006. p. 231–236.

BILGIC, T.; TURKSEN, I. B. Model-based localization for an autonomous mobile robot equipped with sonar sensors. In: *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*. Vancouver, B.C. Canada: IEEE Computer Society Press, 1995. p. 3718–3723.

BING, M.; LAKSHMANAN, S.; HERO, A. A robust bayesian multisensor fusion algorithm for joint lane and pavement boundary detection. In: *Proceedings of IEEE International Conference on Image Processing*. Thessaloniki, Greece: IEEE Computer Society Press, 2001. v. 1, p. 762–765.

BORENSTEIN, J. et al. Mobile robot positioning: sensors and techniques. *Journal of Robotic Systems*, v. 14, n. 4, p. 231–249, 1997.

BORGES, G. A. et al. Local map building for mobile robots by fusing laser range finder and monocular video images. In: *Proceedings of the VII Simpósio Brasileiro de Automação Inteligente*. São Luís, MA, Brasil: Sociedade Brasileira de Automática, 2005. ISBN 85-85048-55-7.

BRIGGS, A. et al. Mobile robot navigation using self-similar landmarks. In: *Proceedings of the IEEE International Conference on Robotics and Automation*. San Francisco: IEEE Computer Society Press, 2000. p. 1428–1434.

CALLIOLI, C. A.; DOMINGUES, H. H.; COSTA, R. C. F. *Álgebra Linear e Aplicações*. São Paulo, SP, Brazil: Atual Editora Ltda, 1995.

CARPENTER, J.; CLIFFORD, P.; FERNHEAD, P. Improved particle filter for nonlinear problems. *Radar, Sonar and Navigation, IEE Proceedings*, v. 146, n. 1, p. 2–7, February 1999. ISSN 1350-2395.

CHANG, Y.-L.; AGGARWAL, J. K. Line correspondences from cooperating spatial and temporal grouping processes for a sequence of images. *Computer Vision and Image Understanding*, v. 67, n. 2, p. 186–201, August 1997.

CHIUSO, A. et al. Structure from motion caeually integrated over time. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, IEEE Computer Society Press, Washington, DC, EUA, v. 24, n. 4, p. 523–535, april 2002. ISSN 0162-8828.

CHUNG, W.; KIM, M. Probabilistic localization of the service robot by map matching algorithm. In: *Proceedings of International Conference on Control, Automation and Systems*. Muju, Korea: [s.n.], 2002. p. 1667–1627.

COSTA, P. L. O. *Estatística*. São Paulo, SP, Brazil: Editora Edgar Blücher Ltda, 1977.

COX, I. J.; WILFONG, G. T. (Ed.). *Autonomous robot vehicles*. New York, NY, EUA: Springer-Verlag New York, Inc., 1990. ISBN 0-387-97240-4.

DAVISON, A. J.; MURRAY, D. W. Simultaneous localization and map-building using active vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 24, n. 7, p. 865–880, 2002.

DOUCET, A. *On Sequential Simulation-Based Methods for Bayesian Filtering*. Cambridge, Inglaterra, 1998 1998.

DOUCET, A.; GORDON, N. J.; KRISHNAMURTHY, V. Particle filters for state state estimation of jump markov linear systems. *IEEE Transactions on Signal Processing*, v. 49, n. 3, p. 613–624, March 1999.

DUDEK, G.; JENKIN, M. *Computational Principles of Mobile Robotics*. Cambridge, Inglaterra: Cambridge University Press, 2000. ISBN 0 521 56876 5.

FOX, D. et al. Monte carlo localization: Efficient position estimation for mobile robots. In: *Proceedings of the 16th National Conference on Artificial Intelligence*. Orlando, Florida, EUA: ACM Press, 1999. p. 343–349.

FOX, D.; BURGARD, W.; THRUN, S. Markov localization for reliable robot navigation and people detection. In: *Selected Papers from the International Workshop on Sensor Based Intelligent Robots*. London, Inglaterra: Springer-Verlag, 1998. p. 1–20.

FOXLIN, E. Pedestrian tracking with shoe-mounted inertial sensors. *IEEE Computer Graphics and Applications*, IEEE Computer Society Press, v. 25, n. 6, p. 38–46, 2005.

FRINTROP, S.; JENSFELT, P.; CHRISTENSEN, H. I. Attentional landmark selection for visual slam. In: *Proceedings the IEEE/RSJ International Conference on Intelligent Robots and Systems*. Beijing, China: IEEE Computer Society Press, 2006. p. 2582–2587.

FUKUDA, T. et al. Navigation system based on ceiling landmark recognition for autonomous mobile robot-landmark detection based on fuzzy template-matching (ftm). In: *Proceedings of the IEEE Conference on Intelligent Robots and Systems*. Pittsburgh, PA, EUA: IEEE Computer Society Press, 1995. v. 2, p. 150–155. ISBN 0-8186-7108-4.

GONZALEZ, R. C.; WOODS, R. E. *Digital Image Processing*. Reading, Massachusetts, EUA: Addison-Wesley Publishing Company, inc., 1992.

GORDON, N. J.; SALMOND, D. J.; SMITH, A. F. M. Novel approach to nonlinear/non-gaussian bayesian state estimation. *International Journal of Radar and Signal*, IEE Proceedings, Hingham, MA, EUA, v. 140, n. 2, p. 107–113, 1993.

GRANDJEAN, P. 3-d modeling of indoor scenes by fusion of noisy range and stereo data. In: *Proceedings of IEEE International Conference of Robotics and Automation, Scottsdale*. Scottsdale, AZ: IEEE Computer Society Press, 1989. p. 681–687.

GRIMSON, W. E. L. *Object recognition by computer: the role of geometric constraints*. Cambridge, MA, EUA: MIT Press, 1990. ISBN 0-262-07130-4.

GUTMANN, J. S. Markov-kalman localization for mobile robots. In: *Proceedings of the 2002 International Conference on Pattern Recognition*. Quebec, Canada: IEEE Computer Society Press, 2002. v. 2, p. 601–604.

GUTMANN, J. S. et al. An experimental comparison of localization methods. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. Victoria, BC, Canada: IEEE Computer Society Press, 1998. p. 736–743.

HAYKIN, S. *Adaptive Filter Theory*. 3rd ed. New Jersey, EUA: Prentice Hall, 1995. ISBN 013322760X.

HAYKIN, S. *Kalman Filtering and Neural Networks*. New York, NY, EUA: Wiley Publishing, 2001.

ISARD, M.; BLAKE, A. Condensation - conditional density propagation for visual tracking. *International Journal of Computer Vision*, Kluwer Academic publishers, Hingham, MA, EUA, v. 29, n. 1, p. 5–28, 1998. ISSN 0920-5691.

JAIN, R.; KASTURI, R.; SCHUNCK, B. G. *Machine Vision*. New York, NY, EUA: McGraw-Hill, 1995. ISBN 0-07-032018-7.

JENSFELT, P. *Localization using laser scanning and minimalistic environmental models*. April 1999. Licentiate thesis, Automatic Control, Royal Institute of Technology, SE-100 44 Stockholm, Sweden. Disponível em: <citeseer.ist.psu.edu/jensfelt99localization.html>.

JENSFELT, P.; KRISTENSEN, S. Active global localization for a mobile robot using multiplehypothesis tracking. In: *Advances in Neural Information Processing Systems 2002*. Cambridge, MA, Inglaterra: MIT Press, 2001. v. 17, p. 748–760.

JULIER, S.; UHLMANN, J. A new extension of the kalman filter to nonlinear systems. In: KADAR, I. (Ed.). *Proceedings of AeroSense: The 11th International Symposium on Aerospace/Defense Sensing, Simulation and Controls*. Orlando, Florida: SPIE, 1997. v. 3068, p. 182–193.

JULIER, S. J. The scaled unscented transformation. In: *Proceedings of the IEEE American Control Conference*. Anchorage AK, EUA: IEEE Computer Society Press, 2002. p. 4555–4559.

JULIER, S. J.; UHLMANN, J. K. Reduced sigma point filters for the propagation of means and covariances through nonlinear transformations. In: *Proceedings of the IEEE American Control Conference*. Anchorage, Alaska: IEEE Computer Society Press, 2002. p. 887–892.

JULIER, S. J.; UHLMANN, J. K.; DURRANT-WHYTE, H. F. A new approach for filtering nonlinear systems. In: *Proceedings of the IEEE American Control Conference*. Seattle, Washington: IEEE Computer Society Press, 1995. v. 3, p. 1628–1632.

KAILATH, T.; SAYED, A. H.; HASSIBI, B. *Linear Estimation*. New Jersey, EUA: Prentice Hall, 2000.

KALMAN, R. E. A new approach to linear filtering and prediction problems. *Transactions of the ASME: Journal of Basic Engineering*, v. 1960, p. 35–45, March 1960.

KITAGAWA, G. Monte carlo filter and smoother for non-gaussian nonlinear state space models. *Journal of Computational and Graphical Statistics*, v. 5, p. 1–25, 1996.

KLEEMAN, L. Optimal estimation of position and heading for mobile robots using ultrasonic beacons and dead-reckoning. In: *Proceedings of the IEEE International Conference on Robotics and Automation*. Nice, France: IEEE Computer Society Press, 1992. p. 2582–2587. ISBN 0-262-61102-3.

KORTENKAMP, D.; WEYMOUTH, T. Topological mapping for mobile robots using a combination of sonar and vision sensing. In: *Proceedings of the 12th national conference on Artificial intelligence*. Seattle, Washington, United

States: American Association for Artificial Intelligence, 1994. p. 979–984. ISBN 0-262-61102-3.

KRAFT, E. A quaternion-based unscented kalman filter for orientation tracking. In: *Proceedings of the 6th International Conference of Information Fusion*. Cairns, Australia: IEEE Computer Society Press, 2003. p. 47–54.

KUC, R.; SIEGEL, M. W. Physically based simulation model for acoustic sensor robot navigation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, IEEE Computer Society Press, Washington, DC, EUA, v. 9, n. 6, p. 766–778, 1987. ISSN 0162-8828.

LANTHIER, M. A.; NUSSBAUM, D.; SHENG, A. Improving vision-based maps by using sonar and infrared data. In: *Proceedings of the 10th IASTED International Conference on Robotics and Applications*. Honolulu, Hawaii, EUA: ACTA Press, 2004.

LAVIOLA, J. J. A comparison of unscented and extended kalman filtering for estimating quaternion motion. In: *Proceedings of the IEEE American Control Conference*. Trondheim, Norway: IEEE Computer Society Press, 2003. p. 2435–2440.

LENSER, S.; VELOSO, M. Sensor resetting localization for poorly modelled mobile robots. In: *Proceedings of the IEEE International Conference on Robotics and Automation*. San Francisco, CA, EUA: IEEE Computer Society Press, 2000. v. 2, p. 1225–1232.

LEONARD, J. J.; DURRANT-WHYTE, H. F. Mobile robot localization by tracking geometric beacons. *IEEE Transactions on Robotics and Automation*, v. 7, n. 3, p. 376–382, June 1991.

LEONARD, J. J.; DURRANT-WHYTE, H. F. *Directed Sonar Sensing for Mobile Robot Navigation*. Norwell, MA, EUA: Kluwer Academic publishers, 1992.

LIN, C.; TUMMALA, R. Mobile robot navigation using artificial landmarks. *Journal of Robotic Systems*, v. 14, n. 2, p. 93–106, 1997.

LIU, J. S.; CHEN, R.; LOGVINENKO, T. *Sequential Monte Carlo in Practice*. New York: Springer-Verlag, 2000.

LU, F.; MILIOS, E. Robot pose estimation in unknown environments by matching 2d range scans. *Journal of Intelligent and Robotics Systems*, v. 18, n. 3, p. 249–275, 1997.

MA, Y. et al. *An Invitation to 3-D Vision*. New York, NY, EUA: Springer-Verlag, 2004.

MERWE, R. van der et al. The unscented particle filter. In: LEEN, T. K.; DIETTERICH, T. G.; TRESP, V. (Ed.). *Advances in Neural Information Processing Systems 13*. Cambridge, MA, Inglaterra: MIT Press, 2001. ISBN 0-262-12241-3.

MOON, I.; MIURA, J.; SHIRAI, Y. On-line selection of stable visual landmarks under uncertainty. *IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE Computer Society Press, Kyongju, Korea, AES-30, n. 2, p. 504–510, October 1999.

MOURAGNON, E. et al. Monocular vision based slam for mobile robots. In: *Proceedings of the 18th International Conference on Pattern Recognition*. Kunming, China: IEEE Computer Society Press, 2006. p. 1027–1031.

MURARKA, A.; MODAYIL, J.; KUIPERS, B. Building local safety maps for a wheelchair robot using vision and lasers. In: *Proceedings of the 3rd Canadian Conference on Computer and Robot Vision*. Austin, Texas, EUA: IEEE Computer Society Press, 2006. p. 25–25.

NAKAMURA, Y.; XU, Y. Geometrical fusion method for multisensor robotic systems. In: *Proceedings of the IEEE International Conference on Robotics and Automation*. Scottsdale, Arizona, EUA: IEEE Computer Society Press, 1989. p. 668–673.

NIETO, J. et al. Real time data association for fastslam. In: *Proceedings of the IEEE International Conference on Robotics and Automation*. Taipei, Taiwan: IEEE Computer Society Press, 2003. p. 412–418.

ORDERUD, F. Comparison of kalman filter estimation approaches for state space models with nonlinear measurements. In: *Scandinavian Conference on Simulation and Modeling*. Trondheim, Norway: [s.n.], 2005.

QIAN, G.; CHELLAPPA, R.; ZHENG, Q. Robust structure from motion estimation using inertial data. *Journal of the Optical Society of America*, v. 18, n. 12, p. 2982–2997, December 2001.

REID, D. An algorithm for tracking multiple targets. *IEEE Transactions on Automatic Control*, IEEE Computer Society Press, AC-24, p. 843–854, 1979.

RISTIC, B. et al. Performance bounds and comparison of nonlinear filters for tracking a ballistic object on re-entry. *IEE Proceedings, Radar, Sonar and Navigation*, IEEE Computer Society Press, v. 150, n. 2, p. 65–70, April 2003.

ROECKER, J. A class of near optimal jpda algorithms. *IEEE Transactions on Aerospace and Electronic Systems*, IEEE Computer Society Press, AES-30, n. 2, p. 504–510, 1994.

ROECKER, J.; PHILLIS, G. Suboptimal joint probabilistic data association. *IEEE Transactions on Aerospace and Electronic Systems*, IEEE Computer Society Press, AES-29, n. 2, p. 510–517, 1993.

RUSSELL, S. J.; NORVIG, P. *Artificial Intelligence: A Modern Approach*. New Jersey, EUA: Prentice Hall, 1995.

SAYED, A. H.; KAILATH, T. A state-space approach to adaptive rls filtering. *Signal Processing Magazine*, n. 3, p. 18–60, April 1994.

SCHMIDT, F. Applications of state space methods to navigation problems. *Advances in Control Systems*, v. 3, p. 293–340, 1966.

- SCHROETER, C. et al. Multi-sensor monte-carlo-localization combining omni-vision and sonar range sensors. In: *Proceedings of the 2nd European Conference on Mobile Robots*. Ancona, Italia: [s.n.], 2005. p. 164–169.
- ST-PIERRE, M.; GINGRAS, D. Comparison between the unscented kalman filter and the extended kalman filter for the position estimation module of an integrated navigation information system. In: *Proceedings of the IEEE Symposium on Intelligent Vehicles*. Parma, Italia: IEEE Computer Society Press, 2004. p. 831–835.
- STARK, H.; WOODS, J. W. *Probability and Random Processes with Applications to Signal Processing*. New Jersey, EUA: Prentice Hall, 2002.
- TARDÓS, J. D. et al. Robust mapping and localization in indoor environments using sonar data. *International Journal of Robotic Research*, v. 21, n. 4, p. 311–330, 2002.
- TEUKOLSKY, S. A.; VETTERLING, W. T.; FLANNERY, B. P. *Numerical Recipes in C: the Art of Scientific Computing*. Cambridge, Inglaterra: Cambridge University Press, 1992.
- THRUN, S. et al. Robust monte carlo localization for mobile robots. *Artificial Intelligence*, v. 128, n. 1-2, p. 99–141, 2001.
- WELCH, G. *SCAAT: Incremental Tracking with Incomplete Information - Thesis*. Dissertação (Mestrado) — University of North Carolina at Chapel Hill, Chapel Hill, NC, EUA, 1996.
- WELCH, G.; BISHOP, G. Scaat: incremental tracking with incomplete information. In: *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*. New York, NY, EUA: ACM Press/Addison-Wesley Publishing Co., 1997. p. 333–344. ISBN 0-89791-896-7.
- WIJK, O.; JENSFELT, P.; CHRISTENSEN, H. I. Triangulation based fusion of ultrasonic sensor data. In: *Proceedings of the IEEE International Conference on Robotics and Automation*. Leuven, Bélgica: IEEE Computer Society Press, 1998. p. 3419–3424.
- WILHELM, T.; BÖHME, H. J.; GROSS, H. M. Sensor fusion for vision and sonar based people tracking on a mobile service robot. In: *Proceedings of the International Workshop on Dynamic Perception*. Bochum, Alemanha: IOS Press infx, 2002. p. 315–320.

Apêndice I – Calibração do Sistema

A calibração é realizada em duas etapas. Na primeira são determinados os parâmetros do modelo de câmera adotado: distância focal horizontal (f_u), distância focal vertical (f_v), tamanho do plano de projeção (u_{count} e v_{count}) e o centro do plano de projeção (u_0 e v_0).

Na segunda etapa a câmera é alinhada sobre o robô de modo que o eixo principal da câmera fique paralelo à direção que o robô se deslocaria durante um movimento retilíneo.

I.1 Determinação dos parâmetros do modelo de câmera

Para determinar os parâmetros da câmera foi usado um procedimento disponibilizado no *framework* OpenCV. OpenCV é um *framework open source* mantido pela *Intel* e escrito em C++ para facilitar o desenvolvimento de algoritmos de visão computacional. OpenCV é um projeto de código livre liderado pelo Intel, e nele estão disponíveis uma série de algoritmos e programas para tratamento de imagem e vídeo. Entre os algoritmos disponíveis um tem função de determinar os parâmetros de calibração da câmera segundo o modelo de câmera CCD.

O algoritmo trabalha com um conjunto de imagens obtidas da câmera a ser calibrada observando um tabuleiro de xadrez de dimensões conhecidas, como a da figura I.1, em diferentes posições. Com base nessas imagens os parâmetros desejados são obtidos. Quanto mais imagens coletadas, maior a precisão do resultado. Neste trabalho foi determinado que é necessário coletar 40 imagens para calibrar a câmera. Com esse número de imagens não foi observado variação significativa nos parâmetros determinados pelo algoritmo em diferentes execuções.

Para coletar as imagens necessárias foi criado um programa de calibração cuja função é controlar as duas etapas de calibração. Para a primeira etapa ele coleta as imagens necessárias e as fornece ao algoritmo.

Figura I.1: Padrão usado na calibração da câmera.

I.2 Determinação da posição e orientação da câmera sobre o robô

A segunda etapa da calibração é mais crítica pois nela é necessário alterar a posição da câmera manualmente a fim de conseguir o alinhamento correto. Para tal, o robô é posicionado no centro de uma estrutura em U conforme mostra a figura I.2. O robô deve ficar posicionado de forma equidistante das paredes laterais e de modo que caso execute movimento retilíneo sua trajetória interceptaria a parede frontal perpendicularmente.

Figura I.2: Posicionamento do robô do interior da estrutura de calibração

Para auxiliar o posicionamento, o programa de calibração monitora as observações dos dois sonares frontais e dos dois sonares laterais e exibe a média das últimas 10 leituras de cada um. O robô estará posicionado corretamente quando as médias das observações dos sonares frontais ficarem iguais e quando a médias das observações dos sonares laterais também ficarem iguais. Apesar desse alinhamento inicial do robô dentro da estrutura gerar bons resultados, é necessário verificar o posicionamento correto com o auxílio de uma trena e esquadros.

Para alinhar a câmera é preciso primeiro posicionar o suporte da câmera na

região central do robô. Durante esse posicionamento é importante tentar alinhar a câmera de modo que o plano de projeção fique paralelo à parede frontal da estrutura em U. É necessário que a posição (não orientação) da câmera sobre o robô seja bem determinada nesse primeiro alinhamento, pois ela será considerada a partir de agora como a correta.

Para realizar o ajuste fino da orientação da câmera sobre o robô é usado um padrão desenhado na parede frontal da estrutura conforme mostra a figura I.3. A linha vertical do padrão está equidistante das paredes laterais, e devido aos alinhamentos realizados até agora está bem na frente do robô. O programa de alinhamento exibe para o usuário a imagem que está sendo captada pela câmera e traça uma linha vertical em vermelho bem no centro da imagem. O objetivo principal é fazer essa linha coincidir com a linha central do padrão desenhado na parede rotacionando a câmera sobre ao redor do eixo vertical.

Figura I.3: Padrão para alinhamento da câmera sobre o robô

Quando as duas linhas coincidirem (ou ficarem o mais próximo possível), o programa usa os quadrados do padrão para determinar se a câmera precisa ser rotacionada em relação aos eixos horizontais, ou seja, se é necessário erguer algum dos lados da base da câmera. O programa indica quais extremos da base da câmera precisam ser erguidos: o extremo esquerdo ou direito, e o extremo frontal ou traseiro.

Para facilitar a correção da inclinação, a câmera utilizada foi montada sobre uma base de madeira quadrada apoiada sobre quatro pés cujas alturas podem ser reguladas.

Após a orientação sobre os eixos horizontais ter sido regulada, alinha-se novamente a câmera em relação ao eixo vertical, verificando-se se é possível agora aproximar mais a linha do padrão da parede à linha de referência do programa.

Apêndice II – Estimadores Bayesianos

II.1 Introdução aos Estimadores

O processo de obtenção da localização a partir de um conjunto de leituras sensoriais pode ser visto como uma função que mapeia um espaço vetorial em outro. Se as observações realizadas pelos sensores fossem livres de erro, então bastaria determinar o modelo de observação correto para obter a localização livre de incertezas. Porém, como as observações são corrompidas por ruídos, então o que se obtém é uma estimativa da localização real.

O modelo de observação pode ser representado da seguinte maneira:

$$\mathbf{y} = h(\mathbf{x}) + v, \quad (\text{II.1})$$

onde $\mathbf{y}$ é um vetor de dimensão q que representa a observação sensorial, $\mathbf{x}$ é um vetor de dimensão r que representa o estado (a postura no caso de um estimador usado em localização), $h(.)$ é uma função, denominada de modelo de observação, que representa a ação dos sensores sobre o estado para gerar a observação, e v é um vetor de dimensão q que representa a incerteza da observação.

Seria interessante se fosse possível representar o modelo de observação da seguinte forma:

$$\mathbf{x} = h^{-1}(\mathbf{y} - v). \quad (\text{II.2})$$

Apesar desse modelo facilitar a estimação da localização, o problema é que nem sempre a função $h^{-1}(.)$ existe, ou seja, para um dado $\mathbf{y}$, pode existir mais de um $\mathbf{x}$ que satisfaça a equação II.2. Se o sensor utilizado suportar uma função do tipo $h^{-1}(\mathbf{y})$, ela pode ser usada diretamente para obter a localização. Nakamura e Xu (1989) apresentam um algoritmo de combinação sensorial que utiliza o modelo de observação dessa forma inversa.

II.2 Estimadores Bayesianos

O método mais usado na literatura de localização e mapeamento para a construção de estimadores deriva da teoria bayesiana. Nesta teoria o vetor de estado $\mathbf{x}$ e o vetor de incertezas v passam a ser tratados como vetores de variáveis aleatórias. É considerado que são realizadas observações do estado em diversos instantes gerando um seqüência discreta de observações, representadas por $\mathbf{y}(\mathbf{i})$, onde o índice i indica o instante em que a observação foi realizada. O estado no instante i é representado por $\mathbf{x}(\mathbf{i})$.

Agrupando as observações realizadas em instantes distintos e representando-as pela matriz $\mathbf{Y}(\mathbf{n}) = [\mathbf{y}(\mathbf{1}), \dots, \mathbf{y}(\mathbf{n})]$, os estimadores Bayesianos são obtidos a partir da distribuição de probabilidade *a posteriori* de $\mathbf{x}(\mathbf{n})$, dada por:

$$p_{\mathbf{x}|\mathbf{Y}}(\mathbf{x}(\mathbf{n})|\mathbf{Y}(\mathbf{n})) = \frac{p_{\mathbf{Y}|\mathbf{x}}(\mathbf{Y}(\mathbf{n})|\mathbf{x}(\mathbf{n}))p_{\mathbf{x}}(\mathbf{x}(\mathbf{n}))}{p_{\mathbf{Y}}(\mathbf{Y}(\mathbf{n}))}, \quad (\text{II.3})$$

onde $p_{\mathbf{x}}(\mathbf{x}(\mathbf{n}))$ indica a distribuição *a priori* de $\mathbf{x}(\mathbf{n})$, ou seja, essa *pdf* (função densidade de probabilidade) representa o conhecimento que se pode ter sobre o estado antes de qualquer observação.

Pode-se determinar a estimativa do estado a partir da equação II.3 através de vários métodos. Um dos métodos mais usados na literatura de localização é calculando a expectativa de $p_{\mathbf{x}|\mathbf{Y}}(\mathbf{x}(\mathbf{n})|\mathbf{Y}(\mathbf{n}))$:

$$E_x[\mathbf{x}(\mathbf{n})|\mathbf{Y}(\mathbf{n})] = \hat{\mathbf{x}}(\mathbf{n}) = \int_{\mathbf{x}} \mathbf{x}(\mathbf{n})p_{\mathbf{x}|\mathbf{Y}}(\mathbf{x}(\mathbf{n})|\mathbf{Y}(\mathbf{n}))d\mathbf{x}. \quad (\text{II.4})$$

O estimador obtido por esse modo é representado por $\hat{\mathbf{x}}(\mathbf{n})$ pois ele é o estimador que minimiza o erro quadrático médio (BERGMAN, 1999), também conhecido como estimador de mínima variância.

II.3 Estimadores Bayesianos Recursivos

Apesar da equação II.3 representar corretamente a distribuição de probabilidade do estado considerando as observações obtidas, é muito difícil obtê-la para algum problema real. Mesmo que boas aproximações sejam possíveis, o custo computacional para processá-las em geral é muito alto para que tenham uso prático em problemas de localização, onde tempo de resposta baixo em geral é um requisito importante.

Então se faz necessário realizar simplificações na *pdf* para poder usá-la. A primeira aproximação realizada é considerar que a evolução do estado no tempo pode ser modelada por um processo de Markov de ordem 1 (STARK; WOODS, 2002). Isso implica em aceitar que a probabilidade do estado no instante n , $\mathbf{x}(n)$, depende apenas do estado no instante anterior, $n - 1$:

$$p(\mathbf{x}(n)|\mathbf{x}(n-1), \dots, \mathbf{x}(0)) = p(\mathbf{x}(n)|\mathbf{x}(n-1)). \quad (\text{II.5})$$

Apesar de ser uma afirmação aparentemente muito restritiva, um grande número de sistemas são bem aproximados por processos de Markov de primeira ordem, incluindo o deslocamento do robô adotado neste trabalho.

Trabalhando apenas com processos de Markov de ordem 1 restringe-se a correlação temporal do estado apenas ao instante anterior, permitindo ignorar o estado nos demais instantes no cálculo do estado no instante atual. Caso fosse necessário considerar o estado em todos os instantes, a complexidade computacional iria crescer junto com o crescimento do índice temporal, tornando inviável o seu cálculo a partir de um certo instante.

A partir das distribuições de probabilidade envolvidas, e aplicando a equação de Chapman-Kolmogorov (STARK; WOODS, 2002), obtém-se as equações

$$p(\mathbf{x}(n)|\mathbf{Y}(n-1)) = \int p(\mathbf{x}(n)|\mathbf{x}(n-1))p(\mathbf{x}(n-1)|\mathbf{Y}(n-1))d\mathbf{x}(n-1) \quad (\text{II.6})$$

e

$$p(\mathbf{x}(n)|\mathbf{Y}(n)) = \frac{p(\mathbf{y}(n)|\mathbf{x}(n))p(\mathbf{x}(n)|\mathbf{Y}(n-1))}{p(\mathbf{y}(n)|\mathbf{Y}(n-1))}, \quad (\text{II.7})$$

que descrevem a base do Estimador Recursivo Bayesiano.

A equação II.6 é conhecida como a fase de predição do estimador: a partir da *pdf* que descreve o estado no instante anterior, $p(\mathbf{x}(n-1)|\mathbf{Y}(n-1))$, e da *pdf* que descreve a evolução do estado, $p(\mathbf{x}(n)|\mathbf{x}(n-1))$, obtém-se a *pdf* do estado no instante atual, porém sem considerar as observações no instante atual, $p(\mathbf{x}(n)|\mathbf{Y}(n-1))$. No caso da localização de robôs móveis, a equação II.6 corresponde ao uso do modelo de dinâmica.

O modelo de dinâmica é o responsável por transportar os resultados do instante anterior para o instante atual, e é representado pela função $h(\cdot)$, responsável

por determinar a transformação de $\mathbf{x}(\mathbf{n})$ em $\mathbf{x}(\mathbf{n} + 1)$ conforme:

$$\mathbf{x}(\mathbf{n}) = f(\mathbf{x}(\mathbf{n} - 1)) + u, \quad (\text{II.8})$$

onde u é um vetor da mesma dimensão que o estado e representa a incerteza do modelo de dinâmica.

A equação II.7 é conhecida como fase de correção do estimador: a partir da *pdf* obtida pela equação II.6, $p(\mathbf{x}(\mathbf{n})|\mathbf{Y}(\mathbf{n} - 1))$, e da *pdf* que indica a probabilidade das observações dado o estado, $p(\mathbf{y}(\mathbf{n})|\mathbf{x}(\mathbf{n}))$, obtém-se a *pdf* que descreve o estado no instante atual. A *pdf* $p(\mathbf{y}(\mathbf{n})|\mathbf{Y}(\mathbf{n} - 1))$ é denominada evidência e, como não depende do estado, não influencia na determinação da estimativa do estado. Essa equação corresponde à aplicação do modelo de observação.

Esse par de equações determina a arquitetura de um Estimador Bayesiano Recursivo. O fato dele utilizar o resultado obtido no instante anterior no cálculo da estimativa do instante atual permite a construção de estimadores mais eficientes, que precisam integrar a cada iteração uma quantidade bem menor de dados.

Combinando as equações dos modelos de dinâmica e observação obtém-se o sistema que caracteriza os Estimadores Bayesianos Recursivos:

$$\begin{aligned} \mathbf{x}(\mathbf{n}) &= f(\mathbf{x}(\mathbf{n} - 1)) + u \\ \mathbf{y}(\mathbf{n}) &= h(\mathbf{x}(\mathbf{n})) + v. \end{aligned} \quad (\text{II.9})$$

II.4 Estimadores Lineares Gaussianos

As equações II.6 e II.7 admitem *pdfs* que podem ser bastante complexas e difíceis de modelar. Para apresentar estimadores que possam ser usados na prática, é necessário impor restrições adicionais.

O conjunto de restrições considerado nessa seção limita tanto as *pdfs* como os modelos envolvidos. É considerado que os sistemas descritos pelos modelos $f(\cdot)$ e $h(\cdot)$ das equações II.9 são lineares, e que $\mathbf{x}(\mathbf{n})$ e $\mathbf{y}(\mathbf{n})$ são variáveis aleatórias que seguem distribuições gaussianas de média zero, independentes entre si e entre diferentes instantes (KAILATH; SAYED; HASSIBI, 2000).

Considerando essas restrições, o sistema II.9 pode ser escrito da seguinte

forma:

$$\begin{aligned}\mathbf{x}(\mathbf{n}) &= \mathbf{F}(\mathbf{n}).\mathbf{x}(\mathbf{n} - 1) + u(n); \quad u(n) \sim N(0, \mathbf{Q}(\mathbf{n})) \\ \mathbf{y}(\mathbf{n}) &= \mathbf{H}(\mathbf{n}).\mathbf{x}(\mathbf{n}) + v(n); \quad v(n) \sim N(0, \mathbf{R}(\mathbf{n})).\end{aligned}\tag{II.10}$$

As matrizes $\mathbf{F}(\cdot)$ e $\mathbf{H}(\cdot)$ são as versões lineares dos modelos de dinâmica e observação, respectivamente, sendo que $v(n)$ e $u(n)$ representam a perturbação gaussiana e obedecem à:

$$\begin{aligned}E[v(i).u(j)] &= 0 \quad \forall i, j \\ E[u(i).u(j)] &= 0, \quad \forall i \neq j \\ E[v(i).v(j)] &= 0, \quad \forall i \neq j,\end{aligned}\tag{II.11}$$

e $\mathbf{R}(\cdot)$ e $\mathbf{Q}(\cdot)$ são suas matrizes de covariância, respectivamente. É importante notar que apesar de serem lineares, $\mathbf{F}(\cdot)$ e $\mathbf{H}(\cdot)$ ainda podem ser variáveis no tempo.

Assumindo que todas variáveis que atuam sobre o sistema seguem *pdfs* gaussianas, então o vetor aleatório do estado pode ser totalmente descrito pelos seus dois primeiros momentos, a média e a covariância.

Kalman (1960) apresentou um estimador para o problema descrito, conhecido como Filtro de Kalman. Esse é o estimador ótimo para as condições apresentadas acima: sistema linear com incertezas descritas por ruído branco. Kailath, Sayed e Hassibi (2000) e Haykin (1995) apresentam duas deduções distintas das equações do Filtro de Kalman. O Filtro de Kalman mantém, entre duas iterações, a estimativa e a covariância do estado.

O Filtro de Kalman opera do seguinte modo: primeiro, com base na estimativa do estado na iteração anterior, $\bar{\mathbf{x}}(\mathbf{n} - 1)$, é predito o estado atual, $\bar{\mathbf{x}}(\mathbf{n}|\mathbf{n} - 1)$, usando o modelo de dinâmica $\mathbf{F}(\mathbf{n})$:

$$\bar{\mathbf{x}}(\mathbf{n}|\mathbf{n} - 1) = \mathbf{F}(\mathbf{n}).\bar{\mathbf{x}}(\mathbf{n} - 1).$$

A seguir, o estado predito é corrigido pelas observações dos sensores, $\mathbf{y}(\mathbf{n})$. Para isso o Filtro de Kalman usa o modelo de observação para obter qual seria a observação dos sensores caso o estado real fosse dado por $\bar{\mathbf{x}}(\mathbf{n}|\mathbf{n} - 1)$. Essa observação é denominada de observação predita e é dada por:

$$\bar{\mathbf{y}}(\mathbf{n}) = \mathbf{H}(\mathbf{n}).\bar{\mathbf{x}}(\mathbf{n}|\mathbf{n} - 1).$$

Usando a observação dos sensores e a observação predita, obtém-se a inovação, $\mathbf{e}(\mathbf{n})$, que é a diferença de uma pela outra:

$$\mathbf{e}(\mathbf{n}) = \mathbf{y}(\mathbf{n}) - \bar{\mathbf{y}}(\mathbf{n}).$$

A estimativa do estado, $\bar{\mathbf{x}}(\mathbf{n})$, é obtida somando-se ao estado predito a inovação multiplicada por um fator de ganho, $\mathbf{K}(\mathbf{n})$, conhecido por Ganho de Kalman:

$$\bar{\mathbf{x}}(\mathbf{n}) = \bar{\mathbf{x}}(\mathbf{n}|\mathbf{n} - 1) + \mathbf{K}(\mathbf{n}).\mathbf{e}(\mathbf{n}).$$

O Ganho de Kalman indica a importância dada à observação sobre a predição (estimativa obtida com o modelo de dinâmica). Quanto mais precisa for a observação em relação à predição, maior será o ganho. O Ganho de Kalman também tem por função transformar a inovação, que está no espaço vetorial da observação, para que possa ser somada ao estado.

O Ganho de Kalman é dado por:

$$\mathbf{K}(\mathbf{n}) = \mathbf{P}_x(\mathbf{n}|\mathbf{n} - 1).\mathbf{H}^T(\mathbf{n})(\mathbf{H}(\mathbf{n}).\mathbf{P}_x(\mathbf{n}|\mathbf{n} - 1).\mathbf{H}^T(\mathbf{n}) + \mathbf{R}(\mathbf{n}))^{-1},$$

onde $\mathbf{P}_x(\mathbf{n}|\mathbf{n} - 1)$ representa a covariância do estado predito e será obtida mais à frente. Pode-se ver que o Ganho de Kalman usa $\mathbf{P}_x(\mathbf{n}|\mathbf{n} - 1)$ e $\mathbf{R}(\mathbf{n})$ para determinar o quanto da inovação será incorporada no estado. Considerando que $\mathbf{P}_x(\mathbf{n}|\mathbf{n} - 1)$ e $\mathbf{R}(\mathbf{n})$ sejam escalares e que $\mathbf{H}(\mathbf{n})$ seja 1, tem-se:

$$\mathbf{K}(\mathbf{n}) = \frac{\mathbf{P}_x(\mathbf{n}|\mathbf{n} - 1)}{\mathbf{P}_x(\mathbf{n}|\mathbf{n} - 1) + \mathbf{R}(\mathbf{n})}.$$

Quanto maior for $\mathbf{P}_x(\mathbf{n}|\mathbf{n} - 1)$ em relação a $\mathbf{R}(\mathbf{n})$ (maior a incerteza sobre o estado predito em relação às observações), tem-se $\mathbf{K}(\mathbf{n})$ mais próximo da unidade, aumentando a participação da observação sobre a estimativa do estado. No caso contrário, $\mathbf{K}(\mathbf{n})$ se aproxima de zero, aumentando a contribuição relativa da predição na estimativa corrigida do estado.

Obtém-se $\mathbf{P}_x(\mathbf{n}|\mathbf{n} - 1)$ a partir da covariância da estimativa do estado na iteração anterior, $\mathbf{P}_x(\mathbf{n} - 1)$, do modo similar ao que é obtido o estado predito:

$$\mathbf{P}_x(\mathbf{n}|\mathbf{n} - 1) = \mathbf{F}(\mathbf{n}).\mathbf{P}_x(\mathbf{n} - 1).\mathbf{F}^T(\mathbf{n}) + \mathbf{Q}(\mathbf{n}).$$

Já a covariância do estado no instante atual, $\mathbf{P}_x(\mathbf{n})$ é obtida pela seguinte equação:

$$\mathbf{P}_x(\mathbf{n}) = (\mathbf{I} - \mathbf{K}(\mathbf{n})).\mathbf{P}_x(\mathbf{n}|\mathbf{n} - 1).$$

Haykin (1995) demonstra como se obtém as equações acima de forma detalhada.

O resumo das equações necessárias para o Filtro de Kalman se encontra a seguir:

$$\begin{aligned}
 \bar{\mathbf{x}}(\mathbf{n}|\mathbf{n}-1) &= \mathbf{F}(\mathbf{n}).\bar{\mathbf{x}}(\mathbf{n}-1) \\
 \bar{\mathbf{x}}(\mathbf{n}) &= \bar{\mathbf{x}}(\mathbf{n}|\mathbf{n}-1) + \mathbf{K}(\mathbf{n}).\mathbf{e}(\mathbf{n}) \\
 \mathbf{e}(\mathbf{n}) &= \mathbf{y}(\mathbf{n}) - \bar{\mathbf{y}}(\mathbf{n}) = \mathbf{y}(\mathbf{n}) - \mathbf{H}(\mathbf{n}).\bar{\mathbf{x}}(\mathbf{n}|\mathbf{n}-1) \\
 \mathbf{K}(\mathbf{n}) &= \mathbf{P}_x(\mathbf{n}|\mathbf{n}-1).\mathbf{H}^T(\mathbf{n})(\mathbf{H}(\mathbf{n}).\mathbf{P}_x(\mathbf{n}|\mathbf{n}-1).\mathbf{H}^T(\mathbf{n}) + \mathbf{R}(\mathbf{n}))^{-1} \\
 \mathbf{P}_x(\mathbf{n}|\mathbf{n}-1) &= \mathbf{F}(\mathbf{n}).\mathbf{P}_x(\mathbf{n}-1).\mathbf{F}^T(\mathbf{n}) + \mathbf{Q}(\mathbf{n}) \\
 \mathbf{P}_x(\mathbf{n}) &= (\mathbf{I} - \mathbf{K}(\mathbf{n})).\mathbf{P}_x(\mathbf{n}|\mathbf{n}-1),
 \end{aligned} \tag{II.12}$$

II.5 Estimadores Não Lineares Gaussianos

O Filtro de Kalman pode ser aplicado apenas a sistemas lineares. Porém boa parte dos sistemas robóticos e modelos sensoriais não podem ser aproximados por esse tipo de sistema. Para contornar essa limitação, foram desenvolvidas variações do Filtro de Kalman para lidar com sistemas não-lineares. Entre as diversas variações, duas foram selecionadas para serem avaliadas neste trabalho: o Filtro de Kalman Estendido (EKF) e o Filtro de Kalman Unscented (UKF).

Partindo do sistema II.10, pode-se obter o sistema considerado nos estimadores não lineares gaussianos trocando as matrizes $\mathbf{F}(\cdot)$ e $\mathbf{H}(\cdot)$ que representam as versões lineares dos modelos de dinâmica e observação, respectivamente, pelas funções não-lineares $f(\cdot)$ e $h(\cdot)$:

$$\mathbf{x}(\mathbf{n}) = f(\mathbf{x}(\mathbf{n}-1)) + u(n); \quad u(n) \sim N(0, \mathbf{Q}(\mathbf{n})) \tag{II.13}$$

$$\mathbf{y}(\mathbf{n}) = h(\mathbf{x}(\mathbf{n})) + v(n); \quad v(n) \sim N(0, \mathbf{R}(\mathbf{n})), \tag{II.14}$$

sendo que $v(n)$ e $u(n)$ representam a perturbação gaussiana e obedecem as regras dadas pelas equações II.11.

Apesar de $f(\cdot)$ e $h(\cdot)$ não receberem um índice temporal próprio, elas podem variar com o tempo.

Tanto o EKF quanto o UKF têm por objetivo adaptar o Filtro de Kalman a sistemas não-lineares, cada um adotando uma abordagem distinta.

II.5.1 Filtro de Kalman Estendido

O Filtro de Kalman Estendido (HAYKIN, 1995) utiliza o conceito de *linearização local* para obter representações lineares a partir dos modelos não-lineares. Algumas linearizações são apresentadas por Kailath, Sayed e Hassibi (2000), porém Schmidt (1966) descreve primeiramente a mais conhecida, que é a adotada pelo EKF. O EKF aproxima as funções $f(\cdot)$ e $h(\cdot)$ através de uma expansão de Taylor de primeira ordem.

A expansão em série de Taylor da função $f(\cdot)$ é feita sobre o valor $\bar{\mathbf{x}}(\mathbf{n} - 1)$ e é dada por:

$$f(\mathbf{x}(\mathbf{n} - 1)) = f(\bar{\mathbf{x}}(\mathbf{n} - 1)) + \mathbf{F}_{\mathbf{n}}(\mathbf{x}(\mathbf{n} - 1) - \bar{\mathbf{x}}(\mathbf{n} - 1)), \quad (\text{II.15})$$

onde $\mathbf{F}_{\mathbf{n}}$ é dada pela derivada $f(\cdot)$ por $\mathbf{x}(\mathbf{n} - 1)$, sobre o valor $\bar{\mathbf{x}}(\mathbf{n} - 1)$:

$$\mathbf{F}(\mathbf{n} - 1) = \frac{\partial f(\mathbf{x}(\mathbf{n} - 1))}{\partial \mathbf{x}(\mathbf{n} - 1)} \mid \mathbf{x}(\mathbf{n} - 1) = \bar{\mathbf{x}}(\mathbf{n} - 1). \quad (\text{II.16})$$

$\mathbf{F}_{\mathbf{n}}$ não foi denominada como $\mathbf{F}_{\mathbf{n}-1}$ pois é usada para transformar o estado no instante $n - 1$ no estado no instante n , assim como a matriz $\mathbf{F}(\mathbf{n})$ no caso do sistema linear II.10.

A expansão em série de Taylor da função $h(\cdot)$ é feita sobre o valor $\bar{\mathbf{x}}(\mathbf{n}|\mathbf{n} - 1)$ e é dada por:

$$h(\mathbf{x}(\mathbf{n}|\mathbf{n} - 1)) = h(\bar{\mathbf{x}}(\mathbf{n}|\mathbf{n} - 1)) + \mathbf{H}_{\mathbf{n}}(\mathbf{x}(\mathbf{n}|\mathbf{n} - 1) - \bar{\mathbf{x}}(\mathbf{n}|\mathbf{n} - 1)), (\text{II.17})$$

onde $\mathbf{H}_{\mathbf{n}}$ é dada pela derivada $h(\cdot)$ por $\mathbf{x}(\mathbf{n}|\mathbf{n} - 1)$, sobre o valor $\bar{\mathbf{x}}(\mathbf{n}|\mathbf{n} - 1)$:

$$\mathbf{H}_{\mathbf{n}} = \frac{\partial h(\mathbf{x}(\mathbf{n}|\mathbf{n} - 1))}{\partial \mathbf{x}(\mathbf{n}|\mathbf{n} - 1)} \mid \mathbf{x}(\mathbf{n}|\mathbf{n} - 1) = \bar{\mathbf{x}}(\mathbf{n}|\mathbf{n} - 1).$$

O EKF utiliza as matrizes $\mathbf{F}_{\mathbf{n}}$ e $\mathbf{H}_{\mathbf{n}}$ no lugar das matrizes $\mathbf{F}(\mathbf{n})$ e $\mathbf{H}(\mathbf{n})$, respectivamente, nas equações do Filtro de Kalman no cálculo de $\mathbf{K}(\mathbf{n})$ e de $\mathbf{P}_{\mathbf{x}}(\mathbf{n})$. Para o cálculo de $\bar{\mathbf{x}}(\mathbf{n}|\mathbf{n} - 1)$ e de $\mathbf{e}(\mathbf{n})$ são usadas as próprias funções não lineares $f(\cdot)$ e $h(\cdot)$, respectivamente. Haykin (1995) demonstra a correteza dessa aproximação.

O resumo das equações do Filtro de Kalman EKF para sistemas lineares dadas

por II.12 são reescritas a seguir realizando as alterações citadas acima:

$$\begin{aligned}
 \bar{\mathbf{x}}(\mathbf{n}|\mathbf{n}-1) &= f(\bar{\mathbf{x}}(\mathbf{n}-1)) \\
 \bar{\mathbf{x}}(\mathbf{n}) &= \bar{\mathbf{x}}(\mathbf{n}|\mathbf{n}-1) + \mathbf{K}(\mathbf{n}).\mathbf{e}(\mathbf{n}) \\
 \mathbf{e}(\mathbf{n}) &= \mathbf{y}(\mathbf{n}) - \bar{\mathbf{y}}(\mathbf{n}) = \mathbf{y}(\mathbf{n}) - h(\bar{\mathbf{x}}(\mathbf{n}|\mathbf{n}-1)) \\
 \mathbf{K}(\mathbf{n}) &= \mathbf{P}_{\mathbf{x}}(\mathbf{n}|\mathbf{n}-1).\mathbf{H}_{\mathbf{n}}^T(\mathbf{H}_{\mathbf{n}}.\mathbf{P}_{\mathbf{x}}(\mathbf{n}|\mathbf{n}-1).\mathbf{H}_{\mathbf{n}}^T + \mathbf{R}(\mathbf{n}))^{-1} \\
 \mathbf{P}_{\mathbf{x}}(\mathbf{n}|\mathbf{n}-1) &= \mathbf{F}_{\mathbf{n}}.\mathbf{P}_{\mathbf{x}}(\mathbf{n}-1).\mathbf{F}_{\mathbf{n}}^T + \mathbf{Q}(\mathbf{n}) \\
 \mathbf{P}_{\mathbf{x}}(\mathbf{n}) &= (\mathbf{I} - \mathbf{K}(\mathbf{n})).\mathbf{P}_{\mathbf{x}}(\mathbf{n}|\mathbf{n}-1).
 \end{aligned}
 \tag{II.18}$$

Estabilização do Filtro de Kalman Estendido

O cálculo de $\mathbf{P}_{\mathbf{x}}(\mathbf{n})$, dado por:

$$\mathbf{P}_{\mathbf{x}}(\mathbf{n}) = (\mathbf{I} - \mathbf{K}(\mathbf{n})).\mathbf{P}_{\mathbf{x}}(\mathbf{n}|\mathbf{n}-1), \tag{II.19}$$

pode causar instabilidade numérica no sistema da forma em que está escrito, devido à impossibilidade de se representar os números com precisão infinita. Sendo a matriz $\mathbf{P}_{\mathbf{x}}(\mathbf{n})$ uma matriz de covariância, ela deve ser positivo-definida, porém ela é obtida através de uma diferença de matrizes $(\mathbf{I} - \mathbf{K}(\mathbf{n}))$ que pode resultar numa matriz negativo-definida devido a precisão limitada. Esse problema é conhecido como o Fenômeno da Divergência (HAYKIN, 2001).

Uma solução simples para contornar esse problema é reescrever a equação do cálculo de $\mathbf{P}_{\mathbf{x}}(\mathbf{n})$ de modo que ela fique da forma $\mathbf{B}.\mathbf{P}.\mathbf{B}^T + \mathbf{C}$, onde $\mathbf{P}$ e $\mathbf{C}$ são matrizes positivo-definidas, e $\mathbf{B}.\mathbf{P}.\mathbf{B}^T$ é positivo-definida, ou nula caso as linhas de $\mathbf{B}$ sejam linearmente dependentes.

Haykin (2001) mostra que a equação II.19 pode ser reescrita da seguinte forma:

$$\mathbf{P}_{\mathbf{x}}(\mathbf{n}) = (\mathbf{I} - \mathbf{K}(\mathbf{n})).\mathbf{P}_{\mathbf{x}}(\mathbf{n}|\mathbf{n}-1).(\mathbf{I} - \mathbf{K}(\mathbf{n}))^T + \mathbf{K}(\mathbf{n})\mathbf{R}(\mathbf{n})\mathbf{K}^T(\mathbf{n})$$

Considerando essa modificação, as equações do EKF ficam, então, da seguinte

forma:

$$\begin{aligned}
\bar{\mathbf{x}}(\mathbf{n}|\mathbf{n}-1) &= f(\bar{\mathbf{x}}(\mathbf{n}-1)) \\
\bar{\mathbf{x}}(\mathbf{n}) &= \bar{\mathbf{x}}(\mathbf{n}|\mathbf{n}-1) + \mathbf{K}(\mathbf{n}).\mathbf{e}(\mathbf{n}) \\
\mathbf{e}(\mathbf{n}) &= \mathbf{y}(\mathbf{n}) - \bar{\mathbf{y}}(\mathbf{n}) = \mathbf{y}(\mathbf{n}) - h(\bar{\mathbf{x}}(\mathbf{n}|\mathbf{n}-1)) \\
\mathbf{K}(\mathbf{n}) &= \mathbf{P}_{\mathbf{x}}(\mathbf{n}|\mathbf{n}-1).\mathbf{H}^T(\mathbf{n})(\mathbf{H}(\mathbf{n}).\mathbf{P}_{\mathbf{x}}(\mathbf{n}|\mathbf{n}-1).\mathbf{H}^T(\mathbf{n}) + \mathbf{R}(\mathbf{n}))^{-1} \\
\mathbf{P}_{\mathbf{x}}(\mathbf{n}|\mathbf{n}-1) &= \mathbf{F}(\mathbf{n}).\mathbf{P}_{\mathbf{x}}(\mathbf{n}-1).\mathbf{F}^T(\mathbf{n}) + \mathbf{Q}(\mathbf{n}) \\
\mathbf{P}_{\mathbf{x}}(\mathbf{n}) &= (\mathbf{I} - \mathbf{K}(\mathbf{n})).\mathbf{P}_{\mathbf{x}}(\mathbf{n}|\mathbf{n}-1).(\mathbf{I} - \mathbf{K}(\mathbf{n}))^T + \mathbf{K}(\mathbf{n})\mathbf{R}(\mathbf{n})\mathbf{K}^T(\mathbf{n})
\end{aligned} \tag{II.20}$$

Outra solução para o problema seria usar uma implementação conhecida como Filtro de Kalman Estendido da Raiz Quadrada, ou SR-EKF (SAYED; KAILATH, 1994). Essa implementação propaga o fator de Cholesky¹ da matriz de covariância do estado em vez da própria $\mathbf{P}$. Solução similar é adotada neste trabalho para eliminar o mesmo problema no próximo estimador a ser descrito, o Filtro de Kalman Unscented.

II.5.2 Filtro de Kalman Unscented

O funcionamento do EKF depende do quanto a linearização realizada se aproxima da função não-linear, pela região em que a linearização é considerada. Por exemplo, suponha o modelo de dinâmica de um robô que descreve uma trajetória circular. A linearização sobre um ponto da trajetória seria uma reta tangente ao ponto tomado como base para a linearização. A fidelidade da linearização com a trajetória real depende da distância percorrida entre iterações e do raio da trajetória. Quanto maior a distância ou menor o raio, pior será a linearização obtida.

Julier e Uhlmann (1997) apontaram esse problema com o EKF usando como exemplo a transformação de conversão de coordenadas polares para coordenadas cartesianas. Eles apontam que vários modelos usados na prática não são bem representados por linearizações de primeira (usada pelo EKF) ou segunda ordem. Nesse mesmo trabalho eles sugerem uma abordagem alternativa, que utiliza a Transformação Unscented (UT) (JULIER; UHLMANN; DURRANT-WHYTE, 1995). O estimador resultante foi batizado de Filtro de Kalman Unscented (UKF).

A transformação UT representa a *pdf* do vetor aleatório a ser transformado

¹O fator de Cholesky $\mathbf{A}$ da matriz $\mathbf{P}$ é uma matriz triangular inferior tal que $\mathbf{A}.\mathbf{A}^T = \mathbf{P}$. O fator de Cholesky é obtido fatorando $\mathbf{P}$ através da decomposição de Cholesky.

por um conjunto de valores, denominado de *pontos sigma*. Esses *pontos sigma* são calculados de forma determinística usando a estimativa ($\bar{\mathbf{x}}(\mathbf{n})$) e a covariância ($\mathbf{P}_{\mathbf{x}}(\mathbf{n})$) da *pdf* como parâmetros. Cada *ponto sigma* do conjunto é aplicado à função não linear, e o conjunto resultante representa a *pdf* transformada do vetor aleatório original. A partir dessa *pdf* são calculadas a média e covariância, obtendo assim as características que descrevem o vetor aleatório transformado.

Considerando que o estado tenha dimensão N , o algoritmo básico de UT utiliza $2N + 1$ *pontos sigma* obtidos conforme descrito pelas seguintes equações:

$$\begin{aligned}\mathcal{X}^{(0)}(\mathbf{n}) &= \bar{\mathbf{x}}(\mathbf{n}) \\ \mathcal{X}^{(i)}(\mathbf{n}) &= \bar{\mathbf{x}}(\mathbf{n}) + (\sqrt{(N + \delta) \cdot \mathbf{P}_{\mathbf{x}}(\mathbf{n})})^{(i)}, \quad i = \{1, \dots, N\} \\ \mathcal{X}^{(i+N)}(\mathbf{n}) &= \bar{\mathbf{x}}(\mathbf{n}) - (\sqrt{(N + \delta) \cdot \mathbf{P}_{\mathbf{x}}(\mathbf{n})})^{(i)}, \quad i = \{1, \dots, N\},\end{aligned}\quad (\text{II.21})$$

onde $\mathcal{X}(\mathbf{n})$ é uma matriz cujas colunas são os *pontos sigma*, $\mathbf{P}_{\mathbf{x}}$ é a matriz de covariância do estado, e $(\sqrt{(N + \delta) \cdot \mathbf{P}_{\mathbf{x}}})^{(i)}$ indica a coluna i da matriz $\sqrt{(N + \delta) \cdot \mathbf{P}_{\mathbf{x}}}$. A matriz $\sqrt{(N + \delta) \cdot \mathbf{P}_{\mathbf{x}}}$ é o fator de Cholesky (TEUKOLSKY; VETTERLING; FLANNERY, 1992) da matriz $(N + \delta) \cdot \mathbf{P}_{\mathbf{x}}$, de tal modo que $\sqrt{(N + \delta) \cdot \mathbf{P}_{\mathbf{x}}} \cdot \sqrt{(N + \delta) \cdot \mathbf{P}_{\mathbf{x}}}^T = (N + \delta) \cdot \mathbf{P}_{\mathbf{x}}$; δ é uma constante de escala, definida por $\delta = \alpha^2(N + \kappa) - N$, sendo que α determina o espalhamento dos *pontos sigma* sobre $\bar{\mathbf{x}}$ (JULIER, 2002), e κ é um parâmetro secundário de escala (JULIER; UHLMANN; DURRANT-WHYTE, 1995). Julier e Uhlmann (2002) descrevem uma variação na obtenção dos *pontos sigma* onde o estado é representado por apenas $N + 1$ *pontos sigma*.

A transformação UT funciona da seguinte forma: sendo $\mathcal{Z}$ a matriz que contém os *pontos sigma*, aplica-se a função não-linear, $g(\cdot)$, por exemplo, a cada coluna de $\mathcal{Z}$, e com base nos *pontos sigma* resultantes, $\mathcal{T}$, são determinadas a média e a covariância do estado transformado, conforme as seguintes equações:

$$\begin{aligned}\mathcal{T}^{(i)} &= g(\mathcal{Z}^{(i)}) \\ \bar{\mathbf{t}} &= \sum_{i=0}^{2N} W_m^{(i)} \mathcal{T}^{(i)} \\ \mathbf{P}_{\mathbf{t}} &= \sum_{i=0}^{2N} W_c^{(i)} (\mathcal{T}^{(i)} - \bar{\mathbf{t}})(\mathcal{T}^{(i)} - \bar{\mathbf{t}})^T,\end{aligned}\quad (\text{II.22})$$

onde $W_m^{(i)}$ e $W_c^{(i)}$ são pesos relativos a cada *pontos sigma* usados para determinar

a média e a covariância, respectivamente. Os pesos são dados por:

$$\begin{aligned} W_m^{(0)} &= \frac{\delta}{N + \delta} \\ W_c^{(0)} &= \frac{\delta}{N + \delta} + (1 - \alpha^2 + \beta) \\ W_c^{(i)} &= W_m^{(i)} = \frac{1}{2(N + \delta)}, \quad i = \{1, \dots, 2N\}, \end{aligned} \quad (\text{II.23})$$

onde β é um parâmetro usado para incorporar conhecimento prévio da distribuição (JULIER, 2002). Para distribuições gaussianas o valor ótimo é 2.

Julier e Uhlmann (1997) argumentam que, apesar da UT considerar vetores aleatórios que seguem distribuições gaussianas, a transformação consegue determinar com precisão os dois primeiros momentos do vetor, seja qual for a distribuição que ele obedece.

O UKF utiliza a UT para determinar a média e covariância dos vetores transformados pelos modelos de dinâmica e observação. Antes de mostrar as equações do UKF, o Filtro de Kalman deve ser escrito de uma forma alternativa, onde as covariâncias e covariâncias cruzadas entre o estado e a observação são usadas para determinar o Ganho de Kalman ($\mathbf{K}(\mathbf{n})$):

$$\begin{aligned} \bar{\mathbf{x}}(\mathbf{n}) &= \bar{\mathbf{x}}(\mathbf{n}|\mathbf{n}-1) + \mathbf{K}(\mathbf{n}) \cdot \mathbf{e}(\mathbf{n}) \\ \mathbf{e}(\mathbf{n}) &= \mathbf{y}(\mathbf{n}) - \bar{\mathbf{y}}(\mathbf{n}|\mathbf{n}-1) \\ \mathbf{K}(\mathbf{n}) &= \mathbf{P}_{\mathbf{x}\mathbf{e}}(\mathbf{n}|\mathbf{n}-1) \cdot \mathbf{P}_{\mathbf{e}}(\mathbf{n}|\mathbf{n}-1)^{-1} \\ \mathbf{P}_{\mathbf{x}}(\mathbf{n}) &= \mathbf{P}_{\mathbf{x}}(\mathbf{n}|\mathbf{n}-1) - \mathbf{K}(\mathbf{n}) \mathbf{P}_{\mathbf{e}}(\mathbf{n}|\mathbf{n}-1) \mathbf{K}(\mathbf{n})^T, \end{aligned} \quad (\text{II.24})$$

onde $\mathbf{P}_{\mathbf{e}}(\mathbf{n}|\mathbf{n}-1)$ é a matriz de covariância da inovação $\mathbf{e}(\mathbf{n})$, e $\mathbf{P}_{\mathbf{x}\mathbf{e}}(\mathbf{n}|\mathbf{n}-1)$ é a matriz de covariância cruzada entre o estado predito e a inovação.

A UT é usada então para determinar as transformações das funções $f(\cdot)$ e $h(\cdot)$ que resultam em $\bar{\mathbf{x}}(\mathbf{n}|\mathbf{n}-1)$ e $\bar{\mathbf{y}}(\mathbf{n}|\mathbf{n}-1)$, respectivamente, e para calcular as matrizes de covariância $\mathbf{P}_{\mathbf{x}}(\mathbf{n}|\mathbf{n}-1)$, $\mathbf{P}_{\mathbf{x}\mathbf{e}}(\mathbf{n}|\mathbf{n}-1)$ e $\mathbf{P}_{\mathbf{e}}(\mathbf{n}|\mathbf{n}-1)$. Primeiro é obtido o estado predito $\bar{\mathbf{x}}(\mathbf{n}|\mathbf{n}-1)$ a partir dos *pontos sigma* $\mathcal{X}(\mathbf{n})$:

$$\begin{aligned} \mathcal{X}^{(i)}(\mathbf{n}|\mathbf{n}-1) &= f(\mathcal{X}^{(i)}(\mathbf{n}-1)) \\ \bar{\mathbf{x}}(\mathbf{n}|\mathbf{n}-1) &= \sum_{i=0}^{2n} W_m^{(i)} \mathcal{X}^{(i)}(\mathbf{n}|\mathbf{n}-1). \end{aligned}$$

Em seguida, obtém-se a observação predita $\bar{\mathbf{y}}(\mathbf{n}|\mathbf{n}-1)$:

$$\begin{aligned}\mathcal{Y}^{(i)}(\mathbf{n}|\mathbf{n}-1) &= h(\mathcal{X}^{(i)}(\mathbf{n}|\mathbf{n}-1)) \\ \bar{\mathbf{y}}(\mathbf{n}|\mathbf{n}-1) &= \sum_{i=0}^{2N} W_m^{(i)} \mathcal{Y}^{(i)}(\mathbf{n}|\mathbf{n}-1)\end{aligned}$$

Usando os *pontos sigma* transformados por $f(\cdot)$, dados em $\mathcal{X}(\mathbf{n})$, obtém-se $\mathbf{P}_{\mathbf{x}}(\mathbf{n}|\mathbf{n}-1)$:

$$\begin{aligned}\mathbf{P}_{\mathbf{x}}(\mathbf{n}|\mathbf{n}-1) &= \mathbf{Q} + \\ &\sum_{i=0}^{2N} W_c^{(i)} (\mathcal{X}^{(i)}(\mathbf{n}|\mathbf{n}-1) - \bar{\mathbf{x}}(\mathbf{n}|\mathbf{n}-1)) (\mathcal{X}^{(i)}(\mathbf{n}|\mathbf{n}-1) - \bar{\mathbf{x}}(\mathbf{n}|\mathbf{n}-1))^T\end{aligned}$$

Finalmente obtém-se as matrizes de covariância $\mathbf{P}_{\mathbf{e}}(\mathbf{n}|\mathbf{n}-1)$ e $\mathbf{P}_{\mathbf{x}\mathbf{e}}(\mathbf{n}|\mathbf{n}-1)$ usando os *pontos sigma* transformados por $f(\cdot)$ e $g(\cdot)$:

$$\begin{aligned}\mathbf{P}_{\mathbf{e}}(\mathbf{n}|\mathbf{n}-1) &= \mathbf{R} + \\ &\sum_{i=0}^{2N} W_c^{(i)} (\mathcal{Y}^{(i)}(\mathbf{n}|\mathbf{n}-1) - \bar{\mathbf{y}}(\mathbf{n}|\mathbf{n}-1)) (\mathcal{Y}^{(i)}(\mathbf{n}|\mathbf{n}-1) - \bar{\mathbf{y}}(\mathbf{n}|\mathbf{n}-1))^T \\ \mathbf{P}_{\mathbf{x}\mathbf{e}}(\mathbf{n}|\mathbf{n}-1) &= \sum_{i=0}^{2N} W_c^{(i)} (\mathcal{X}^{(i)}(\mathbf{n}|\mathbf{n}-1) - \bar{\mathbf{x}}(\mathbf{n}|\mathbf{n}-1)) (\mathcal{Y}^{(i)}(\mathbf{n}|\mathbf{n}-1) - \bar{\mathbf{y}}(\mathbf{n}|\mathbf{n}-1))^T\end{aligned}$$

O cálculo de $\mathbf{P}_{\mathbf{e}}(\mathbf{n}|\mathbf{n}-1)$ é a soma das covariâncias da observação com a predição. Isso vem da suposição que a observação num dado instante é independente das observações nos demais instantes. A incerteza da observação no instante atual também não influi no cálculo de $\mathbf{P}_{\mathbf{x}\mathbf{e}}(\mathbf{n}|\mathbf{n}-1)$ pelo mesmo motivo.

Julier e Uhlmann (1997) argumentam que o UKF contorna as limitações da linearização usada pelo EKF por usar uma representação de fato da *pdf*, não ficando presa a uma característica pontual da função não-linear. O UKF, porém, pode sofrer por um problema oposto. O espalhamento dos *pontos sigma* é determinado pelo tamanho N do vetor de estados. Dependendo de N e da função sendo aproximada, os *pontos sigma* podem ficar muito distantes da média, capturando características não-locais que podem prejudicar o desempenho do filtro. Funções trigonométricas (em especial a tangente) e funções que apresentem descontinuidades são especialmente suscetíveis a problemas por espalhamento excessivo.

Para contornar esse problema, Julier (2002) propõe o uso de parâmetros de escala na definição dos *pontos sigma* para controlar o espalhamento. Esse parâmetro é o parâmetro α visto nas equações anteriores (II.21, II.23).

Devido as diferenças entre o EKF e o UKF, não se pode dizer que um é sempre mais eficiente que o outro. Outros fatores podem ser importantes na escolha de qual dos dois usar. Devido ao uso dos *pontos sigma*, o UKF necessita realizar $2N + 1$ operações com cada função não-linear, enquanto o EKF só precisa realizar uma operação. O EKF, porém, depende do cálculo das derivadas dessas funções, que caso não sejam constantes, precisam ser calculadas a cada iteração, o que pode equilibrar o custo computacional dos dois filtros.

Filtro de Kalman Unscented Raiz Quadrada

O UKF sofre a mesma instabilidade numérica que o EKF no cálculo de $\mathbf{P}_x(\mathbf{n})$:

$$\mathbf{P}_x(\mathbf{n}) = \mathbf{P}_x(\mathbf{n}|\mathbf{n} - 1) - \mathbf{K}(\mathbf{n})\mathbf{P}_e(\mathbf{n}|\mathbf{n} - 1)\mathbf{K}(\mathbf{n})^T.$$

Para contornar esse problema é adotada uma implementação do UKF conhecida por Filtro de Kalman Unscented Raiz Quadrada, ou SR-UKF, que propaga o fator de Cholesky da matriz de covariância do estado no lugar da própria matriz de covariância. A covariância é obtida a partir da multiplicação do fator de Cholesky pelo seu transposto. Essa operação resulta sempre numa matriz positivo-definida, eliminando o problema. O SR-UKF tem uma vantagem adicional, que é a economia de uma operação de raiz quadrada exigida para o cálculo dos *pontos sigma*.

Para a implementação do SR-UKF são usadas duas ferramentas matemáticas ainda não introduzidas: a decomposição QR e atualização de fator de Cholesky (TEUKOLSKY; VETTERLING; FLANNERY, 1992):

Decomposição QR É a fatoração de uma matriz $\mathbf{A} \in \mathbb{R}^{L \times N}$ ($L \leq N$) em dois termos: $\mathbf{A}^T = \mathbf{Q}\mathbf{R}$, onde $\mathbf{Q} \in \mathbb{R}^{N \times N}$, e $\mathbf{R} \in \mathbb{R}^{N \times L}$ e tem um pedaço $\mathbf{R}'$ triangular superior, que é a transposta do fator de Cholesky da matriz de covariância², sendo que $\mathbf{P} = \mathbf{A}\mathbf{A}^T$. A notação $qr(\mathbf{X})$ indica a aplicação da decomposição QR sobre $\mathbf{X}$ retornando $\mathbf{R}'$.

Atualização de fator Cholesky Sendo $\mathbf{P} = \mathbf{S}\mathbf{S}^T$ e sendo $\mathbf{S}$ o fator de Cholesky de $\mathbf{P}$, a atualização de fator Cholesky implica em espelhar em $\mathbf{S}$ a seguinte operação em $\mathbf{P}$: $\mathbf{P} = \mathbf{P} + b.\mathbf{u}.\mathbf{u}^T$, onde b é uma constante e $\mathbf{u}$ um vetor qualquer. A notação $cholup(\mathbf{S}, \mathbf{u}, b)$ indica aplicação da atualização

²O fator de Cholesky é uma forma de se obter a raiz quadrada de uma matriz positivo-definida de modo que o fator seja uma matriz quadrada e triangular inferior, de modo que $\mathbf{R}' \neq \mathbf{A}$

de fator Cholesky. No caso de $\mathbf{u}$ ser uma matriz, basta executar a operação para cada coluna de $\mathbf{u}$.

A decomposição QR e a atualização de fator Cholesky são usadas para obter diretamente a partir dos *pontos sigma* e do fator de Cholesky da matriz de covariância da incertezas da observação ($\mathbf{R}(\mathbf{n})$) e da predição ($\mathbf{Q}(\mathbf{n})$) (que em geral são diagonais) o fator de Cholesky da matriz de covariância da estimativa do estado. A função da atualização de fator Cholesky é apenas de incorporar ao resultado da decomposição QR o primeiro *ponto sigma*, que é ignorado pela decomposição QR.

O primeiro passo é a obtenção dos *ponto sigma* $\mathcal{X}(\mathbf{n} - 1)$:

$$\begin{aligned}\mathcal{X}^{(0)}(\mathbf{n} - 1) &= \bar{\mathbf{x}}(\mathbf{n} - 1) \\ \mathcal{X}^{(i)}(\mathbf{n} - 1) &= \bar{\mathbf{x}}(\mathbf{n} - 1) + (\sqrt{(N + \delta)} \cdot \mathbf{S}_{\mathbf{x}}(\mathbf{n} - 1))^{(i)} \\ \mathcal{X}^{(i+N)}(\mathbf{n} - 1) &= \bar{\mathbf{x}}(\mathbf{n} - 1) - (\sqrt{(N + \delta)} \cdot \mathbf{S}_{\mathbf{x}}(\mathbf{n} - 1))^{(i)},\end{aligned}$$

onde $\mathbf{S}_{\mathbf{x}}(\mathbf{n} - 1)$ é o fator de Cholesky de $\mathbf{P}_{\mathbf{x}}(\mathbf{n} - 1)$ obtido na iteração anterior.

Em seguida $\bar{\mathbf{x}}(\mathbf{n}|\mathbf{n} - 1)$ e $\bar{\mathbf{y}}(\mathbf{n}|\mathbf{n} - 1)$ podem ser obtidas do mesmo modo:

$$\begin{aligned}\mathcal{X}^{(i)}(\mathbf{n}|\mathbf{n} - 1) &= f(\mathcal{X}^{(i)}(\mathbf{n} - 1)) \\ \bar{\mathbf{x}}(\mathbf{n}|\mathbf{n} - 1) &= \sum_{i=0}^{2n} W_m^{(i)} \mathcal{X}^{(i)}(\mathbf{n}|\mathbf{n} - 1) \\ \mathcal{Y}^{(i)}(\mathbf{n}|\mathbf{n} - 1) &= h(\mathcal{X}^{(i)}(\mathbf{n}|\mathbf{n} - 1)) \\ \bar{\mathbf{y}}(\mathbf{n}|\mathbf{n} - 1) &= \sum_{i=0}^{2n} W_m^{(i)} \mathcal{Y}^{(i)}(\mathbf{n}|\mathbf{n} - 1) \\ \mathbf{e}(\mathbf{n}) &= \mathbf{y}(\mathbf{n}) - \bar{\mathbf{y}}(\mathbf{n}|\mathbf{n} - 1),\end{aligned}$$

assim como $\mathbf{P}_{\mathbf{x}\mathbf{e}}(\mathbf{n}|\mathbf{n} - 1)$:

$$\begin{aligned}\mathbf{P}_{\mathbf{x}\mathbf{e}}(\mathbf{n}|\mathbf{n} - 1) &= \\ \sum_{i=0}^{2n} W_i^c (\mathcal{X}(\mathbf{n}|\mathbf{n} - 1)_i - \bar{\mathbf{x}}(\mathbf{n}|\mathbf{n} - 1)) (\mathcal{Y}(\mathbf{n}|\mathbf{n} - 1)_i - \bar{\mathbf{y}}(\mathbf{n}|\mathbf{n} - 1))^T.\end{aligned}$$

Em vez de se calcular $\mathbf{P}_{\mathbf{x}}(\mathbf{n}|\mathbf{n} - 1)$, calcula-se o seu fator de Cholesky, $\mathbf{S}_{\mathbf{x}}(\mathbf{n}|\mathbf{n} - 1)$, aplicando primeiro a decomposição QR seguida pela atualização do fator de Cholesky para levar em conta a primeira coluna de *pontos sigma*:

$$\begin{aligned}\mathbf{S}'_{\mathbf{x}}(\mathbf{n}|\mathbf{n} - 1) &= qr([\sqrt{W_c^{(i)}} \mathcal{X}^{(1, \dots, 2N)}(\mathbf{n}), \sqrt{\mathbf{Q}(\mathbf{n})})] \\ \mathbf{S}_{\mathbf{x}}(\mathbf{n}|\mathbf{n} - 1) &= cholup(\mathbf{S}'_{\mathbf{x}}(\mathbf{n}|\mathbf{n} - 1), \mathcal{X}^{(0)}(\mathbf{n}) - \bar{\mathbf{x}}(\mathbf{n}|\mathbf{n} - 1), W_c^{(0)}),\end{aligned}$$

onde $[\sqrt{W_c^{(i)}} \mathcal{X}^{(1, \dots, 2N)}(\mathbf{n}), \sqrt{\mathbf{Q}(\mathbf{n})}]$ representa uma composição de matrizes cuja primeiras colunas são dadas por $\sqrt{W_c^{(i)}} \mathcal{X}^{(1, \dots, 2N)}(\mathbf{n})$ e as colunas adjacentes por $\sqrt{\mathbf{Q}(\mathbf{n})}$, e $\mathcal{X}^{(1, \dots, 2N)}(\mathbf{n})$ indica uma matriz formada pelas colunas de 1 a $2N$ da matriz $\mathcal{X}(\mathbf{n})$, sem a primeira coluna, referenciada por 0.

A mesma seqüência de operações é realizada na obtenção do fator de Cholesky de $\mathbf{P}_e(\mathbf{n}|\mathbf{n}-1)$, $\mathbf{S}_y(\mathbf{n}|\mathbf{n}-1)$:

$$\begin{aligned} \mathbf{S}'_y(\mathbf{n}|\mathbf{n}-1) &= qr(\sqrt{W_c^{(i)}} \mathcal{Y}^{(1, \dots, 2N)}(\mathbf{n}) | \sqrt{R(n)}) \\ \mathbf{S}_y(\mathbf{n}|\mathbf{n}-1) &= cholup(\mathbf{S}'_y(\mathbf{n}|\mathbf{n}-1), \mathcal{Y}^{(0)}(\mathbf{n}) - \bar{\mathbf{y}}(\mathbf{n}|\mathbf{n}-1), W_c^{(0)}). \end{aligned}$$

Usando essas matrizes calcula-se $\mathbf{K}(\mathbf{n})$:

$$\mathbf{K}(\mathbf{n}) = \mathbf{P}_{xe}(\mathbf{n}|\mathbf{n}-1) \cdot (\mathbf{S}_y(\mathbf{n}|\mathbf{n}-1) \mathbf{S}_y(\mathbf{n}|\mathbf{n}-1)^T)^{-1},$$

e $\mathbf{S}_x(\mathbf{n})$:

$$\mathbf{S}_x(\mathbf{n}) = cholup(\mathbf{S}_x(\mathbf{n}|\mathbf{n}-1), \mathbf{K}(\mathbf{n}) \cdot \mathbf{S}_y(\mathbf{n}|\mathbf{n}-1), -1),$$

lembrando que o SR-UKF propaga para a próxima iteração a matriz $\mathbf{S}_x(\mathbf{n})$ e o vetor $\bar{\mathbf{x}}(\mathbf{n})$.

II.6 Estimadores Não Lineares

Apesar da boa precisão obtida com o EKF e UKF para sistemas não-lineares, esses métodos não são ideais para todas as aplicações. A aproximação de que a evolução do estado e as observações estão sujeitas a apenas incertezas que seguem distribuições gaussianas nem sempre é válida, resultando na degradação da precisão da estimativa obtida, ou mesmo na incapacidade de se obter uma estimativa. No extremo, tem-se situações onde as *pdfs* da evolução do estado e das observações são multi-modais, ou seja, com mais de um ponto de máximo local. Para essas situações, soluções derivadas diretamente do Filtro de Kalman são quase sempre ineficazes. Para tratar esses casos, é necessário trabalhar com métodos que sejam menos restritivos quanto ao formato das *pdfs*.

II.6.1 Localização Markoviana

Um método que possibilita a princípio trabalhar com qualquer tipo de *pdf* é conhecido como Localização Markoviana, ou Localização em Grade (FOX; BUR-

GARD; THRUN, 1998). O método mantém uma representação da *pdf* do estado no instante atual. Para realizar isso, o espaço de estados é dividido em células, como uma grade, e calcula o valor da *pdf* para cada célula. O método só pode ser usado se o espaço de estados não for infinito e aceitar ser discretizado.

Gutmann (1998) realiza uma comparação entre duas versões do EKF e duas versões da localização em Grade, e chegam à conclusão que os algoritmos em Grade são mais robustos a erros grosseiros de observação e ao problema do robô raptado ³, porém o EKF obtém resultados mais precisos quando as medições são ‘bem comportadas’ (sem muitos erros grosseiros de observação). Esse é um resultado interessante, pois mostra que aparentemente o EKF obtém resultados melhores que o algoritmo em Grade. Olhando com mais cuidado o experimento, pode-se atribuir a conclusão que os autores chegaram ao tamanho das células adotadas pelos algoritmos em Grade, que não foi especificado, e ao método usado para adicionar erro aos dados usados para os testes. Foi adicionado ruído gaussiano ao conjunto de dados de teste. Como o Filtro de Kalman é ótimo para processos gaussianos, é esperado um bom desempenho do EKF, desde que a linearização dos modelos não introduza grande distorção. Apesar das ressalvas, é possível concluir dessa comparação que o EKF (e os demais derivados do Filtro de Kalman), apesar de ser muito mais rápido que os algoritmos de localização em Grade, podem apresentar resultados tão bons quanto ou melhores que estes, dependendo do problema abordado.

Apesar da Localização Markoviana ser o estimador bayesiano mais robusto, o método consome muitos recursos computacionais, devido à necessidade de manter atualizada a distribuição de probabilidade por todo o espaço de estados. Para obter uma boa resolução num ambiente de médio ou grande porte, o número de células necessárias torna o uso do algoritmo impraticável em sistemas de resposta rápida.

II.6.2 Amostragem por Importância com Reamostragem

Uma terceira família de métodos usados para localização pondera as qualidades e fraquezas das abordagens anteriores (Filtro de Kalman e Localização Markoviana). Essa família de métodos é conhecida por uma série de nomes distintos, como Filtro Bootstrap (GORDON; SALMOND; SMITH, 1993), Condensação (ISARD; BLAKE, 1998), Localização Monte Carlo (KITAGAWA, 1996) e Filtro de Partículas

³O objetivo do problema do robô raptado é determinar a localização de um robô navegando em um ambiente conhecido, partindo de uma posição inicial desconhecida.

(MERWE, 2001).

Para obter a representação da *pdf*, em vez de manter um mapa completo da *pdf*, o Filtro de Partículas – PF – mantém um conjunto de amostras da *pdf* do estado, de modo que a partir dessas amostras seja possível caracterizar adequadamente a *pdf*. Quanto mais complexa se supõe que a *pdf* seja, mais partículas são necessárias para descrevê-la, porém mais custoso em tempo de processamento o filtro se tornará. Com um número suficientemente grande de partículas, qualquer *pdf* pode ser representada, com a vantagem de não sofrer da perda de precisão da Localização Markoviana imposta pela discretização do espaço de estado.

Pode-se notar que o PF assemelha-se ao UKF, onde um conjunto de partículas também é selecionado para representar a distribuição. A principal diferença é que no UKF as partículas são escolhidas de modo determinístico, considerando a *pdf* gaussiana, e são descartadas em seguida, levando para a próxima iteração apenas os momentos de primeira e segunda ordem da *pdf*. No PF as partículas são selecionadas segundo algum método, porém com um grau de aleatoriedade, e as partículas são mantidas por várias iterações.

A seguinte notação será utilizada para facilitar a visualização das equações:

$$\begin{aligned}\mathbf{X}(\mathbf{n}) &= \{\mathbf{x}(\mathbf{0}), \dots, \mathbf{x}(\mathbf{n})\} \\ \mathbf{Y}(\mathbf{n}) &= \{\mathbf{y}(\mathbf{1}), \dots, \mathbf{y}(\mathbf{n})\}.\end{aligned}$$

A dedução feita a seguir é baseada no trabalho de Merwe (2001). O desenvolvimento do PF começa assumindo que é muito difícil (em geral, impossível) amostrar diretamente da distribuição real $p(\mathbf{X}(\mathbf{n})|\mathbf{Y}(\mathbf{n}))$. Para contornar essa dificuldade, é utilizada uma distribuição auxiliar, $q(\mathbf{X}(\mathbf{n})|\mathbf{Y}(\mathbf{n}))$, escolhida de modo a aproximar a distribuição real, porém de onde é mais fácil gerar amostras. A expectativa de $\mathbf{X}(\mathbf{n})$ pode ser dada por:

$$\begin{aligned}E_x(\mathbf{X}(\mathbf{n})) &= \int \mathbf{X}(\mathbf{n})p(\mathbf{X}(\mathbf{n})|\mathbf{Y}(\mathbf{n}))d\mathbf{X}(\mathbf{n}) \\ &= \int \mathbf{X}(\mathbf{n})p(\mathbf{X}(\mathbf{n})|\mathbf{Y}(\mathbf{n}))\frac{q(\mathbf{X}(\mathbf{n})|\mathbf{Y}(\mathbf{n}))}{q(\mathbf{X}(\mathbf{n})|\mathbf{Y}(\mathbf{n}))}d\mathbf{X}(\mathbf{n}).\end{aligned}$$

Aplicando o teorema de Bayes sobre $p(\mathbf{X}(\mathbf{n})|\mathbf{Y}(\mathbf{n}))$ e rearranjando os termos,

a expectativa fica:

$$\begin{aligned}
 E_x(\mathbf{X}(\mathbf{n})) &= \int \mathbf{X}(\mathbf{n}) \frac{p(\mathbf{Y}(\mathbf{n})|\mathbf{X}(\mathbf{n}))p(\mathbf{X}(\mathbf{n}))}{q(\mathbf{X}(\mathbf{n})|\mathbf{Y}(\mathbf{n}))p(\mathbf{Y}(\mathbf{n}))} q(\mathbf{X}(\mathbf{n})|\mathbf{Y}(\mathbf{n})) d\mathbf{X}(\mathbf{n}) \\
 &= \frac{1}{p(\mathbf{Y}(\mathbf{n}))} \int \mathbf{X}(\mathbf{n}) w_n(\mathbf{X}(\mathbf{n})) q(\mathbf{X}(\mathbf{n})|\mathbf{Y}(\mathbf{n})) d\mathbf{X}(\mathbf{n}) \\
 &= \frac{1}{p(\mathbf{Y}(\mathbf{n}))} E_q(w_n(\mathbf{X}(\mathbf{n}))\mathbf{X}(\mathbf{n})),
 \end{aligned}$$

onde $E_q(\cdot)$ indica a expectativa tomada sobre a densidade de probabilidade $q(\cdot|\mathbf{Y}(\mathbf{n}))$ e $w_n(\mathbf{X}(\mathbf{n}))$ é chamado de peso da importância, dado por:

$$w_n(\mathbf{X}(\mathbf{n})) = \frac{p(\mathbf{Y}(\mathbf{n})|\mathbf{X}(\mathbf{n}))p(\mathbf{X}(\mathbf{n}))}{q(\mathbf{X}(\mathbf{n})|\mathbf{Y}(\mathbf{n}))}. \quad (\text{II.25})$$

Aplicando o teorema da probabilidade total sobre $p(\mathbf{Y}(\mathbf{n}))$, a expectativa pode ser escrita como:

$$\begin{aligned}
 E_x(\mathbf{X}(\mathbf{n})) &= \frac{E_q(w_n(\mathbf{X}(\mathbf{n}))\mathbf{X}(\mathbf{n}))}{\int p(\mathbf{Y}(\mathbf{n})|\mathbf{X}(\mathbf{n}))p(\mathbf{X}(\mathbf{n})) \frac{q(\mathbf{X}(\mathbf{n})|\mathbf{Y}(\mathbf{n}))}{q(\mathbf{X}(\mathbf{n})|\mathbf{Y}(\mathbf{n}))} d\mathbf{X}(\mathbf{n})} \\
 &= \frac{E_q(w_n(\mathbf{X}(\mathbf{n}))\mathbf{X}(\mathbf{n}))}{\int w_n(\mathbf{X}(\mathbf{n})) q(\mathbf{X}(\mathbf{n})|\mathbf{Y}(\mathbf{n})) d\mathbf{X}(\mathbf{n})} \\
 &= \frac{E_q(w_n(\mathbf{X}(\mathbf{n}))\mathbf{X}(\mathbf{n}))}{E_q(w_n(\mathbf{X}(\mathbf{n})))}.
 \end{aligned}$$

Considerando que é possível gerar amostras a partir de q , é possível aproximar as expectativas acima por somatórias sobre um conjunto de amostras extraídas de q . Considerando um conjunto de N_s vetores amostrados da *pdf* $q(\mathbf{X}(\mathbf{n})|\mathbf{Y}(\mathbf{n}))$ representados por $\mathbf{x}^i(\mathbf{n})$, e $\mathbf{X}^i(\mathbf{n}) = \{\mathbf{x}^i(\mathbf{0}), \dots, \mathbf{x}^i(\mathbf{n})\}$, sendo $i = 1, \dots, N_s$, pode-se determinar $E_x(\mathbf{X}(\mathbf{n}))$ da seguinte forma:

$$\begin{aligned}
 E_x(\mathbf{X}(\mathbf{n})) &\approx \frac{\sum_{i=0}^{N_s} w_n(\mathbf{X}^i(\mathbf{n}))\mathbf{X}^i(\mathbf{n})}{\sum_{i=0}^{N_s} w_n(\mathbf{X}^i(\mathbf{n}))} \\
 &\approx \sum_{i=0}^{N_s} \tilde{w}_n(\mathbf{X}^i(\mathbf{n}))\mathbf{X}^i(\mathbf{n}),
 \end{aligned}$$

onde:

$$\tilde{w}_n(\mathbf{X}^i(\mathbf{n})) = \frac{w_n(\mathbf{X}^i(\mathbf{n}))}{\sum_{i=0}^{N_s} w_n(\mathbf{X}^i(\mathbf{n}))}.$$

Das equações acima fica claro o porquê do nome dado a w_n . Cada amostra gerada da distribuição auxiliar é ponderada pelo peso da importância para gerar

a estimativa. Agora é possível calcular a estimativa de $x(n)$ bastando apenas saber como calcular w_n . Para isso, é necessário introduzir as premissas que todos os processos envolvidos são markovianos de ordem 1, incluindo a distribuição q , e que cada observação depende apenas do estado no instante em que foi realizada:

$$q(\mathbf{X}(\mathbf{n})|\mathbf{Y}(\mathbf{n})) = q(\mathbf{X}(\mathbf{n}-1)|\mathbf{Y}(\mathbf{n}-1))q(\mathbf{x}(\mathbf{n})|\mathbf{X}(\mathbf{n}-1), \mathbf{Y}(\mathbf{n})) \quad (\text{II.26})$$

$$p(\mathbf{X}(\mathbf{n})) = p(\mathbf{x}(\mathbf{0})) \prod_{j=1}^n p(\mathbf{x}(\mathbf{j})|\mathbf{x}(\mathbf{j}-1)) \quad (\text{II.27})$$

$$p(\mathbf{Y}(\mathbf{n})|\mathbf{X}(\mathbf{n})) = \prod_{j=1}^n p(\mathbf{y}(\mathbf{j})|\mathbf{x}(\mathbf{j})). \quad (\text{II.28})$$

Multiplicando II.25 por $\frac{w_{n-1}(\mathbf{X}(\mathbf{n}-1))}{w_{n-1}(\mathbf{X}(\mathbf{n}-1))}$ tem-se:

$$\begin{aligned} w_n(\mathbf{X}(\mathbf{n})) &= \frac{w_{n-1}(\mathbf{X}(\mathbf{n}-1))}{w_{n-1}(\mathbf{X}(\mathbf{n}-1))} \frac{p(\mathbf{Y}(\mathbf{n})|\mathbf{X}(\mathbf{n}))p(\mathbf{X}(\mathbf{n}))}{q(\mathbf{X}(\mathbf{n})|\mathbf{Y}(\mathbf{n}))} \\ &= w_{n-1}(\mathbf{X}(\mathbf{n}-1)) \frac{q(\mathbf{X}(\mathbf{n}-1)|\mathbf{Y}(\mathbf{n}-1))}{p(\mathbf{Y}(\mathbf{n}-1)|\mathbf{X}(\mathbf{n}-1))p(\mathbf{X}(\mathbf{n}-1))} \\ &\quad \frac{p(\mathbf{Y}(\mathbf{n})|\mathbf{X}(\mathbf{n}))p(\mathbf{X}(\mathbf{n}))}{q(\mathbf{X}(\mathbf{n})|\mathbf{Y}(\mathbf{n}))}. \end{aligned} \quad (\text{II.29})$$

Substituindo II.26 em II.29:

$$\begin{aligned} w_n(\mathbf{X}(\mathbf{n})) &= w_{n-1}(\mathbf{X}(\mathbf{n}-1)) \frac{q(\mathbf{X}(\mathbf{n}-1)|\mathbf{Y}(\mathbf{n}-1))}{p(\mathbf{Y}(\mathbf{n}-1)|\mathbf{X}(\mathbf{n}-1))p(\mathbf{X}(\mathbf{n}-1))} \\ &\quad \frac{p(\mathbf{Y}(\mathbf{n})|\mathbf{X}(\mathbf{n}))p(\mathbf{X}(\mathbf{n}))}{q(\mathbf{X}(\mathbf{n}-1)|\mathbf{Y}(\mathbf{n}-1))q(\mathbf{x}(\mathbf{n})|\mathbf{X}(\mathbf{n}-1), \mathbf{Y}(\mathbf{n}))} \\ &= w_{n-1}(\mathbf{X}(\mathbf{n}-1)) \frac{1}{p(\mathbf{Y}(\mathbf{n}-1)|\mathbf{X}(\mathbf{n}-1))p(\mathbf{X}(\mathbf{n}-1))} \\ &\quad \frac{p(\mathbf{Y}(\mathbf{n})|\mathbf{X}(\mathbf{n}))p(\mathbf{X}(\mathbf{n}))}{q(\mathbf{x}(\mathbf{n})|\mathbf{X}(\mathbf{n}-1), \mathbf{Y}(\mathbf{n}))} \\ &= w_{n-1}(\mathbf{X}(\mathbf{n}-1)) \frac{p(\mathbf{X}(\mathbf{n}))}{p(\mathbf{X}(\mathbf{n}-1))} \\ &\quad \frac{p(\mathbf{Y}(\mathbf{n})|\mathbf{X}(\mathbf{n}))}{p(\mathbf{Y}(\mathbf{n}-1)|\mathbf{X}(\mathbf{n}-1))} \frac{1}{q(\mathbf{x}(\mathbf{n})|\mathbf{X}(\mathbf{n}-1), \mathbf{Y}(\mathbf{n}))}. \end{aligned} \quad (\text{II.30})$$

Finalmente, substituindo II.27 e II.28 em II.30:

$$w_n(\mathbf{X}(\mathbf{n})) = w_{n-1}(\mathbf{X}(\mathbf{n}-1)) \frac{p(\mathbf{y}(\mathbf{n})|\mathbf{x}(\mathbf{n}))p(\mathbf{x}(\mathbf{n})|\mathbf{x}(\mathbf{n}-1))}{q(\mathbf{x}(\mathbf{n})|\mathbf{X}(\mathbf{n}-1), \mathbf{Y}(\mathbf{n}))}.$$

Essa expressão fornece um método recursivo para calcular os pesos. A única variável ainda não definida é a distribuição auxiliar $q(x(n)|\mathbf{X}(\mathbf{n}-1), \mathbf{Y}(\mathbf{n}))$. A escolha dessa distribuição é determinante para o comportamento dos PF. Alguns filtros derivam do PF básico alterando essa distribuição. Segundo Doucet, Gordon e Krishnamurthy (1999), é interessante a escolha de uma distribuição que

minimize a variância dos pesos condicionados a $\mathbf{X}(\mathbf{n} - 1)$ e $\mathbf{Y}(\mathbf{n})$, e prova que a distribuição $p(x(n)|\mathbf{X}(\mathbf{n} - 1), \mathbf{Y}(\mathbf{n}))$ minimiza essa variância. Apesar disso, a escolha mais comum é a distribuição $p(\mathbf{x}(\mathbf{n})|\mathbf{x}(\mathbf{n} - 1))$, pois simplifica a implementação.

O elemento principal do PF são os pares formados por uma amostra, x^i e o peso associado, w^i . A cada iteração do filtro cada par é atualizado de acordo com as novas observações disponíveis. Para extrair a estimativa do estado numa dada iteração, devem-se combinar os valores das partículas ponderadas pelos respectivos pesos.

O Filtro de Partículas básico está apresentado no algoritmo 1.

Algorithm 1 Filtro de Partículas

Inicialização:

$$\bar{\mathbf{x}}(\mathbf{0}) = \{\mathbf{x}^1(\mathbf{0}), \dots, \mathbf{x}^{N_s}(\mathbf{0})\}; \mathbf{x}^i(\mathbf{0}) \sim p(\mathbf{x}(\mathbf{0}))$$

$$w(\mathbf{0}) = \{w^1(\mathbf{0}), \dots, w^{N_s}(\mathbf{0})\}; w^i(\mathbf{0}) = \frac{1}{N_s}$$

for $n = 0$ to ... **do**

 Amostragem por Importância:

$$\bar{\mathbf{x}}^i(\mathbf{n}) \sim q(\mathbf{x}(\mathbf{n})|\mathbf{X}^i(\mathbf{n}), \mathbf{Y}(\mathbf{n}))$$

$$w^i(n) = w^i(n-1) \frac{p(\mathbf{y}(\mathbf{n})|\bar{\mathbf{x}}(\mathbf{n}))p(\bar{\mathbf{x}}(\mathbf{n})|\mathbf{x}(\mathbf{n}-1))}{q(\bar{\mathbf{x}}(\mathbf{n})|\mathbf{X}(\mathbf{n}-1), \mathbf{Y}(\mathbf{n}))}$$

$$\tilde{w}^i(n) = \frac{w^i(n)}{\sum_{i=0}^{N_s} w^i(n)}$$

 Reamostragem:

if $ESS(w(n)) < \beta N_s$ **then** $\{\beta \text{ é um parâmetro ajustável}\}$

$$\bar{\mathbf{x}}(\mathbf{n}) = REAMOSTRAR(\tilde{w}(n), \hat{x}(n))$$

$$\tilde{w}(n) = \frac{1}{N_s}$$

end if

 Obtenção da estimativa:

$$x(n) = ESTIMA(\tilde{w}(n), \bar{\mathbf{x}}(\mathbf{n}))$$

end for

Como está apresentado, o filtro ainda apresenta uma limitação que precisa ser corrigida. É provado por Doucet, Gordon e Krishnamurthy (1999) que a variância dos pesos tende a crescer com o tempo. Isso significa que com o passar do tempo os pesos vão se distanciando uns dos outros, fazendo com que uns poucos pesos (ou mesmo apenas um peso) que se deslocaram para perto do pico da *pdf* acumulem praticamente toda importância, deixando os demais pesos com valores insignificantes. Esse efeito é chamado de degeneração do conjunto de partículas. Para evitar essa degeneração, é necessário reamostrar o conjunto de partículas de tempos em tempos para manter a variância dos pesos controlada.

O procedimento básico da reamostragem é escolher entre as amostras existentes quais devem ser mantidas e quais devem ser eliminadas por apresentarem peso

muito pequeno. Em seguida, as amostras mais significativas devem ser replicadas para manter o número de partículas em N_s .

O algoritmo de reamostragem mais simples usado é conhecido por SIR, sigla em inglês para Reamostragem da Amostragem por Importância. O algoritmo 2 apresenta uma implementação do SIR com complexidade computacional $O(N_s)$ do algoritmo, apresentada por Carpenter, Clifford e Fernhead (1999), onde N_s é o número de partículas usadas no algoritmo e $U[0, 1]$ indica uma distribuição uniforme entre zero e um.

Algorithm 2 SIR

```

 $\tilde{w}^i = \frac{w^i}{\sum_{i=0}^{N_s} w^i}$ 
 $W_{sum} = \{\sum_{i=0}^j \tilde{w}^i\}; 1 \leq j \leq N_s$ 
 $R = \{-\log(t_1), \dots, -\log(t_{N_s+1})\}; t_i \sim U[0, 1]$ 
 $R_{sum} = \{\sum_{i=0}^j \tilde{T}(j)\}; 1 \leq j \leq N_s + 1$ 
 $RN_{sum} = \{R_{sum}(1)/R_{sum}(N_s + 1), \dots, R_{sum}(1)/R_{sum}(N_s + 1)\}$ 
 $i = 1, j = 1;$ 
for  $i = 0$  to  $N_s$  do
  if  $RN_{sum}(i) < W_{sum}(j)$  then
     $x_{novo}^i = x^i$ 
     $i = i + 1$ 
  else
     $j = j + 1$ 
  end if
end for
return  $x_{novo}^i$ 

```

Outros algoritmos apresentados na literatura tentam melhorar o desempenho do algoritmo básico, tentando minimizar a variância do novo conjunto de pesos. Merwe (2001) apresenta um algoritmo que produz resultados melhores que o SIR básico.

A reamostragem não deve ser realizada a cada iteração do filtro, mas sim apenas quando for detectada que a variância das partículas está muito grande. Para determinar quando uma reamostragem deve ser realizada, Liu, Chen e Logvinenko (2000) sugerem usar o Tamanho Efetivo da Amostra (ESS da sigla em inglês):

$$\begin{aligned}
cv_n^2 &= \frac{VAR(w(n))}{E^2(w(n))} \\
cv_n^2 &= \frac{1}{N_s} \sum_{i=1}^{N_s} (Mw^i(n) - 1)^2 \\
ESS_n &= \frac{M}{1 + cv_n^2}
\end{aligned} \tag{II.31}$$

O PF, apesar de lidar bem com distribuições de probabilidade complexas, não oferece a mesma precisão que o UKF ou o EKF em distribuições monomodais, em especial para aquelas próximas às distribuições gaussianas. Foram propostas na literatura uma série de adaptações ao PF para melhorar sua precisão, robustez, diminuir o número de partículas necessárias e lidar com situações específicas. Essas adaptações podem tornar o PF mais preciso que os Filtros de Kalman, mesmo para distribuições próximas a gaussiana, e diminuir a diferença de complexidade computacional entre as duas famílias de estimadores. Para exemplificar, algumas dessas adaptações são apresentadas a seguir:

Reinicialização Sensorial da localização (SRL) Proposto por Lenser e Veloso (2000), o objetivo dessa abordagem é tornar o filtro mais robusto a erros de modelagem do modelo de deslocamento, que podem levar o filtro a não se comportar bem se o robô mover-se muito tempo sem leituras sensoriais ou se a incerteza associada ao modelo de deslocamento for muito menor do que a incerteza real (incerteza mal dimensionada). Ao fim de cada iteração do filtro, algumas partículas são trocadas por outras amostradas diretamente de $p(\mathbf{y}(\mathbf{n})|\mathbf{x}(\mathbf{n}))$. A limitação dessa técnica é a necessidade de gerar amostras de $\mathbf{x}(\mathbf{n})$ a partir de $p(\mathbf{y}(\mathbf{n})|\mathbf{x}(\mathbf{n}))$, pois, como dito anteriormente, nem sempre $\mathbf{y} = f(\mathbf{x})$ é inversível.

Filtro de Partículas Adaptativo (A-PF) A idéia dessa alteração é permitir que o filtro determine dinamicamente o número de partículas que serão mantidas. Fox (1999) descreve um procedimento para escolha do número de partículas baseado na divergência das *pdfs* $p(\mathbf{x}(\mathbf{n})|\mathbf{y}(\mathbf{n}))$ e $p(\mathbf{x}(\mathbf{n})|\mathbf{x}(\mathbf{n}-1))$. Quanto mais separadas (ou menos casadas) estão essas *pdfs*, maior o número de partículas necessárias. Assim, se os sensores fornecerem observações de má qualidade, o robô é reposicionado ou muda bruscamente de rota (devido a colisão, por exemplo), então mais partículas são geradas até que ambos os modelos (de medição e dinâmica) voltem a concordar. É esperado com esse estimador diminuir o tempo de execução médio do PF, tendo um número

médio de partículas menor, e ao mesmo aumentar a capacidade de recuperação em situações de descontinuidade do deslocamento.

Filtro de Partículas UKF ou EKF (UKF-PF, EKF-PF) Como dito anteriormente, um modo de melhorar a precisão do PF é escolher uma boa distribuição para $q(\mathbf{X}(\mathbf{n})|\mathbf{Y}(\mathbf{n}))$. Doucet (1998) sugere usar o EKF, enquanto Merwe (2001) sugerem o UKF. Em ambos os casos a mudança no PF básico consiste em trocar a distribuição $p(\mathbf{x}^i(\mathbf{n})|\mathbf{x}^i(\mathbf{n}-1))$ pela distribuição normal $N(\bar{\mathbf{x}}^i(\mathbf{n}), \bar{\mathbf{P}}^i(\mathbf{n}))$, onde $\bar{\mathbf{x}}^i(\mathbf{n})$ e $\bar{\mathbf{P}}^i(\mathbf{n})$ são a estimativa e covariância do estado estimado por um Filtro de Kalman, respectivamente. Segundo Merwe (2001), essa variante consegue obter resultados muito mais precisos do que o PF básico.

Apesar das diversas melhorias propostas ao PF, ainda existe uma diferença grande de complexidade computacional entre esses estimadores e os baseados em KF. Por exemplo, para um estado de dimensão 3 (x , y e θ , por exemplo), o número mínimo de partículas para obter ganho de desempenho em relação a um KF foi de 500. A medida que a dimensão do estado aumenta, o número de partículas tende a aumentar geometricamente. A robustez oferecida pelo PF é a maior vantagem que sobre um KF, pois os PFs são menos sensíveis às formas das *pdfs* e das funções não-lineares envolvidas.

II.7 Outros Estimadores

Além dos estimadores já apresentados, outros estimadores foram propostos com objetivo de corrigir limitações das técnicas anteriores. Para exemplificar a diversidade existente, alguns deles são apresentados a seguir:

Localização Multi-hipótese (MHL) Jensfelt e Kristensen (2001) apresentam um estimador um pouco diferente dos vistos anteriormente. A distribuição de probabilidade posterior é aproximada por uma gaussiana, porém são consideradas n hipóteses para a gaussiana. As hipóteses são criadas a partir da *pdf* $p(\mathbf{x}(\mathbf{n})|\mathbf{y}(\mathbf{n}))$, e eliminadas quando a probabilidade de serem verdadeiras estiverem abaixo de um limiar. Cada hipótese é mantida por um EKF.

Localização Markoviana – Filtro de Kalman Estendido (ML-EKF) Gutmann (2002) apresenta um algoritmo que combina a robustez da Localização Markoviana com a velocidade e precisão do EKF. O algoritmo mantém a dis-

tribuição de probabilidade posterior global do estado usando localização Markoviana sobre um espaço de estados reduzido e com menor resolução. Por exemplo, caso o estado fosse representado por x , y e θ , a Localização Markoviana seria aplicada apenas sobre x , y , discretizando o espaço de estado em células de $30cm^2$, por exemplo.

No ponto de máxima da Localização Markoviana é usado um EKF para refinar a estimativa na região onde a presença do robô é mais provável. A Localização Markoviana é então usada para detectar quando a estimativa do EKF não é mais plausível, e reinicia o EKF para a nova situação.