3
Local SLAM

Os algoritmos de SLAM (Simultaneous Localisation and Mapping) surgiram da necessidade de se enviar robôs por ambientes desconhecidos. Os primeiros algoritmos usavam dados de odometria e sensores de distância para ir mapeando o que tinha a frente do robô(figura 3.1), e depois para saber onde ele estava naquele mapa. Há poucos anos foi possível através de computadores e algoritmos melhores fazer tanto o mapeamento quanto a localização usando apenas uma câmera. Essa família de algoritmos foi chamada de Visual SLAM. Em Visual SLAM, a câmera acoplada ao robô tem a sua posição rastreada ao longo do tempo. Logo os pesquisadores perceberam que esse conceito era idêntico ao rastreamento de câmera para realidade aumentada. No entanto, um problema sério surgiu e até hoje está mal resolvido, o erro acumulado, como foi explicado na seção de trabalhos relacionados no trabalho de Davison[14]. O SLAM é usado pelos robôs apenas para evitar obstáculos desconhecidos e ter uma localização razoável do robô no espaço. Já as aplicações de realidade aumentada que propomos acontecem em ambiente conhecido e é essa informação que usamos para propor o conceito de Local SLAM. Como explicado no capítulo de trabalhos relacionados, não podemos esperar que um mapa feito previamente vá funcionar em um segundo momento, por isso propomos usar um algoritmo de Visual SLAM apenas localmente, entre dois marcadores previamente mapeados. Assim que localizado um novo marcador a posição da câmera é reinicializada, o mapa do SLAM é apagado e um novo começa a ser gerado.

Nessa seção é detalhado o algoritmo proposto. Primeiramente, vamos explicar qual tipo de marcador fiducial é usado. Em seguida como rastreamos marcadores naturais. Então explicamos como fazer o mapeamento do ambiente. Em seguida explicamos as modificações feitas num algoritmo de Visual SLAM para suportar o Local SLAM. Em seguida explicamos quando decidimos reinicializar o algoritmo de SLAM. Por fim é descrito o processamento geral do algoritmo.

PUC-Rio - Certificação Digital Nº 0721334/CA

Figura 3.1: Uma montagem clássica de robôs para usar técnicas de SLAM. Fonte: [28]

3.1 Marcadores fiduciais

Hoje em dia, a forma mais popular e simples de identificar objetos e fazer RA é colocar marcadores fiduciais ao redor do objeto. Os arcabouços mais utilizados para marcar, reconhecer e rastrear objetos são o ARToolkit [22], ARTag [17] e muitos outros derivados deles, como por exemplo o ARToolKitPlus[37] e o Studierstube Tracker[31]. Nessas bibliotecas foram adicionadas extensões e otimizações à detecção e ao rastreamento. Além disso, essas bibliotecas são fáceis de usar e são extremamente eficientes em computadores de mesa. Elas conseguem de forma geral fazer o processamento de uma imagem 640x480 em menos de 5ms dependendo do número de marcadores na imagem.

No nosso caso preferimos utilizar a biblioteca ARToolKitPlus(figura 3.2) por ela usar uma matriz 2D com um código do tipo BCH que é capaz de se autovalidar, o que seria análogo a um dígito de segurança. Usando uma matriz 6x6, a biblioteca é capaz de codificar cerca de 4.000 números, mais do que suficiente para nosso algoritmo.

Essa biblioteca fornece a posição da câmera em relação ao marcador. Consideramos posição de câmera como sendo o par (posição do ponto focal da câmera, direção para onde ela está olhando). A fim de que a posição da câmera seja dada em coordenadas métricas, temos uma base de dados com o exato tamanho de cada marcador, assim após identificado o código do marcador, ele é consultado numa base de dados para saber qual o tamanho em metros daquele marcador. Além disso, a biblioteca também fornece as coordenadas no plano da imagem dos quatro vértices do quadrado preto que envolve o marcador.

Como essa biblioteca é vastamente documentada e utilizada, não vamos entrar em maiores detalhes sobre seu funcionamento.

PUC-Rio - Certificação Digital Nº 0721334/CA

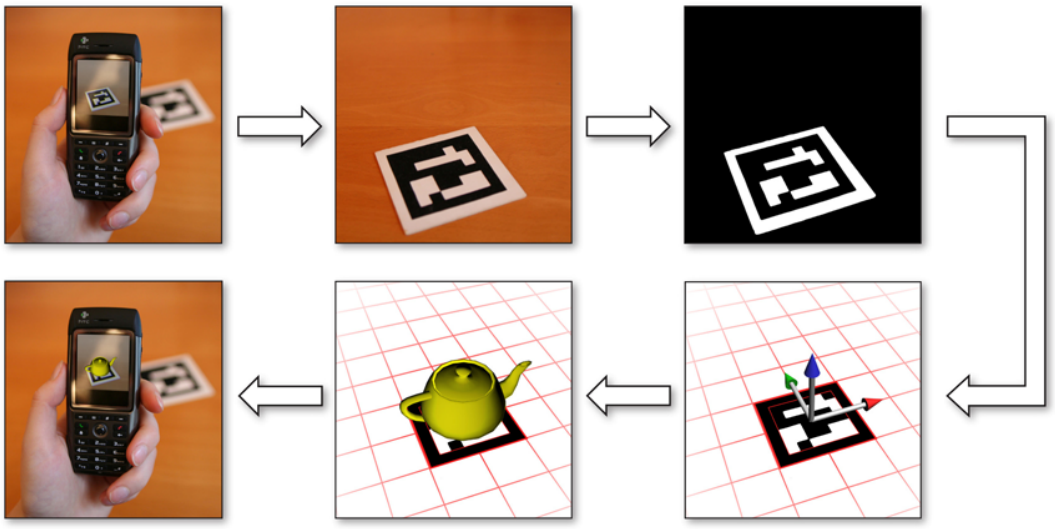

Figura 3.2: Fluxo de processamento de uma aplicação de RA usando marcadores fiduciais. Fonte: [37]

3.2 Marcadores Naturais

Para marcadores naturais não é utilizada uma biblioteca de rastreamento pronta. Neste trabalho é utilizado um reconhecedor de texturas juntamente com um algoritmo de calibração de câmera baseado em pares de pontos que conhecemos sua posição tanto no plano da imagem de textura quanto na imagem capturada pela câmera. Esses pares de pontos são fornecidos pelo algoritmo reconhecedor. Nas próximas seções serão brevemente descritos os processos de reconhecimento de texturas e de calibração de câmera utilizados nesse trabalho.

3.2.1 Reconhecedor de texturas

Atualmente, os dois algoritmos de reconhecimento de textura naturais mais usados são o SIFT [26] e o SURF [6]. No reconhecimento, o SIFT apresenta uma precisão superior à do SURF na maioria dos artigos publicados [3, 21]. Por outro lado, o SURF se mostrou cerca de quatro vezes mais rápido, segundo seu próprio autor e também segundo Bauer et al [3], onde é mostrado um gráfico do que podemos chamar de custo benefício, fig 3.3. O gráfico mostra o número de pontos detectados e pareados corretamente dividido pelo tempo gasto. Nesse gráfico pode-se perceber que o SURF é significativamente superior ao SIFT em todos os casos de teste apresentados. Os casos de teste são representados por variações de atributos da imagem (rotação, ruído branco,

PUC-Rio - Certificação Digital Nº 0721334/CA

escala, iluminação e perspectiva). Sendo assim, neste trabalho optamos por usar o SURF.

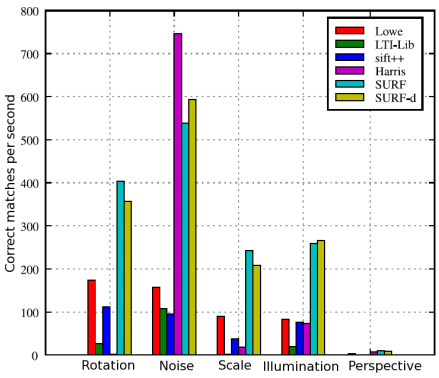

Figura 3.3: Número de pontos reconhecidos corretamente por segundo do SURF em diversas situações. Fonte: [3]

O algoritmo SURF[6, 4], acrônimo para *Speeded Up Robust Features*, é composto por um detector de pontos característicos e um descritor de pontos, que serão usados posteriormente para o reconhecimento das texturas. Os pontos característicos, comumente chamados na literatura de *features*, são pontos na imagem que podem ser localizados automaticamente de diferentes pontos de vista. Os algoritmos que fazem a detecção automática são os detectores de pontos característicos. Para cada ponto característico detectado o SURF usa o descritor de pontos para calcular um identificador do ponto baseado na textura ao redor dele. Em seguida vamos dar um breve resumo do detector, do descritor propostos no algoritmo SURF e do reconhecedor propriamente dito.

Detector de pontos característicos

O detector de pontos característicos foi chamado de *Fast-Hessian Detector*. Como o próprio nome diz, é um detector baseado na matriz Hessiana. A formulação dele é dada a seguir e, para facilitar a compreensão desse detector será seguida a nomenclatura e notação do artigo original. Dado um ponto $\mathbf{x} = (x, y)^T$ na imagem I, a matriz Hessiana $\mathcal{H}(\mathbf{x}, \sigma)$ em x na escala σ é definido como

$$\mathcal{H}(\mathbf{x}, \sigma) = \begin{bmatrix} L_{xx}(\mathbf{x}, \sigma) & L_{xy}(\mathbf{x}, \sigma) \\ L_{xy}(\mathbf{x}, \sigma) & L_{yy}(\mathbf{x}, \sigma) \end{bmatrix}, \tag{3-1}$$

onde $L_{xx}(\mathbf{x}, \sigma)$ é a derivada de segunda ordem da convolução gaussiana de parâmetro σ ($\frac{\partial^2}{\partial x^2} g(\sigma)$) no ponto $\mathbf{x}$, por similaridade temos $L_{yy}(\mathbf{x}, \sigma)$ e $L_{xy}(\mathbf{x}, \sigma)$

PUC-Rio - Certificação Digital Nº 0721334/CA

O nome do detector é hessiana rápida, na tradução literal, porque o algoritmo faz simplificações no cálculo da matriz hessiana. A convolução gaussiana de 9 x 9 na verdade é reduzida a filtros caixa, como desmonstrado na figura 3.4. Esses filtros caixa podem ser calculados rapidamente com o uso de imagem integral como foi definido por Viola e Jones[36]. Com isso também foi resolvido o problema das várias escalas, já que essa aproximação da hessiana pode ser calculada em diferentes escalas apenas aumentando ou diminuindo o tamanho do filtro caixa sem necessidade de efetivamente fazer as reduções da imagem original. Mais detalhes e imagens explicativas são apresentados no artigo original do SURF[6].

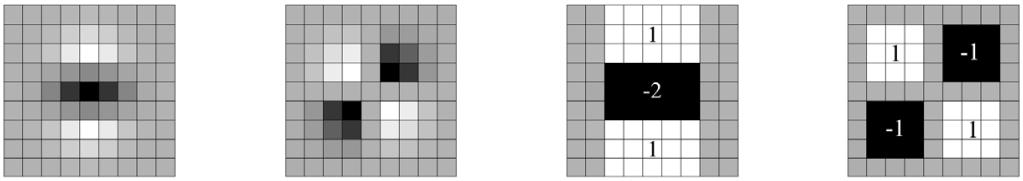

Figura 3.4: A esquerda derivada de segunda ordem da convolução gaussiana(L_{yy} e L_{xy})) e a direita as correspondentes aproximações usando filtros caixa(D_{yy} e D_{xy})). Fonte: [6]

Após calcular a hessiana para toda a imagem em diferentes escalas, em geral a escala original e mais 3 reduções, é calculado o determinante da hessiana rápida em cada ponto pela fórmula 3-2

$$det(\mathcal{H}_{rápida})(\mathbf{x},\sigma) = D_{xx}(\mathbf{x},\sigma)D_{yy}(\mathbf{x},\sigma) - (0.9D_{xy}(\mathbf{x},\sigma))^2, \qquad (3\text{-}2)$$

onde $D_{xx}(\mathbf{x},\sigma)$ corresponde ao filtro caixa correspondente a $L_{xx}(\mathbf{x},\sigma)$, por similaridade temos $D_{yy}(\mathbf{x},\sigma)$ e $D_{xy}(\mathbf{x},\sigma)$

Finalmente, é feita uma supressão de não máximos desses determinantes na vizinhança 3 x 3 x 3 para encontrar os melhores pontos característicos.

Descritor de pontos

O descritor de pontos do SURF é composto principalmente por: $\mathbf{x}$,σ,um vetor de orientação em $\mathbb{R}^2$, e uma matriz de 64 posições na forma 4 x 4 x 4 com informações sobre os vizinhos do ponto. Os dois primeiros atributos já podem ser calculados na fase de detecção. A orientação é o vetor calculado a partir da resposta da Haar wavelet na direção x e na direção y. Da mesma forma que a gaussiana, as Haar wavelets podem ser computadas rapidamente por filtros

PUC-Rio - Certificação Digital Nº 0721334/CA

caixa. O cálculo desse vetor torna o descritor invariante à rotação, ou seja, é possível rotacionar a imagem que o ponto descrito continua sendo identificado. Por fim, a matriz com as informações sobre os vizinhos é calculada. Essa matriz é calculada a partir do retalho de tamanho 20 x 20 pixels centrado no ponto **x** na imagem I reduzida pelo fator σ e orientado pelo vetor de orientação. Esse retalho é dividido em 4 x 4 sub-retalhos cada um com 5x5 pixels como mostra a figura 3.5. Cada um dos 5 x 5 pixels tem computada sua resposta da Haar wavelet na direção x' e y', que são as direções x e y corrigidas pelo vetor orientação e vamos chamá-las de dx e dy. O filtro caixa para a resposta da Haar wavelet é do tamanho 2 x 2. Para cada sub-retalho é feito o somatório em cada um dos seus 25 pixels de dx e dy e de seus valores normalizados $|dx|$ e $|dy|$. Com isso cada um dos sub-retalhos passa a ter como atributos $\sum dx, \sum |dx|, \sum dy, \sum |dy|$ que são a terceira dimensão de tamanho 4 da matriz de 64 posições que descreve a vizinhança do ponto. A invariância a mudança uniforme de iluminação é obtida graças ao uso do Haar wavelet que mede a diferença de iluminação entre os vizinhos ao invés do valor propriamente. A invariância a mudança de contraste pode ser obtida se normalizarmos a matriz de 64 posições para ter soma dos elementos unitária. Para saber se dois pontos são parecidos basta calcular a distância entre as duas matrizes. Essa distância é dada considerando cada matriz como um ponto em $\mathbb{R}^{64}$ e calculando a distância euclidiana no espaço do descritor entre os pontos, ou seja, $dist = \sqrt{\sum_{i=0}^{64}(A_i - B_i)^2}$

Reconhecimento de texturas

O reconhecimento de texturas do SURF, diferente do reconhecimento de marcadores fiduciais, precisa ser feito por comparação das possíveis texturas. Portanto, é necessário em pré-processamento fazer uma base de dados das texturas que serão procuradas. Cada textura é processada pelo detector de pontos característicos. Em seguida, é calculado o descritor de cada um desses pontos. O ponto característico com um descritor associado recebe o nome de pontos-chave, ou na literatura como *keypoints*. O processo de identificação da textura na imagem capturada pela câmera durante o processamento é igual. Os pontos característicos são detectados nela e em seguida descritos. Para procurar se existe alguma textura pertencente ao nosso banco de dados, precisamos comparar cada ponto chave da imagem da câmera com todos os pontos chave de todas as texturas, uma por uma. Em cada uma computamos os pares de pontos da imagem da câmera e da textura procurada que tenham a distância

PUC-Rio - Certificação Digital Nº 0721334/CA

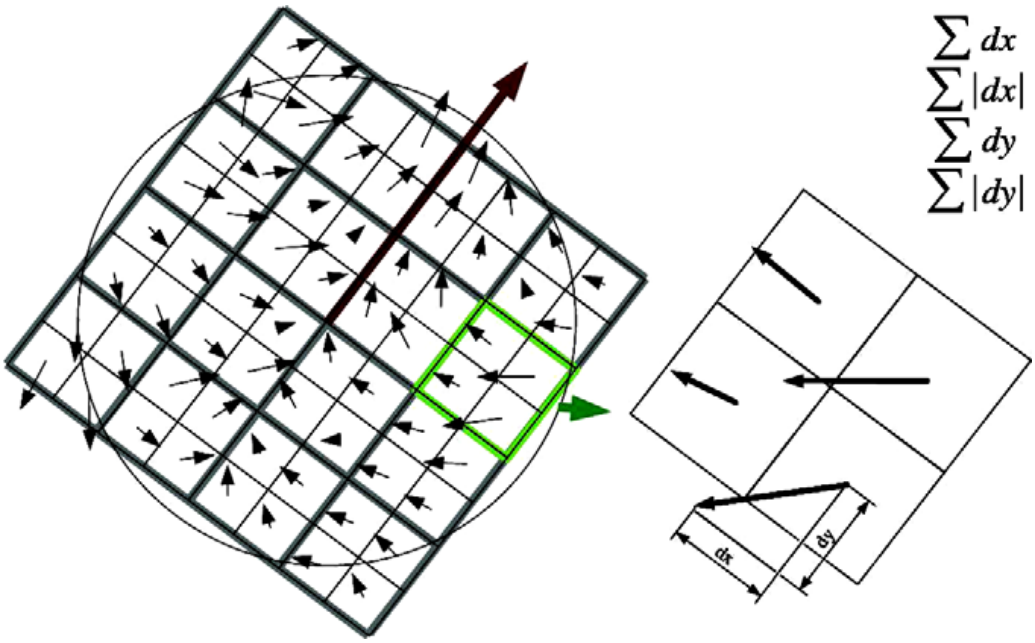

Figura 3.5: Para construir o descritor, o retalho orientado é subdividido numa grade de 4x4 sub-retalhos. Dentro de cada um as respostas ao wavelet foram calculadas de 5 x 5 amostras(por questões ilustrativas, só tem 2x2). Fonte: [6]

PUC-Rio - Certificação Digital Nº 0721334/CA

euclidiana no espaço do descritor razoavelmente pequena. O número de pares de pontos com cada textura i da base de dados é N_i. Normalmente para saber qual textura está aparecendo na imagem da câmera, considerando que só possa ter uma de cada vez, é a textura que tem N_i maior, mas isso pode nos levar para falsas detecções, pois o pareamento dos pontos foi considerando um valor maior que zero para a distância entre os descritores, pois é impossível esperar distância zero com o descritor sendo um vetor de números reais, levando a um inescapável erro numérico por menor que seja, além de que a imagem da câmera é a representação discreta da cena vista pela câmera, que é um espaço contínuo. Sendo assim utilizamos o critério S_r que balanceia o N_i com a distância entre os pontos-chave, esse método foi proposto em [5]. O critério S_r é definido como

$$S_r = \underset{i}{\operatorname{argmax}} \left(\frac{N_i}{\sqrt{\sum_{j=i}^{N_i} dist_{ij}^2}} \right), \tag{3-3}$$

onde i representa a i-ésima textura do banco de dados e j o j-ésimo keypoint da textura i. Para aceitar como válido o reconhecimento, S_r precisa ser maior que 0.8 vezes do que o segundo maior S_r e $N_i maiorque 8$.

3.2.2 Calibração de câmera

Calibração de câmera é o procedimento de descobrir os parâmetros intrínsecos e extrínsecos de uma câmera. Os parâmetros intrínsecos são os inerentes ao conjunto sensor de captura (CCD) e lente. Eles só variam em três situações: caso a câmera mude, a lente mude, ou a distância focal mude. Os parâmetros são o centro focal no plano da imagem, a distância focal, o tamanho da imagem e os coeficientes de distorção radial. Os parâmetros extrínsecos são a rotação e a translação que definem a posição da câmera em relação ao sistema de coordenadas de referência.

Os parâmetros intrínsecos não foram abordados nesse trabalho. Para encontrá-los, usamos um método manual auxiliado por um kit de ferramenta(toolkit) de calibração de parâmetros intrínsecos no MATLAB, o *Camera Calibration Toolbox for Matlab*[7] feito por Bouguet. O algoritmo utilizado por ele não influenciou os detalhes de projeto do nosso algoritmo. O mais importante é que ele capaz de calcular e verificar a qualidade dos parâmetros intrínsecos com bastante precisão a partir dos dados de entrada do operador, sendo assim supomos que o operador ajustou os dados de entrada até que a precisão dos parâmetros intrínsecos tenham ficado num limiar aceitável para o tipo de aplicação. Vale lembrar que os parâmetros intrínsecos calculados são os mesmos para todos os algoritmos de calibração desse trabalho, ou seja, o algoritmo explicado nesta seção, o algoritmo de calibração interno do ArToolKitPlus e o algoritmo de SLAM.

Para calcular os parâmetros extrínsecos, é utilizado o método descrito no artigo de Zhang[40] ligeiramente modificado. Esse método é constituído de dois passos, o cálculo da homografia e em seguida o cálculo da rotação e da translação da câmera levando em conta os parâmetros intrínsecos e a homografia calculada. Homografia é uma matriz 3x3 que representa a transformação de perspectiva entre dois planos. A homografia é calcula pelo algoritmo *Direct Linear Transformation (DLT)* [20]. Para usar o DLT é necessário que todos os pares de ponto sejam corretos, mas o SURF não garante absoluta certeza nos pares retornados, então usamos a técnica RANSAC (Random Sample Consensus)[20] minimizando o erro de reprojeção para tentar eliminar os pares incorretos. O uso do DLT e do RANSAC são as modificações feitas no método original de Zhang que usa outro tipo de cálculo de homografia. Esse algoritmo consegue a homografia H a partir de um conjunto de pares de pontos($\mathbf{m} \leftrightarrow \mathbf{m}'$) suficientemente grande. Um par de pontos é constituído por $\mathbf{m} = (u\ v\ 1)^T$,o ponto na imagem, e $\mathbf{m}' = (x\ y\ 1)^T$,o ponto no modelo. O modelo é a textura armazenada previamente, seja fruto do escaneamento ou

PUC-Rio - Certificação Digital Nº 0721334/CA

mesmo uma foto, já a imagem é a imagem da câmera no momento que se quer saber sua posição 3D. A relação entre eles é descrita como $\mathbf{m} = H\mathbf{m}'$. A técnica RANSAC propõe que use apenas um subgrupo escolhido aleatoriamente dos pares de ponto para o cálculo da homografia e com essa estimativa projete todos os $\mathbf{m}'$ e calcule a distância para $\mathbf{m}$,ou seja,$\xi_i = ||H\mathbf{m_i'} - \mathbf{m_i}||$, o operador $||.||$ é a norma. Caso o $\frac{1}{n}\sum_i \xi_i$, onde n e o número de pares, seja abaixo de um limiar a homografia é considera certa, caso contrário o algoritmo tenta mais algumas vezes o mesmo processo, se todas resultarem acima do limiar ele descarta o reconhecimento feito pelo SURF e o processamento não continua. Com a homografia calculada e os parâmetros intrínsecos conhecidos podemos passar para o cálculo dos parâmetros extrínsecos. Em Zhang[40] é feita a demonstração matemática do método. O apêndice B apresenta as fórmulas necessárias para encontrar os parâmetros extrínsecos já diretamente relacionados aos dados de entrada, ou seja, a homografia e os parâmetros intrínsecos. Para que os parâmetros extrínsecos sejam dados em coordenadas métricas, basta que os pontos no modelo($\mathbf{m}'$) também estejam em coordenadas métricas e os pontos da imagem($\mathbf{m}$) em número de pixels.

3.3 Mapeamento do ambiente

O mapeamento do ambiente é importante para o nosso algoritmo, pois é ele que permite criarmos um sistema de coordenada comum a todos os marcadores. Como foi explicado nas seções acima os rastreadores de marcadores só informam a posição da câmera relativa a um marcador específico. Então, para descobrirmos a posição global da câmera é necessário conhecer também a posição global do marcador e concatenar as duas informações. A outra função do mapeamento é selecionar os pontos que serão acompanhados pelo SLAM quando não estiver sendo utilizado o rastreador de marcador. Para isso precisamos selecionar pontos que sejam facilmente identificados pelo detector de característica interno do algoritmo de SLAM utilizado, no nosso caso o detector é o *Shi e Tomasi*[33]. Além de selecionar os pontos é necessário conhecer a posição deles no sistema de coordenada global, para que o SLAM também calcule sua posição em coordenadas globais. Por fim, o mapeamento precisa calcular o fator métrico que faz com que os rastreadores calculem a posição relativa da câmera ao marcador na escala métrica.

Nesta seção, primeiramente descrevemos o equipamento utilizado para realizar o mapeamento, em seguida explicamos como fazer as medições nos marcadores fiduciais e naturais e como concatenar a posição relativa ao marcador e com a posição global do marcador.

PUC-Rio - Certificação Digital Nº 0721334/CA

3.3.1
Equipamento de medição

O mapeamento dos marcadores foi feito usando um equipamento de topografia chamado *estação total*(figura 3.6). Esse equipamento é capaz de medir coordenadas 3D com bastante precisão. Ele utiliza três sensores para fazer as medições: um sensor de rotação que mede as rotações do equipamento sobre o eixo colinear com a força da gravidade(pan), outro que mede a rotação vertical (tilt) e, por fim, um sensor laser para medir a distância do ponto que se deseja até o equipamento. Os sensores de rotação tem precisão de 6" de grau e o de distâncias de 3 a 5 mm para pequenas distâncias. Com os dados provenientes desses três sensores e alguns pontos de referência conhecidos o equipamento calcula internamente as coordenadas 3D. Para mapas maiores que necessitam de medidas de várias posições da estação pode ser usado um algoritmo de mínimos quadrados para diminuir o erro de medição e o erro acumulado proveniente da movimentação.

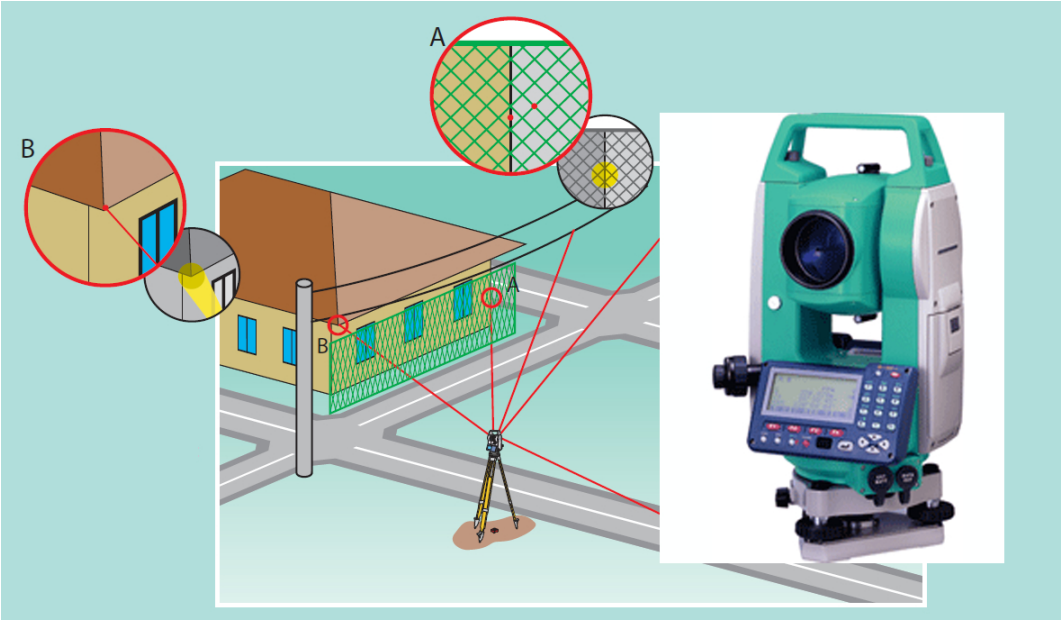

Figura 3.6: Estação Total. Fonte: Sokkia Corp.

3.3.2
Mapeamento dos marcadores fiduciais

Nos marcadores fiduciais medimos com a estação as coordenadas dos quatro vértices($\mathbf{p_1}$ $\mathbf{p_2}$ $\mathbf{p_3}$ $\mathbf{p_4}$) em $\mathbb{R}^3$, do quadrado preto que envolve o marcador (ver figura 3.7). Assim, pela média desses pontos temos o ponto

PUC-Rio - Certificação Digital Nº 0721334/CA

médio dos vértices que é o centro($\mathbf{p_c}$) do sistema de coordenada local do marcador. A normal pode ser calculada pelo produto vetorial. Assim temos

$$\mathbf{p_c} = \frac{1}{4}(\mathbf{p_1} + \mathbf{p_2} + \mathbf{p_3} + \mathbf{p_4}) \tag{3-4}$$

$$\vec{n} = (\mathbf{p_3} - \mathbf{p_4}) \times (\mathbf{p_1} - \mathbf{p_4}) \tag{3-5}$$

Para os cálculos necessários para transformar os pontos calculados relativamente ao marcador para o sistema de coordenada global vamos escrever essa transformação na forma de uma matriz 4x4 de transformação de corpo rígido, vamos chamar de Q. Vamos chamar de $\mathbf{p^M} = (x^M\ y^M\ z^M\ 1)$ o ponto no sistema do marcador e $\mathbf{p^G} = (x^G\ y^G\ z^G\ 1)$ no sistema de coordenada global. A matriz Q satisfaz a relação

$$\mathbf{p^M} = Q_{MG}\mathbf{p^G}, \tag{3-6}$$

onde Q_{MG}é a transformação de corpo rígido que leva do sistema de coordenada global para o do marcador. Como sabemos que uma transformação de corpo rígido é inversível, temos que

$$\mathbf{p^G} = Q_{MG}^{-1}\mathbf{p^M} = Q_{GM}\mathbf{p^M}, \tag{3-7}$$

onde Q_{GM} é a transformação de corpo rígido que leva do sistema de coordenada do marcador para o global. No apêndice C é mostrado como montar essa matriz a partir de três vetores. Esse apêndice mostra como calcular a matriz Q que transforma de um sistema de coordenada qualquer U, desde que respeite a regra da mão direita, para o sistema de coordenada da base canônica. O método apresentado no apêndice C tem como dados de entrada dois vetores que representam o eixo Z e o eixo Y do sistema de coordenada S na base canônica, além da posição do centro de coordenadas de U no sistema de coordenadas canônica.

Sendo assim para calcularmos Q_{MG} basta termos $\mathbf{p_c}$, $\vec{n}$ e o vetor para o eixo Y é $ei\vec{x}o_y = (\mathbf{p_1} - \mathbf{p_4})$.

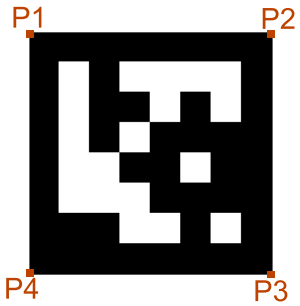

Figura 3.7: Configuração dos pontos medidos no marcador fiducial

PUC-Rio - Certificação Digital Nº 0721334/CA

A seleção dos pontos a serem acompanhados pelo SLAM é fácil nesse tipo de marcador, pois os vértices do retângulo são perfeitamente localizados. Outra questão facilitadora é que nenhum cálculo é necessário para encontrar os pontos em coordenadas globais, pois eles são exatamente os pontos de medição $\mathbf{p_1}$ $\mathbf{p_2}$ $\mathbf{p_3}$ $\mathbf{p_4}$. Além disso, o fator métrico é apenas a medição do tamanho do lado do retângulo que pode ser tanto calculado por $||\mathbf{p_1} - \mathbf{p_2}||$ quanto fornecido pelo usuário, já que o marcador fiducial foi impresso por ele.

3.3.3 Mapeamento dos marcadores naturais

A medição dos marcadores naturais é um pouco mais complicada, pois os pontos medidos pela estação total precisam ser marcados manualmente na imagem para fazer a correspondência, já que a textura natural não tem artefatos predefinidos como os fiduciais. Para fazer a medição da posição em $\mathbb{R}^3$ do marcador é necessário escolher 3 pontos na textura,$\mathbf{b_1} = (u, v, 1)^T$,$\mathbf{b_2}$ e $\mathbf{b_3}$. Depois mede-se suas posições correspondentes no espaço 3D $\mathbf{b'_1} = (x, y, z, 1)$,$\mathbf{b'_2}$,$\mathbf{b'_3}$, calculadas com a estação total, de preferência o mais separados possíveis e não colineares para que seja possível determinar um plano. O ideal é que sejam na forma de L, igual ao $\mathbf{p_1}$,$\mathbf{p_3}$,$\mathbf{p_4}$ na figura 3.7, mas muitas vezes isso não é possível como mostrado na figura 3.8. O sistema de coordenadas calculado será sempre considerando que $\mathbf{b_1}$ está mais próximo ao canto inferior esquerdo, $\mathbf{b_2}$ está mais próximo ao canto inferior direito e $\mathbf{b_3}$ está mais próximo ao canto superior esquerdo.

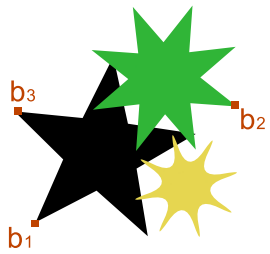

Figura 3.8: Configuração dos pontos medidos no marcador natural

O centro do sistema de coordenadas do marcador natural vai ser o ponto médio dos 3 pontos medidos igual ao dos marcadores fiduciais, mas isso não garante mais que será o centro da textura já que ele vai depender de $\mathbf{b_1}$,$\mathbf{b_2}$ e $\mathbf{b_3}$. As fórmulas para cálculo do centro e da normal no sistema do modelo e da imagem são

$$\mathbf{b_c} = \frac{1}{3}(\mathbf{b_1} + \mathbf{b_2} + \mathbf{b_4}) \tag{3-8}$$

PUC-Rio - Certificação Digital Nº 0721334/CA

$$\mathbf{b'_c} = \frac{1}{3}(\mathbf{b'_1} + \mathbf{b'_2} + \mathbf{b'_4}) \tag{3-9}$$

$$\vec{n} = (\mathbf{b_2} - \mathbf{b_1}) \times (\mathbf{b_3} - \mathbf{b_1}) \tag{3-10}$$

$$\vec{n'} = (\mathbf{b'_2} - \mathbf{b'_1}) \times (\mathbf{b'_3} - \mathbf{b'_1}) \tag{3-11}$$

Sendo assim para calcularmos Q_{MG} basta termos $\mathbf{b'_c}$, $\vec{n'}$ e o vetor para o eixo Y é $ei\vec{x}o_y = (\mathbf{b'_3} - \mathbf{b'_1})$. Os dados $\mathbf{b_c}$, $\vec{n}$ serão usados a seguir.

O rastreador de marcadores naturais detecta os pontos-chave do modelo($\mathbf{m'} = (x\ y\ 1)^T$) em relação a uma das extremidades da imagem, no nosso caso temos o canto inferior esquerdo crescendo para o cima, e na escala em pixels. Para que o rastreador calcule a posição da câmera na mesma escala e considerando o mesmo centro é necessário corrigir a posição dos pontos da seguinte forma

$$\breve{\mathbf{m}} = s(\mathbf{m'} - \mathbf{b_c}), \tag{3-12}$$

onde s é o fator de escala. Sabendo que a relação entre Q_{MG} é uma transformação de corpo rígido, então ela preserva os tamanhos relativos e as distâncias, logo $s = ||\mathbf{b_2} - \mathbf{b_1}||/||\mathbf{b'_2} - \mathbf{b'_1}||$.

Os pontos característicos que serão acompanhados pelo SLAM precisam ser facilmente identificados pelo seu detector, mas diferentemente dos marcadores fiduciais, os marcadores naturais não contam com o retângulo preto ao redor, então é necessário realizar um processamento de imagem para encontrá-los. O processamento é, resumidamente, executar o detector na textura e escolher dentre os pontos detectados os mais afastados no formato mostrado na figura 3.8.

3.4 Algoritmo de rastreamento usando SLAM

O algoritmo proposto nesse trabalho, teoricamente, não está atrelado a nenhum algoritmo de SLAM específico. Um algoritmo de SLAM sempre depende de uma posição a partir de onde constrói um mapa para manter sua posição conhecida. Esse é a única assertiva que assumimos para propor o conceito de Local SLAM. No entanto, é preciso existir uma forma de mapear as características do marcador com o mapa do algoritmo de SLAM. Para fim de demonstração de conceito, vamos usar o algoritmo de Davison[14]. Ele foi escolhido por algumas razões: o próprio autor do algoritmo disponibiliza uma implementação para o algoritmo; a biblioteca é bem documentada; a implementação funciona em tempo real; tem se mostrado funcional para fins de demonstração de conceitos em diversos artigos científicos. O funcionamento

PUC-Rio - Certificação Digital Nº 0721334/CA

desse algoritmo já foi explicado com maiores detalhes na seção de trabalhos relacionados. A seguir será dado apenas um breve resumo a fim de facilitar o entendimento.

Algoritmo 1 algoritmo de atualização do Algoritmo de Davison

1: **loop**
2: captura novo quadro da câmera
3: preditar posição da câmera no quadro usando filtro de Kalman Estendido(EKF).
4: dentro do conjunto de pontos conhecidos do mapa, selecionar os que estão visíveis.
5: procurar os retalhos de cada ponto na região onde o ponto deveria estar baseado na posição preditada pelo EKF
6: Aplicar o corretor do EKF para corrigir a posição da câmera usando os pontos localizados.
7: Processo de adicionar novos pontos ao mapa.
8: **end loop**

Nesse algoritmo, não existe nenhum tipo de inicialização. Ele considera que a câmera está sempre à mesma distância de um retângulo preto, como mostrado na figura 3.9 e o mapa contem apenas os quatro vértices do retângulo. Então para ser possível fazer a inicialização com outros tipos de marcadores e de diferentes posições da câmera, o algoritmo foi modificado. A modificação foi a inclusão de um algoritmo substitutivo ao algoritmo original 1 para quando for requisitado uma nova inicialização. Quando não for necessária a inicialização, o algoritmo 1 é executado. O algoritmo de inicialização 2 tem como parâmetros de entrada os dados que são fixos no algoritmo tradicional e a imagem sobre o qual os parâmetros foram calculados. Eles são a posição da câmera em coordenadas globais, a posição dos retalhos do marcador em coordenadas globais e a posição deles na imagem da câmera atual.

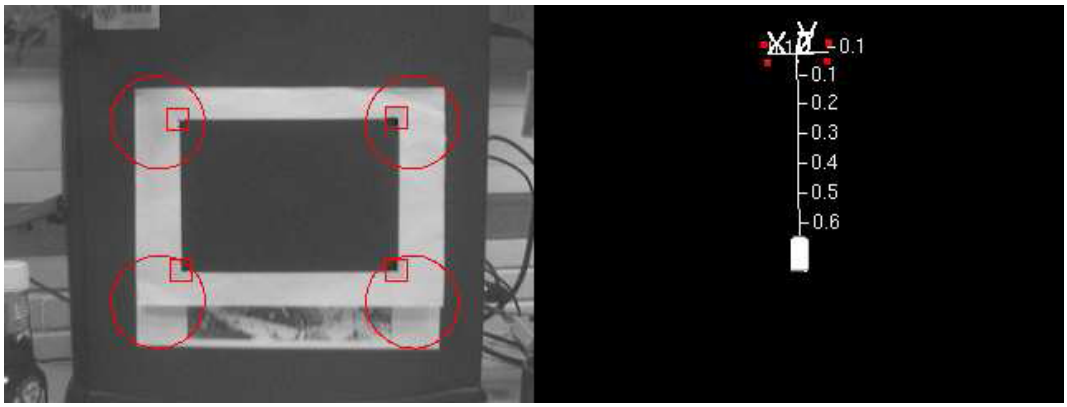

Figura 3.9: Configuração padrão de inicialização do algoritmo de Davison

PUC-Rio - Certificação Digital Nº 0721334/CA

Algoritmo 2 Processo de inicialização para o Algoritmo de Davison

1: adiciona os retalhos conhecidos ao mapa.
2: posiciona a câmera usando a posição dada como entrada.
3: preditar posição da câmera no quadro usando filtro de Kalman Estendido(EKF).
4: Associa os pontos 3D conhecidos do marcador com suas posições no plano da imagem.
5: Aplicar o corretor do EKF para corrigir a posição da câmera usando os pontos localizados.
6: Sai da inicialização e volta para o passo 1 do processo normal.

Na próxima seção será explicado quando usar o algoritmo 1 ou o algoritmo 2, ou seja, quando é necessário fazer a inicialização.

3.5 Heurísticas de reinicialização

Definir quando reinicializar o algoritmo de SLAM não é uma tarefa óbvia, pois queremos reinicializar quando dispomos de uma posição de maior qualidade dos rastreadores de marcadores, ao mesmo tempo em que não queremos piorar o rastreamento que muitas vezes pode ter uma posição da câmera mais precisa. O rastreador de marcadores fiduciais é bem confiável, mas o rastreador de marcadores naturais pode dar resultados errados do posicionamento da câmera, caso o algoritmo de RANSAC não consiga remover os pontos correspondentes errados. Além do mais, os dois tipos de marcadores se vistos de muito longe também não fornecem uma posição boa para a câmera por conta da pouca área na imagem que eles ocupam. Sendo assim, resolvemos propor duas Heurísticas de reinicialização para serem testadas.

3.5.1 Heurística 1 - Quando possível

Nessa heurística, toda vez que os rastreadores encontrarem um marcador o algoritmo de SLAM passa pela inicialização. Essa heurística é baseada na suposição que os rastreadores de marcador são sempre melhores que a posição dada pelo SLAM. O algoritmo de SLAM seria apenas um interpolador de posições entre marcadores. No entanto, a reinicialização não pode impedir que o SLAM mapei um número mínimo de pontos antes do marcador sair da imagem. Por conta disso, estabelecemos duas regras. A primeira regra é o número mínimo de pixel que o marcador precisa ocupar na imagem para conseguirmos uma reinicialização de qualidade. O segundo é que o marcador precisa estar a mais de um certo numero de pixel da borda da imagem, pois

PUC-Rio - Certificação Digital Nº 0721334/CA

nessa posição ele está prestes a sair da imagem então precisamos criar um mapa ao redor antes que ele saia da imagem e a calibração seja impossível.

3.5.2 Heurística 2 - Apenas com marcador no centro

Nessa heurística, quando os rastreadores encontrarem um marcador no centro da imagem, de frente para câmera e com uma área na imagem mínima, o algoritmo de SLAM passa pela inicialização. Essa heurística é baseada na ideia que quando a câmera está de frente, centralizada e o marcador ocupando uma boa área da imagem os rastreadores funcionam com mais precisão.

3.6 Processamento geral

Considerando que hoje em dia a maioria dos processadores tem mais de um núcleo, o algoritmo foi desenhado para funcionar utilizando dois processos em paralelo. A primeira parte do algoritmo tem a tarefa de executar o processamento do Visual SLAM. Como esse processo tem coerência temporal, é vital que não tenha gargalos. O segundo processo ficou responsável por detectar e rastrear os marcadores. Esse processo foi chamado de relocalizador, já que ele tenta calcular constantemente os dados para reposicionar o algoritmo de SLAM. Esse processo não tem a obrigação de funcionar em tempo real já que sempre que o rastreador de marcadores naturais for executado o tempo real será perdido. Cada um dos processos vai ser mais detalhado abaixo.

3.6.1 Relocalizador

A programação paralela pode trazer uma série de problemas de concorrência. Então decidimos por manter esse processo com um mínimo de acoplamento com o processo principal. Ele tem um espaço de memória protegido por semáforos para entrada de dados e outro, também protegido, para saída de dados. No espaço de entrada fica armazenada a próxima imagem a ser processada, a última posição da câmera calculada com sucesso e a marca temporal (timestamp) na qual a imagem foi capturada. No espaço de saída é armazenado o resultado do último marcador encontrado com sucesso. O resultado é composto pela posição da câmera em coordenadas globais (coordenada da origem e orientação da câmera), pelos retalhos juntamente com suas posições tanto em coordenadas globais quanto na imagem e pela marca de tempo da imagem de onde esses dados foram calculados.

PUC-Rio - Certificação Digital Nº 0721334/CA

O processamento principal do Relocalizador é basicamente: caso haja uma imagem nova disponível no espaço de entrada, procurar um marcador fiducial, caso não encontre, procura um marcador natural. Se alguma das buscas resultar em sucesso, corrige a posição da câmera para coordenadas globais e a copia para o espaço de saída juntamente com os dados adicionais. O marcador fiducial é procurado primeiro, pois ele funciona em tempo real. Sendo assim, caso ele tenha sucesso, é possível manter o relocalizador funcionando em tempo real. Isso não seria possível se o rastreador de marcadores naturais fosse feito primeiro, já que ele não consegue atingir tempo real.

3.6.2 Processo de Atualização do Visual SLAM

A tarefa desse processo é executar a atualização do SLAM e decidir quando reinicializá-lo. Esse processo apenas testa se a heurística de reinicialização é verdadeira ao mesmo tempo que existe a detecção de um marcador no Relocalizador feita há pouco tempo atrás. Se essas duas condições são satisfeitas o algoritmo de inicialização é executado, caso contrário a atualização original é feita no algoritmo de SLAM.

PUC-Rio - Certificação Digital Nº 0721334/CA