6
Algoritmos de Exploração e Navegação Propostos

O presente capítulo apresenta o algoritmo de exploração proposto e detalha todas suas componentes. As seções deste capítulo são:

- 6.1 - Algoritmo de Exploração;
- 6.2 - Componente 1 - Criando Imagens Panorâmicas Automaticamente;
- 6.3 - Componente 2 - Segmentação da Imagem por *Quadtree* Baseada em Entropia;
- 6.4 - *Visual Tracking*: Acompanhamento de *Features* e Navegação;
- 6.5 - Componente 3 - Navegação para Nó Desconhecido;
- 6.6 - Componente 4 - Navegação para Nó Conhecido;
- 6.7 - Componente 5 - Refinando o modelo criado através de algoritmos genéticos;

PUC-Rio - Certificação Digital Nº 0410268/CA

6.1.
Algoritmo de Exploração

Nesta seção é apresentado um algoritmo para a modelagem e exploração de um ambiente *indoor* estático. Este algoritmo trabalha com diferentes componentes que serão abordadas em detalhes nas seções que se seguem neste capítulo. O algoritmo de modelagem proposto se diferencia da grande maioria dos algoritmos encontrados por não fazer a modelagem através de um mapeamento usual, e sim através da definição e exploração de posições no ambiente, escolhidas através da análise visual do mesmo.

O modelo criado pelo robô é uma árvore de nós. Cada adjacência carrega a informação de distância entre dois nós e a diferença angular entre eles. Cada nó guarda a imagem panorâmica criada naquele ponto do espaço. Esta modelagem possibilitaria uma futura navegação baseada em busca de imagens, seja de objetos, lugares, ou outros.

A Figura 6-1 ilustra um possível ambiente que o robô teria explorado. Os círculos em preto representam os locais que o robô explorou, ou seja, os nós

criados. As linhas entre os círculos representam os caminhos que o robô navegou entre um nó e outro. Também são apresentados os símbolos θ e I, que representam respectivamente os ângulos entre os nós e as imagens panorâmicas. A representação deste modelo topológico é uma árvore tal como a apresentada na Figura 6-2, que corresponde à exploração da Figura 6-1.

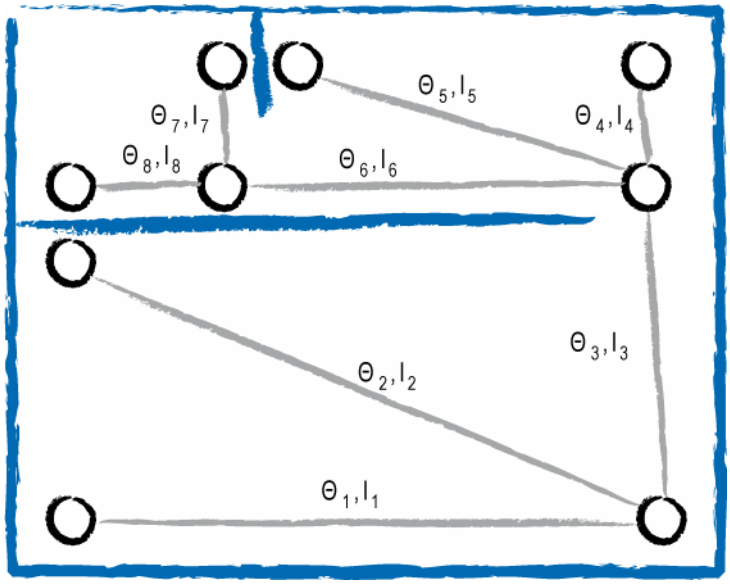

Figura 6-1: Mapa de um ambiente navegado por nós

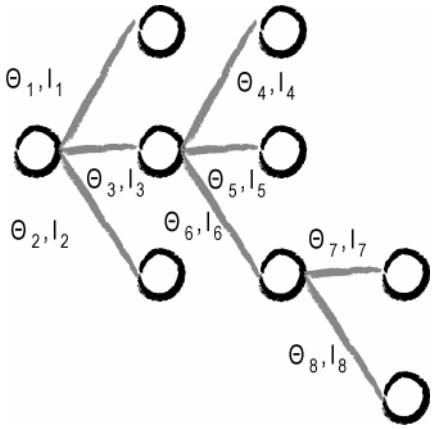

Figura 6-2: Árvore de nós

O robô deve navegar através de diferentes locais do ambiente de modo a construir um modelo topográfico que é esta coleção de nós em um grafo. Cada local explorado pelo robô é marcado como um nó no mapa criado. Para cada nó

PUC-Rio - Certificação Digital Nº 0410268/CA

explorado, uma série de procedimentos são realizados. Uma visão geral do algoritmo pode ser vista na Figura 6-3.

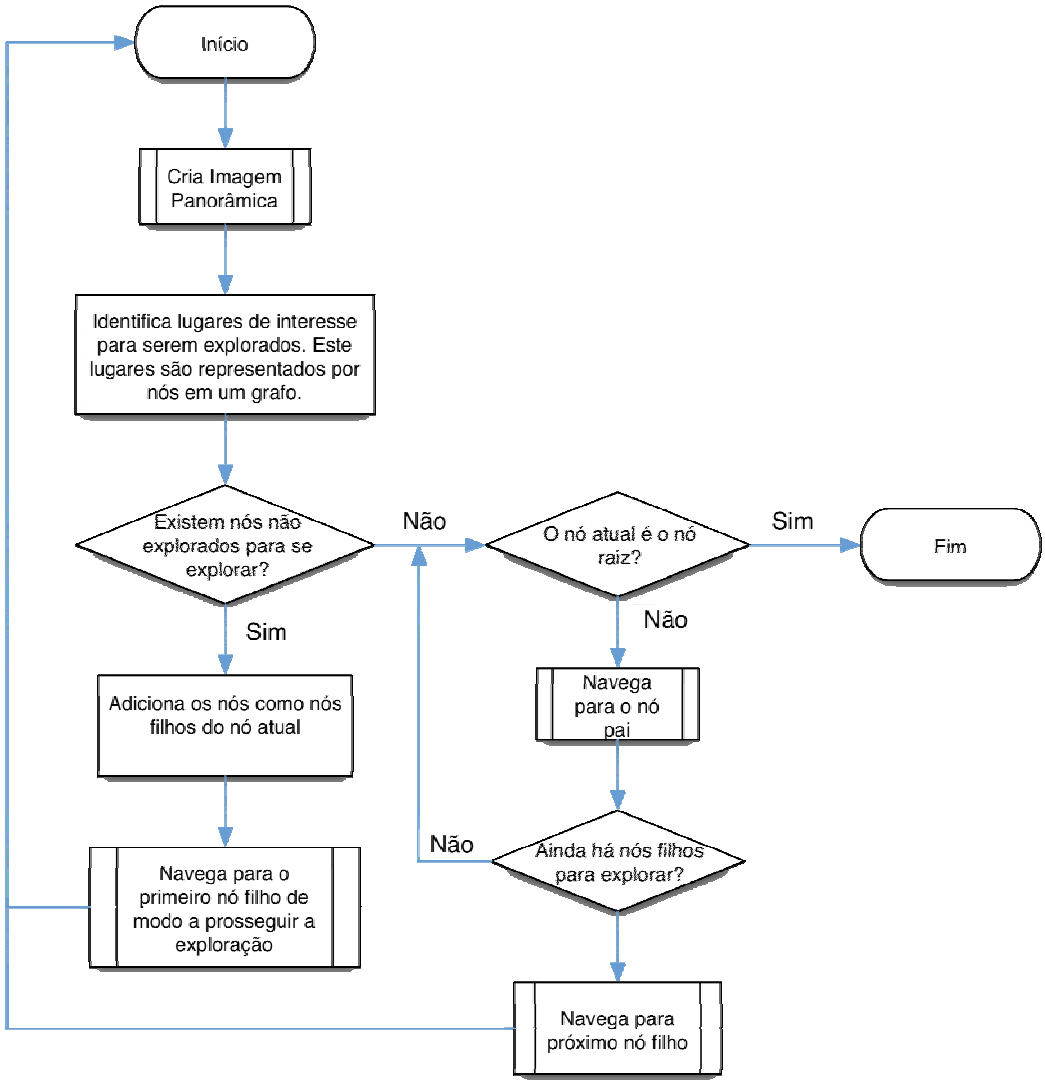

Figura 6-3: Visão Geral do Algoritmo

O algoritmo pode ser entendido como se segue:

1. A primeira posição do robô é definida como o nó raiz da árvore a ser criada;
2. Para o nó atual, registra-se uma imagem panorâmica do ambiente;
3. A partir da imagem panorâmica, novos lugares para serem explorados são escolhidos. Cada um destes lugares passa a ser representado como um nó ainda não explorado no modelo criado;

PUC-Rio - Certificação Digital Nº 0410268/CA

4. Verifica-se a existência de nós a serem explorados a partir do nó atual. Caso existam, navega-se para o primeiro destes nós de modo a prosseguir a exploração. Deste novo nó, a exploração segue da etapa 2;
5. Caso não existam nós para serem explorados a partir do nó atual, navega-se para o nó pai do nó atual;
6. Após chegar no nó pai, verifica-se se este nó é o nó raiz. Caso seja, o procedimento foi terminado. Caso não seja, o procedimento segue da etapa 7;
7. Uma nova verificação é feita de modo a se constatar se existem nós não explorados a partir do nó atual. Caso não existam, navega-se para o nó pai do nó atual e se prossegue a partir da etapa 5. Caso existam nós filhos para serem explorados, navega-se para o primeiro destes nós e segue a exploração da etapa 2;

A Figura 6-4 mostra um exemplo do algoritmo que ilustra a construção do modelo interno ao longo de uma exploração. No primeiro momento o robô está no nó raiz e encontra 3 novos nós para serem explorados. O robô se dirige ao primeiro destes nós e verifica que não há novos nós para serem explorados a partir deste, voltando então para o nó pai. Do nó raiz o robô vai para o segundo nó filho e neste encontra mais 3 nós para serem explorados. Se dirige ao primeiro destes nós e encontra mais 2 nós a serem explorados. O exemplo termina aqui mas em um caso real, o robô só terminaria o algoritmo após explorar todos os nós definidos.

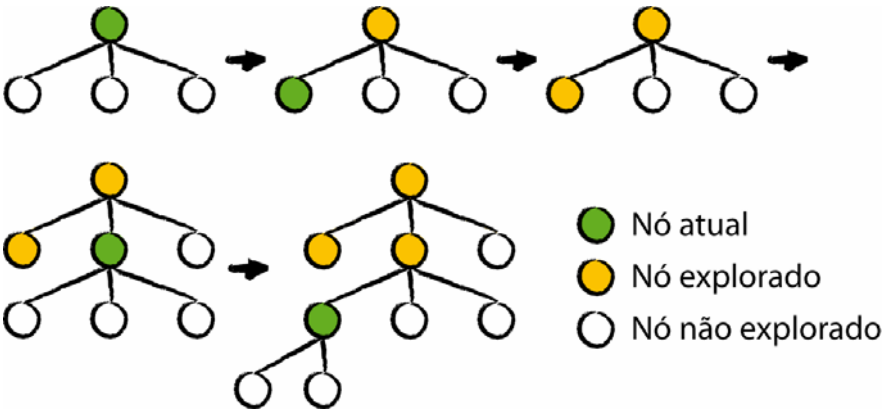

Figura 6-4: Exemplo de como é feita a exploração

PUC-Rio - Certificação Digital Nº 0410268/CA

O algoritmo consiste do uso de quatro componentes primários:

- Componente 1 - Geração de imagens panorâmicas: Para cada posição explorada, o robô captura uma série de imagens ao redor de si. Com estas imagens cria-se uma imagem panorâmica do ambiente para aquela posição. Esta componente é apresentada na seção 6.2;
- Componente 2 - Identificação de lugares de interesse para futura exploração: Através das imagens panorâmicas, são definidos os locais que o robô deverá explorar. A componente 2 será vista na seção 6.3;
- Componente 3 - Navegação para lugares já conhecidos: Quando o robô sai de um nó filho e busca retornar para um nó pai. Esta componente é descrita na seção 6.4;
- Componente 4 - Navegação para lugares ainda não conhecidos: A seção 6.5 explica o que é feito quando o robô navega na direção de um lugar ainda não conhecido, onde um novo nó será criado;

Para após a exploração, é proposta uma quinta componente:

- Componente 5 – Refinamento do modelo criado através de algoritmos genéticos: Esta componente tem como objetivo refinar o grafo criado ao longo da exploração e foi experimentada dentro de ambiente simulado. Esta componente será apresentada na seção 6.7;

Nas seções dadas, os detalhes de cada componente e suas possíveis implementações são apresentados, com exceção da componente 5. Na presente seção 6.1, será apresentada uma pequena descrição das abordagens utilizadas para cada uma das componentes do algoritmo.

PUC-Rio - Certificação Digital Nº 0410268/CA

6.1.1. Panorâmicas

Em cada nó, uma série de imagens é obtida sequencialmente com o robô girando 360^{o} em torno de uma localização dada. Consecutivamente, estas imagens devem ser combinadas automaticamente de modo a construir uma composição panorâmica relativa àquele nó.

Os métodos de combinação de imagens, comumente conhecidos como métodos de registro de imagens, consistem em encontrar pontos, ou regiões,

correlatos entre duas imagens para então calcular um modelo que transforme uma das imagens para que esta possa ser sobreposta à outra (como discutido no capítulo 4). Por fim a sobreposição é feita através de ajuste radiométrico dos *pixels* de cada uma das imagens.

Para se encontrar as regiões coincidentes entre as imagens subseqüentes optou-se por trabalhar com Correlação Cruzada. Esta escolha foi feita devido ao custo computacional da técnica em relação a técnicas mais robustas como SIFT. Posteriormente são aplicados procedimentos diversos de modo a se eliminar regiões mal correlacionadas.

O capítulo 4 descreve algumas teorias e técnicas de registro de imagens utilizados no presente trabalho. A seção 6.2 apresenta como foram desenvolvidos os experimentos para a geração automática das imagens panorâmicas.

A partir da imagem panorâmica deve se fazer a identificação dos lugares para os quais o robô irá navegar como será visto na próxima seção.

PUC-Rio - Certificação Digital Nº 0410268/CA

6.1.2. Identificação de Lugares de Interesse

A segunda componente do algoritmo é a identificação de potenciais nós filhos para uma localização dada. Para cada nó, potenciais nós filhos são escolhidos através de uma análise da imagem panorâmica. Esta análise tem como objetivo a identificação de áreas de interesse. Estas áreas devem ser selecionadas de acordo com características que as diferenciem do resto da imagem.

A abordagem escolhida para se definir as regiões de interesse da imagem panorâmica foi baseada no cálculo de entropia para diferentes seções da imagem. A entropia de uma imagem é uma métrica da quantidade de informação presente no histograma de tons de cinza da mesma. Ao se buscar regiões com grande entropia, é possível se encontrar áreas distintas com grandes variações na imagem tais como quinas, portas, objetos, e outros.

Se determinada região de interesse contiver quantidade suficiente de informação, então esta área é identificada como um nó potencial e deve ser explorada.

A seção 6.3 do presente capitulo apresenta algumas possibilidades para se fazer a segmentação de uma imagem de modo a se obter regiões de interesse

baseadas em informação. Os métodos descritos se baseiam em dividir uma imagem através de um *quadtree* que trabalha com o valor de entropia das regiões para determinar contínuas sub-divisões.

Identificados os locais de interesse, o robô deverá fazer a navegação para estes como será descrito na seção seguinte.

6.1.3. Navegação para Nós Desconhecidos

A navegação para nós desconhecidos consiste em alinhar o robô continuamente com a região de interesse escolhida na imagem panorâmica que represente o nó a ser explorado. Tendo o robô se direcionado corretamente, este segue em frente até encontrar algum obstáculo ou então começar a perder de vista a região de interesse procurada.

Escolheu-se trabalhar com descritores SIFT de modo a encontrar a região de interesse nas imagens obtidas para a partir de então se fazer o acompanhamento destas regiões durante a navegação do robô.

O procedimento pode ser entendido através de duas etapas:

- Deve-se encontrar na imagem vista a região de interesse que represente o nó a ser explorado. Esta correspondência é feita através de descritores SIFT;
- Tendo-se encontrado pontos em comum entre a região de interesse e a imagem vista pelo robô, passa a se fazer um acompanhamento visual destes pontos através de *Visual Tracking* por correlação, permitindo o alinhamento adequado constante do robô com o nó pai. Este procedimento é feito até se encontrar algum obstáculo ou até que a região de interesse comece a sair da tela por cima ou por baixo;

A seção 6.4 descreve as possibilidades de se trabalhar com *Visual Tracking* de pontos na tela a partir de correlação.

A seção 6.5 explica em maiores detalhes a navegação para nós desconhecidos utilizando-se de descritores SIFT e de *Visual Tracking*.

Para voltar para os nós já conhecidos outros procedimentos são abordados como visto na próxima seção.

PUC-Rio - Certificação Digital Nº 0410268/CA

6.1.4.
Navegação para Nós Conhecidos

A navegação para nós já conhecidos usa como base de comparação a imagem que se espera ver quando se chegar ao nó. Esta imagem é extraída da panorâmica do nó pai, para o qual se navegará, tendo como base a direção em que aponta o nó filho para o nó pai.

Tendo esta imagem como referência, dois procedimentos diferentes foram experimentados de modo a alinhar o robô com o nó pai e navegar para este. As duas abordagens diferentes tomadas foram: transformações invariáveis e uso da transformação SIFT junto com *Visual Tracking*.

Na primeira abordagem, a imagem de referência é constantemente comparada com a visão atual do robô utilizando-se uma transformação invariável. Quando a comparação começa a retornar valores inferiores ao esperado, é feita uma busca pelo melhor alinhamento do robô para que a navegação possa continuar. Algumas das transformações apresentadas no capítulo 2 foram testadas.

A segunda abordagem consiste de duas etapas:

- Busca da área da imagem atual correspondente à imagem de referência através de descritores SIFT;
- Realização de um acompanhamento dos pontos encontrados por correlação em comum entre imagens na etapa anterior. Este acompanhamento é em tempo real (5 a 10 Hz nos experimentos realizados) permitindo um constante alinhamento do robô com o nó pai;

A seção 6.4 do presente capítulo descreve como é feito o *Visual Tracking* de modo a se perseguir os pontos encontrados através do uso da transformação SIFT.

A seção 6.6 descreve em detalhes tanto a navegação para um nó conhecido usando a transformação SIFT quanto a primeira abordagem feita do problema utilizando-se transformações invariáveis.

6.2.
Componente 1 - Criando Imagens Panorâmicas Automaticamente

A presente seção descreve a componente 1 do algoritmo de exploração proposto.

PUC-Rio - Certificação Digital Nº 0410268/CA

É desejado criar uma imagem panorâmica através de uma seqüência de imagens obtidas por um robô girando sobre seu próprio centro. O robô vai girando de um número pequeno de graus por vez e obtendo as imagens a serem combinadas. Após um giro completo de 360^{o}, estas imagens devem ser utilizadas para compor uma imagem panorâmica da posição em que o robô efetuou o giro. Para tal é necessário combinar cada par de imagens vizinhas através de um processo de registro automático.

A combinação destas imagens a partir da informação de ângulo (que poderia vir a ser extraída do sistema utilizado) implicaria em erros de sobreposição das imagens derivados dos erros das medidas de ângulos. Para o sistema experimental utilizado, o problema é ainda mais crítico, dado que a informação de orientação do robô não pode ser obtida diretamente.

Este mapeamento panorâmico pode ser usado num segundo momento para que se conheça a imagem que o robô estaria vendo em um ângulo arbitrado, mesmo que o robô não tenha obtido uma imagem para aquele ângulo específico.

Outro uso futuro está em se utilizar as imagens panorâmicas obtidas para se escolher áreas interessantes do ambiente para serem exploradas. A partir da comparação da panorâmica com a visão do robô, este pode se dirigir para estas áreas de interesse e explorá-las.

O capítulo 4 apresentou uma visão geral de como são usados métodos de registro. Para se compor a panorâmica algumas das técnicas apresentadas foram utilizadas.

A técnica proposta encontra os pontos de controle automaticamente através da correlação de blocos da imagem a ser transformada com a imagem de referência. Os parâmetros da função de transformação são obtidos utilizando-se a matriz pseudoinversa da jacobiana que relaciona a posição na imagem base dos pontos de controle com os parâmetros. Por fim, a sobreposição das imagens é feita utilizando-se ajuste radiométrico das intensidades dos *pixels* sobrepostos por *dégradée*.

A imagem panorâmica resultante pode ser utilizada para extração da imagem de um ângulo arbitrado. Isto pode ser feito porque as posições no panorama referentes ao centro de cada imagem obtida são guardadas tal como os ângulos referentes a estas imagens. Daí, esta informação pode ser utilizada para se

PUC-Rio - Certificação Digital Nº 0410268/CA

interpolar onde estaria no panorama o centro de uma imagem em um ângulo arbitrado.

A seção seguinte descreve como foi feita a comparação das imagens de um modo geral.

6.2.1. Comparação Entre Imagens e Construção da Matriz de Transformação *T*

Optou-se por trabalhar com um método rápido e eficiente, que atendesse ao sistema experimental usado, trabalhando-se com correlação. A busca dos pontos em comum por correlação cruzada não é tão poderosa quanto o uso de técnicas mais complexas, como SIFT, mas tem como vantagem menor custo computacional. Apesar de não ser extremamente robusto, o método demonstrou ser confiável e eficiente para a geração de panorâmicas usando-se o ER1.

Os experimentos feitos para se comparar cada par de imagem seguiram de um modo geral as seguintes etapas:

- Detecção de pontos de interesse: Como dito, optou-se trabalhar com correlação cruzada. As imagens são sub-dividas em blocos para então serem casadas em outras imagens. Inicialmente blocos com baixa entropia podem ser descartados. Este tópico está detalhado na seção 6.2.2;
- Descarte de pontos inconsistentes: Algumas das possibilidades testadas serão apresentadas na seção 6.2.3, outras foram vistas no capítulo 4;
- Cálculo final da matriz de transformação T como visto no capítulo 4;

A detecção de pontos de interesse está detalhada na seção a seguir.

PUC-Rio - Certificação Digital Nº 0410268/CA

6.2.2. Detecção e Casamento dos Pontos de Controle por Correlação Cruzada

Sempre se busca encontrar pontos de uma imagem em outra, sendo a primeira a imagem a ser transformada e a segunda a imagem de referência. Estes pontos comuns entre as imagens serão chamados de pontos de controle.

Inicialmente, deve-se dividir a imagem a ser transformada em blocos de tamanho N x M. Estes blocos serão procurados na imagem de referência. A idéia está exemplificada na Figura 6-5 que mostra à esquerda a imagem a ser transformada dividida em blocos, e à direita a imagem de referência onde alguns blocos são encontrados.

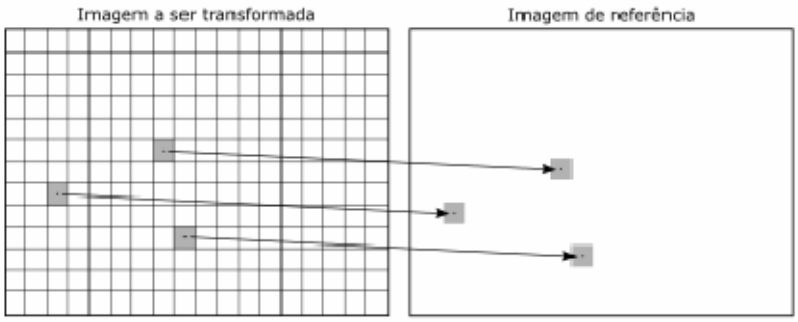

Figura 6-5: Detecção e casamento de pontos de controle

É interessante que o tamanho definido seja adequado: nem muito pequeno, dificultando a procura por falta de informação suficiente, nem muito grande, o que torna a procura mais lenta e ineficaz.

De um modo geral, ao dividir-se a imagem em blocos, pode-se criar um número de blocos muito grande. Portanto, caso todos os blocos sejam procurados, pode-se ter um tempo de procura muito grande e até desnecessário. É interessante buscar maneiras de se escolher um conjunto de blocos do todo que sejam mais adequados para a procura de modo a tornar a detecção dos pontos de controle mais rápida.

Percebeu-se que blocos com maior entropia têm maior chance de serem encontrados corretamente na imagem de referência. Isto acontece porque quanto mais distinto for um bloco, maior será a chance de se encontrar um bloco em outra imagem que case com ele de modo único. Blocos com baixa entropia são mais uniformes e portanto mais fáceis de serem erroneamente identificados.

PUC-Rio - Certificação Digital Nº 0410268/CA

Portanto, é uma boa idéia de pré-seleção de blocos, se eliminar aqueles com baixa entropia. Isto pode ser feito tanto definindo-se um limiar quanto definindo-se um número *k* de blocos com maior entropia para serem procurados.

A seção 6.3.2 deste capítulo descreve o uso da entropia de imagens.

A entropia de cada um destes blocos é calculada pela fórmula de entropia [17]:

$$H(q_1,q_2,...,q_n) = -\sum_{k=1}^{n} q_k \log_2 q_k \tag{165}$$

$$q_i = f_i / N \quad \text{para} \quad \text{i} = 1 \ldots \text{N}_{\text{gray}} \tag{166}$$

Onde:

- q: representa o histograma de tons de cinza da imagem;
- f_i: Número de *pixels* na imagem com o tom de cinza i;
- N: Número de *pixels* da imagem;
- Ngray: Número de níveis de cinza utilizados;

Então, selecionam-se k blocos com maior entropia. Como dito, isto é feito de modo a descartar aqueles blocos que por terem baixa quantidade de informação, poderiam levar a casamentos incorretos de blocos.

Os blocos restantes são procurados na imagem de referência através da maximização da correlação entre os blocos e a imagem como apresentado em 4.2.

Obtém-se a posição x, y de casamento de cada bloco na imagem base encontrando-se x' e y' que maximize a correlação cruzada CC(x',y'). Os blocos têm seus centros utilizados como pontos de controle.

O casamento dos blocos por correlação possui a desvantagem de não funcionar muito bem para blocos que tenham sofrido grande distorção ou variação de intensidade. Observe que o procedimento apresentado é bem específico para o casamento de imagens com mudanças de escala e rotação bem pequenas, se adaptando bem para o caso da construção de panorâmicas e sendo bastante eficiente em termos de processamento em relação a procedimentos mais complexos para a definição dos parâmetros do modelo de transformação entre as imagens como os vistos no capítulo 4.

É proposto que após encontrar blocos correspondentes, como apresentado, seja feita uma seleção dos pontos de controle consistentes utilizando os métodos aqui propostos e outros apresentados no capítulo 4.

PUC-Rio - Certificação Digital Nº 0410268/CA

Após encontrar blocos em comum, é interessante fazer o descarte de pontos de controle inconsistentes como visto na próxima seção.

6.2.3. Descarte de Pontos de Controle Inconsistentes

Após ser feita a busca dos blocos por correlação, muitos pontos não serão casados corretamente. Algumas idéias foram experimentadas de modo a descartar pontos inconsistentes e serão aqui descritas.

6.2.3.1. Baixa Correlação

É feito um processo de seleção de pontos eliminando blocos que receberam baixo valor de correlação. Esse valor é relativo à correlação máxima entre bloco e imagem de referência. Tal como para a seleção de blocos por entropia, é possível fazer a eliminação destes pontos utilizando-se um limiar ou escolhendo um número *k* de pontos com maior correlação.

O uso de um limiar de correlação é bem interessante e eficaz. Porém, tem como desvantagem não ser possível se determinar o número de pontos restantes, ou mesmo se haverá pontos restantes.

O uso de um número determinado de pontos com melhores valores de correlação implica em ou não se eliminar pontos mal correlacionados, ou se eliminar pontos com boa correlação.

6.2.3.2. Má Correlação

Uma outra maneira de se verificar se um ponto está sendo bem ou mal correlacionado consiste em conferir se este obteve alta correlação para um grande número de pontos na imagem de referência. Quando isto ocorre pode-se entender que este ponto não está sendo identificado de modo único porque existem vários locais para os quais o bloco possui grande semelhança. Será portanto qualificado como um ponto de má correlação.

PUC-Rio - Certificação Digital Nº 0410268/CA

A idéia proposta está em definir C_{max} como o maior valor de correlação obtido e a partir de então contar para quantos pontos se obteve uma correlação próxima desta. O valor utilizado para a comparação é definido como:

$$Caux = \rho C\max \tag{167}$$

Com $0 < \rho < 1$.

A partir de então, é contado o número de pontos com correlação superior a C_{max}. Caso o percentual de pontos computado (em relação ao número total de pontos correlacionados) seja superior a β dado, então este ponto pode ser descartado e qualificado como um ponto com má correlação.

Esta técnica também será usada mais adiante na descrição dos procedimentos de "*Visual Tracking"*.

6.2.3.3. Eliminação por Análise Amostral

PUC-Rio - Certificação Digital Nº 0410268/CA

É proposta uma análise dos vetores que levam o bloco casado da imagem de referência para o bloco original da imagem a ser transformada. São descartados aqueles blocos cujo vetor de movimento fuja do padrão geral. Isto é feito de modo a tentar garantir que os blocos resultantes tenham sido encontrados na imagem de referência corretamente.

A representação pontual dos blocos é dada pelo seu centro. Dados os centros dos blocos que apresentaram correspondências, é possível definir um vetor que apresenta o movimento relativo de um bloco ao outro. Sabendo-se da posição x_1,y_1 do bloco w na imagem a ser transformada e da posição x_2,y_2 na imagem base, temos um vetor de movimento de cada bloco dado por:

$$V_w = \begin{bmatrix} x_2 - x_1 \\ y_2 - y_1 \end{bmatrix} \tag{168}$$

Este vetor possui norma:

$$[V_w] = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{169}$$

E ângulo:

$$\theta_w = \arctan\left(\frac{y_2 - y_1}{x_2 - x_1}\right) \tag{170}$$

Nas imagens panorâmicas a serem compostas, a relação de coordenadas entre imagens subseqüentes pode ser razoavelmente aproximada por uma

translação em *(x,y)*. Portanto, o vetor de movimento calculado, caso corresponda a pontos equivalentes entre as imagens, constitui de uma boa aproximação da transformação entre coordenadas.

Consecutivamente, são calculados as médias e o desvio padrão para as normas e ângulos dos blocos. São descartados todos blocos que ficam fora do desvio padrão para norma ou ângulo.

Após o descarte de blocos, o procedimento pode ser repetido até que nenhum bloco seja descartado.

Observe que este procedimento não trará bons resultados para comparação de imagens que sofreram transformações geométricas ou de rotação, mas somente para casos em que a transformação entre imagens pode ser aproximada por uma translação. Como este é o caso de imagens panorâmicas, esta proposta traz bons resultados.

Uma desvantagem da técnica está em que o número de pontos descartados pode ser grande, podendo-se chegar a somente dois pontos restantes. Se este for o caso, é necessário aproximar a transformação T por uma transformação Procrustes.

PUC-Rio - Certificação Digital Nº 0410268/CA

6.2.3.4. RANSAC e Ajuste de Matriz de Pesos

As técnicas RANSAC e ajuste da matriz de pesos apresentadas no capítulo 4 trazem bons resultados no momento de se eliminar pontos mal casados. O uso destas técnicas também é interessante para se ajustar o modelo de transformação T.

Outra técnica que poderia ser aplicada aqui é a Transformada de Hough como descrita na seção 4.4.1, porém esta não foi utilizada nos experimentos realizados.

Finalmente, é visto na seção seguinte como é criada a imagem panorâmica utilizando as técnicas apresentadas.

6.2.4.
Criando a panorâmica – Registro de Múltiplas Imagens

Para se fazer o registro de uma seqüência de imagens são encontrados os pontos de controle para cada par de imagens vizinhas e estimados os parâmetros de todas as transformações entre imagens vizinhas, para só então fazer a união de todas.

É dada uma seqüência de *n* imagens $I_1, I_2, \cdots, I_n$, ordenadas em série de modo que I_x esteja intercalada entre I_{x-1} e I_{x+1} para todo *x*. Encontram-se os pontos de controle entre cada par de imagens vizinhas e são obtidas as transformadas T_{i+1} que levariam as imagens I_{i+1} para I_i (observe que $i+1$ se refere à imagem a ser transformada e *i* à imagem base).

Desta maneira são calculadas as transformadas entre todas as imagens vizinhas e fica estabelecido que:

$$T_1 = \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{vmatrix} \tag{171}$$

A partir de então para se calcular a transformação T^i que levaria a imagem I_i para a panorâmica, basta fazer:

$$T^i = T_1 \cdot \ldots \cdot T_i \tag{172}$$

Por fim cada imagem é por sua vez levada para a panorâmica fazendo-se o devido ajuste radiométrico [39].

Veja na Figura 6-6 o método ilustrado. A figura apresenta uma seqüência de imagens para as quais são calculadas as matrizes T_i para cada par, e por fim são calculadas as matrizes T^i que são usadas para construir a panorâmica.

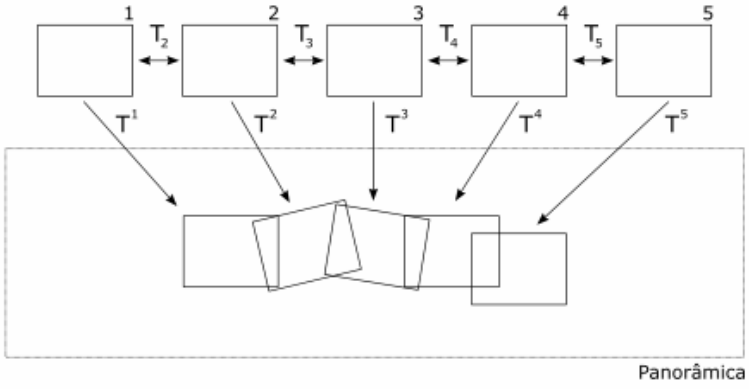

Figura 6-6: Registro de múltiplas imagens

PUC-Rio - Certificação Digital Nº 0410268/CA

A panorâmica criada pode ser usada para se extrair uma imagem vista em ângulo especificado. Isto é descrito na seção seguinte.

6.2.5. Extraindo Imagens da Panorâmica

Cada imagem I_i é obtida em um ângulo Θ_i específico. Portanto, para calcular onde estaria na imagem panorâmica o centro relativo a cada ângulo Θ_i, basta aplicar a transformação T^i usada para a posição do centro de uma imagem de tamanho *sx* x *sy* dado pela aproximação de:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} [sx/2] \\ [sy/2] \end{bmatrix} \tag{173}$$

Desta maneira pode-se criar uma tabela do tipo:

Tabela 6-1: Tabela que acompanha a imagem panorâmica

Ângulo Θ_i	Posição em *x*	Posição em *y*

Portanto ao se arbitrar um ângulo qualquer, basta calcular o centro da imagem a ser extraída através de interpolação utilizando-se a tabela criada.

Caso o modelo de transformação seja o Procrustes, pode-se armazenar na tabela o valor do parâmetro Θ das transformadas e trabalhar com sua interpolação para saber o ângulo da imagem a ser extraída da panorâmica. Outro modo seria trabalhar com imagens horizontais somente.

O tamanho das imagens deve respeitar o tamanho das imagens da panorâmica.

A partir das imagens panorâmicas é definido para onde o robô irá navegar. Esta é a componente 2 do algoritmo de exploração e é vista na seção seguinte.

PUC-Rio - Certificação Digital Nº 0410268/CA

6.3.
Componente 2 - Segmentação da Imagem por *Quadtree* Baseada em Entropia

6.3.1.
Visão Geral

Sujan propõe em [48] o uso de um algoritmo *quadtree* baseado em entropia para encontrar regiões de interesse em uma imagem. Aqui este algoritmo é explorado e adaptado de modo a segmentar uma imagem em regiões de interesse. O objetivo final é definir áreas a serem exploradas pelo robô de acordo com regiões que apresentem maior quantidade de informação para serem exploradas.

A idéia do algoritmo proposto é de dividir a imagem em regiões utilizando-se a técnica de *quadtree*. Para cada região da imagem, esta será subdividida em novas regiões caso possua quantidade de informação desejada. Por fim, o particionamento *quadtree* é utilizado para fazer a segmentação da imagem em regiões e pontos de interesse.

Na Figura 6-7, é apresentada uma visão geral do algoritmo.

O procedimento pode ser entendido através de três componentes:

- Pré-processamento: Aplicação de filtros na imagem de modo a eliminar efeitos de iluminação e possíveis ruídos;
- Divisão da imagem: A imagem é recursivamente sub-dividida através da técnica "*quadtree*" de regiões com alta quantidade de informação. No caso, alta entropia;
- Pós-processamento: A partir da árvore "*quadtree*" gerada, é feita a segmentação da imagem em áreas e pontos de interesse (*feature points*);

As próximas seções detalham o procedimento de segmentação proposto. A seção a seguir apresenta o cálculo de entropia de uma imagem e descreve suas características.

PUC-Rio - Certificação Digital Nº 0410268/CA

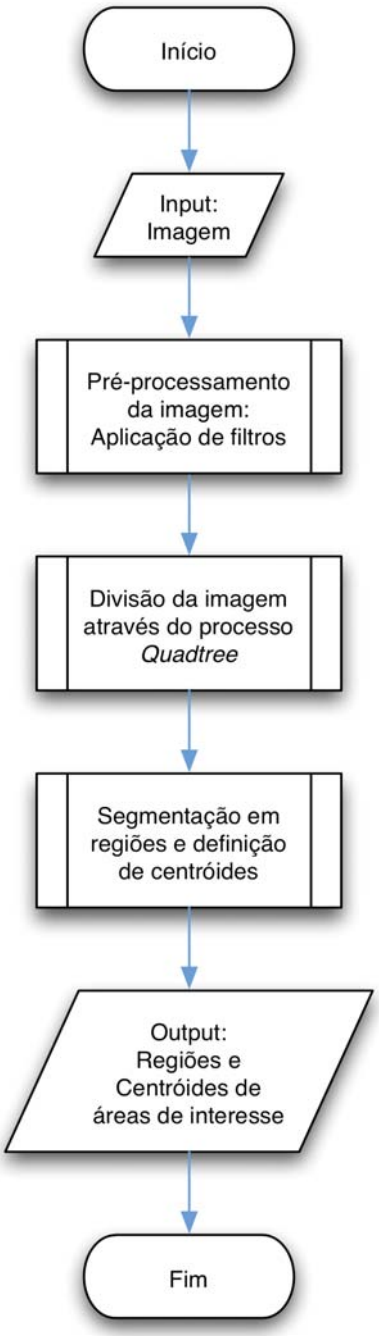

Figura 6-7: Visão geral do algoritmo de segmentação

6.3.2. Cálculo de Entropia

A informação ganha ao se observar um evento específico dentre uma amostra de possibilidades é descrita pela função de entropia dada pela eq. (165) vista na seção 6.2.2.

PUC-Rio - Certificação Digital Nº 0410268/CA

Esta definição de informação [17] também pode ser interpretada como o menor número de estados (bits) necessários para descrever completamente uma amostra de dados.

A teoria da informação dá ênfase na descrição de sinais 1-D. Para sinais em 2-D como imagens, o histograma de tons de cinzas, visto na eq. (166) da seção 6.2.2, pode ser usado para definir a distribuição de probabilidades.

Com esta definição, a entropia máxima de uma imagem será dada por uma distribuição dos níveis de cinzas uniforme, ou seja, máximo contraste. Quanto menos uniforme é o histograma, menor será a entropia da imagem.

Para uma imagem, existem diversas interpretações para a entropia, incluindo:

- A incerteza média relativa ao histograma;
- A quantidade teórica mínima de bits necessários para se codificar o histograma;
- A medida de aleatoriedade na distribuição de tons de cinza da imagem;

Um aumento na entropia corresponde à maior incerteza e maior informação contida na imagem. Aqui o interesse está na terceira interpretação de onde se conclui que a entropia de uma imagem pode ser usada para se apontar regiões de interesse na mesma.

A próxima seção descreve como é feito o procedimento *quadtree* utilizando a métrica de entropia apresentada.

PUC-Rio - Certificação Digital Nº 0410268/CA

6.3.3. Processo *Quadtree*

No particionamento *Quadtree*, a imagem é sucessivamente dividida em 4 quadrantes. Cada quadrante é avaliado de modo a se determinar se novas divisões necessitam ser realizadas. Esta avaliação é baseada em alguma métrica. O processo deve continuar recursivamente até que nenhuma nova divisão seja necessária. A Figura 6-8 ilustra o procedimento para uma imagem que inicialmente é divida em quatro regiões, das quais somente a região 2 é novamente dividida.

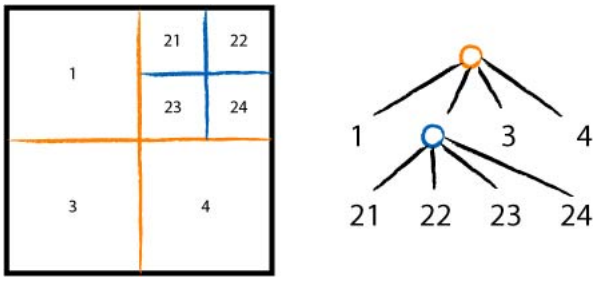

Figura 6-8: *Quadtree*

A métrica utilizada pode ser escolhida de acordo com a necessidade da aplicação. O particionamento *Quadtree* proposto usa como condição para a divisão a quantidade de informação, entropia, da região a ser subdividida. O particionamento *Quadtree* é feito dividindo-se as regiões que estiverem acima de determinado limite escolhido. Repare que alguma outra medida poderia ser usada para tal fim.

O objetivo de se determinar a quantidade de informação na imagem é de se identificar regiões onde existam mudanças significativas. Estas mudanças poderiam indicar a presença de cantos, extremidades, quinas, objetos e outras áreas interessantes para serem exploradas.

Áreas com grande quantidade de ruído também poderiam ser identificadas com alta entropia, por isso indica-se o uso de filtros para se fazer um pré-processamento da imagem antes que esta seja subdividida.

Na Figura 6-9 o procedimento está apresentado em diagrama de fluxo do método usando-se entropia como métrica de decisão. Inicialmente a imagem é sub-dividida em quatro regiões. Cada região pode ser filtrada. Caso a entropia de uma região seja superior a um limite, entra-se em um processo recursivo onde esta será subdividida. O processo termina quando não houver mais regiões a serem subdivididas.

A Figura 6-10 apresenta um exemplo do aplicação da técnica utilizando-se entropia. A figura apresenta uma imagem após ter sido particionada através do *quadtree*. As linhas brancas mostram as divisões encontradas.

A próxima seção explicará como usar a divisão *quadtree* apresentada para se segmentar uma imagem em áreas de interesse.

PUC-Rio - Certificação Digital Nº 0410268/CA

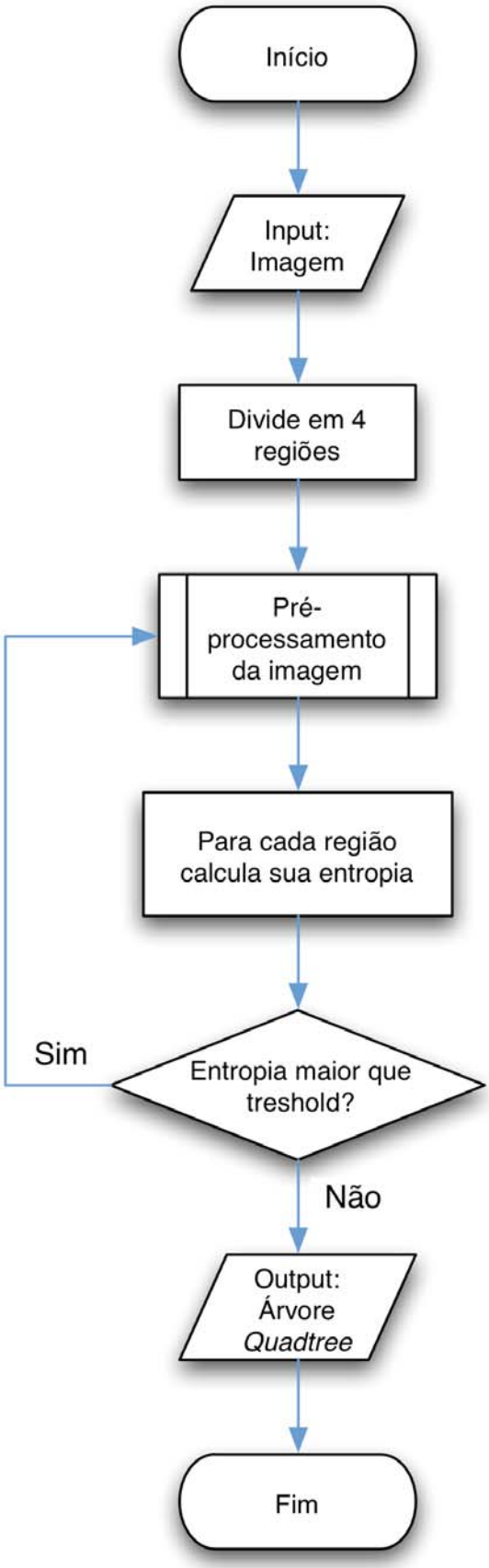

Figura 6-9: Divisão por processo *Quadtree*

PUC-Rio - Certificação Digital Nº 0410268/CA

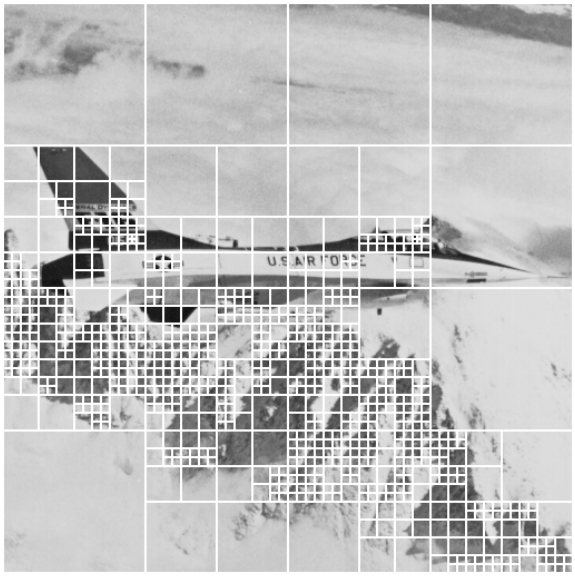

Figura 6-10: Exemplo da divisão baseada em entropia

6.3.4. Segmentação em Áreas de Interesse e Definição de *Feature Points*

Após o particionamento *quadtree* da imagem, deseja-se fazer uma segmentação da mesma em regiões de interesse que representem áreas com grande quantidade de informação, como já foi explicado. Para se fazer esta segmentação são propostos três métodos diferentes:

- Método padrão;
- Método Open-Close;
- Método Close-Open;

Estes métodos tem como linha geral agrupar os menores quadrantes encontrados e aplicar operações morfológicas nestes de modo a se conseguir criar conglomerados que representem as regiões de interesse da imagem. Estes métodos serão descritos nas seções seguintes.

A primeira coisa a ser feita é a criação de uma imagem modelo, do mesmo tamanho da imagem analisada, com todos os seus *pixels* zerados. Esta imagem será usada para se criar o mapa das regiões segmentadas, ou seja, indicará se determinado *pixel* faz parte ou não de uma região.

O procedimento, independente do método utilizado, pode ser resumido da seguinte maneira:

1. Criação de uma imagem *template* com tamanho igual à imagem original com todos os *pixels* desligados. Como dito anteriormente;

PUC-Rio - Certificação Digital Nº 0410268/CA

2. Todos os *pixels* que estejam nas bordas das regiões do *quadtree* são ativados. Isto facilitará o futuro agrupamento das regiões ;
3. Liga todos os *pixels* que estejam nas menores regiões do *quadtree*. Isto pode ser feito para as regiões até um determinado tamanho. A escolha deste tamanho pode ser pré-determinada ou feita a partir de um número de menores tamanhos;
4. Aplica-se uma série de operações morfológicas no *template* de acordo com o método escolhido. Estas operações serão descritas a partir da próxima seção, quando são detalhados os diferentes métodos propostos;
5. Faz-se um pós-processamento no *template* de modo a se descartar regiões muito pequenas;

O resultado deste processo será uma série de regiões. Por fim, são calculados os centróides das regiões segmentadas. Estes centróides são definidos como *feature points*, ou seja, pontos de interesse para se prosseguir a exploração do robô.

As três seções seguintes descrevem os métodos de segmentação propostos.

6.3.5. Método Padrão

O primeiro método apresentado foi denominado de método padrão por ter sido originalmente proposto em [48] , os outros métodos apresentados foram desenvolvidos no presente trabalho.

O método padrão consiste em se aplicar a operação morfológica de abertura na imagem *template* com um disco de raio *r* dado. O diagrama de fluxo do método padrão é apresentado na Figura 6-11. Este método não agrupa muito bem as regiões como pode ser visto na Figura 6-12 que mostra um exemplo do método aplicado. O que pode ser visto na Figura 6-12 são as diversas regiões resultantes do método empregado.

A próxima seção apresenta outro método de segmentação baseado nas operações morfológicas de abertura e fechamento.

PUC-Rio - Certificação Digital Nº 0410268/CA

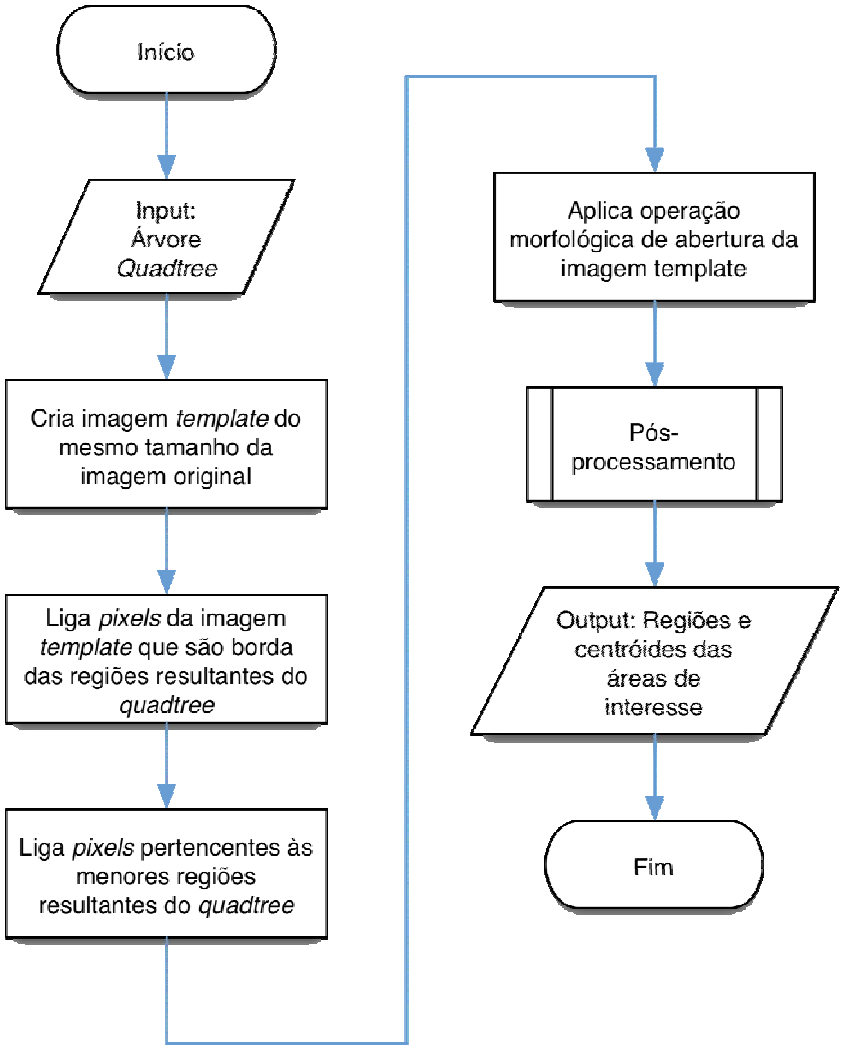

Figura 6-11: Método padrão

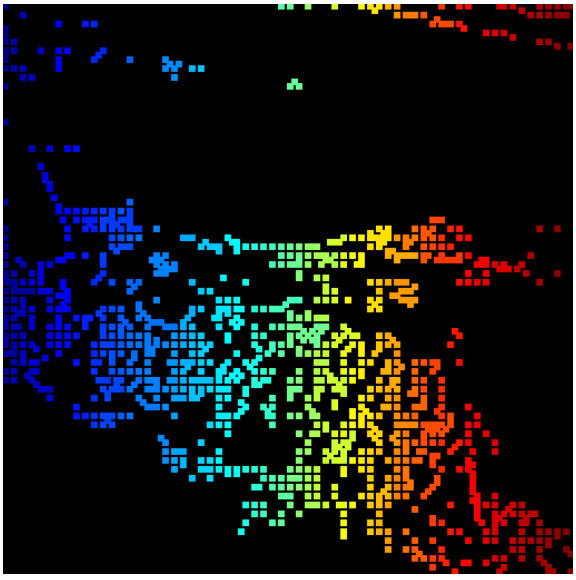

Figura 6-12: Exemplo do método padrão

PUC-Rio - Certificação Digital Nº 0410268/CA

6.3.6.
Método Open Close

O método Open Close se diferencia pela operação morfológica de abertura seguida da operação de fechamento na imagem *template*, ambas com um disco de raio *r* dado. Este método consegue agrupar bem as regiões sendo sensível a *r* escolhido. O diagrama de fluxo do método Open Close é apresentado na Figura 6-13. A Figura 6-14 apresenta um exemplo do método mostrando as regiões encontradas.

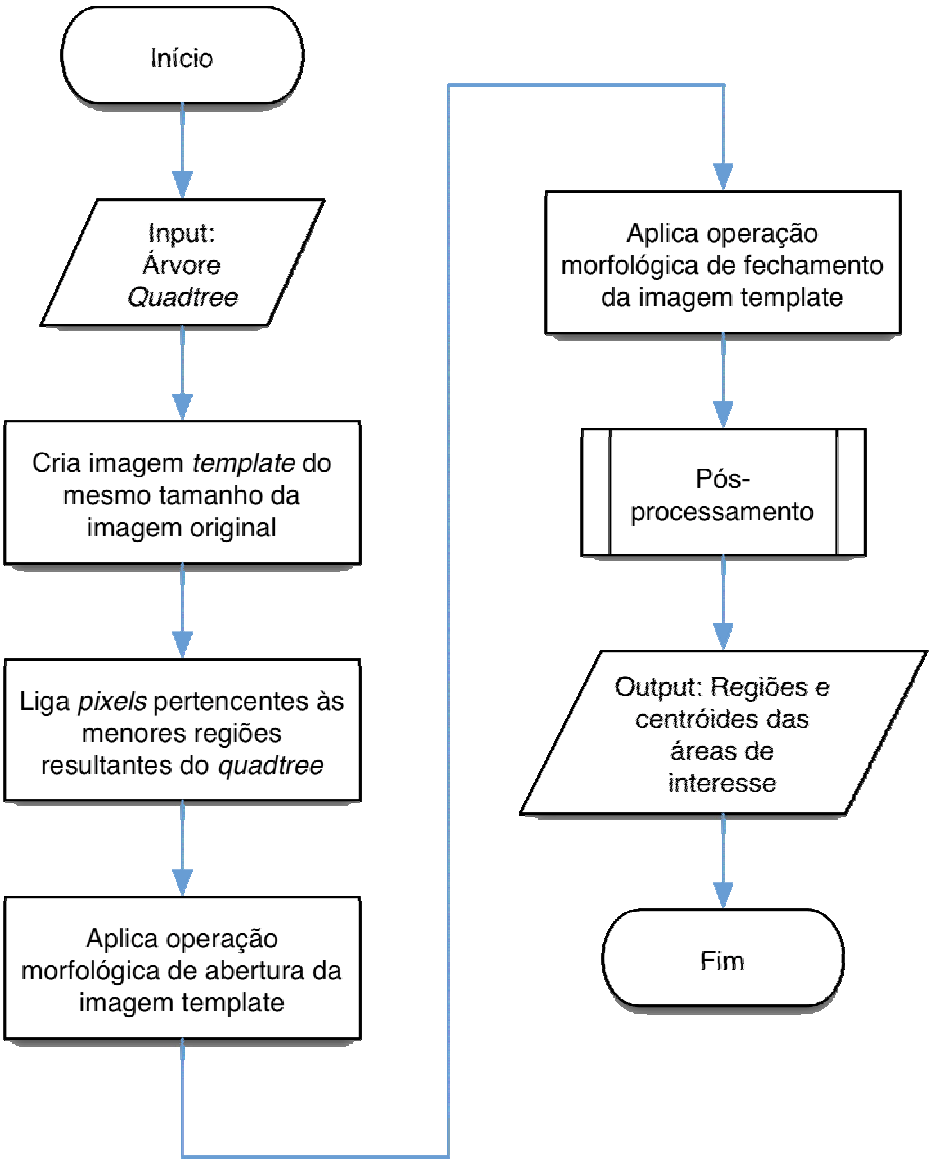

Figura 6-13: Método Open Close

PUC-Rio - Certificação Digital Nº 0410268/CA

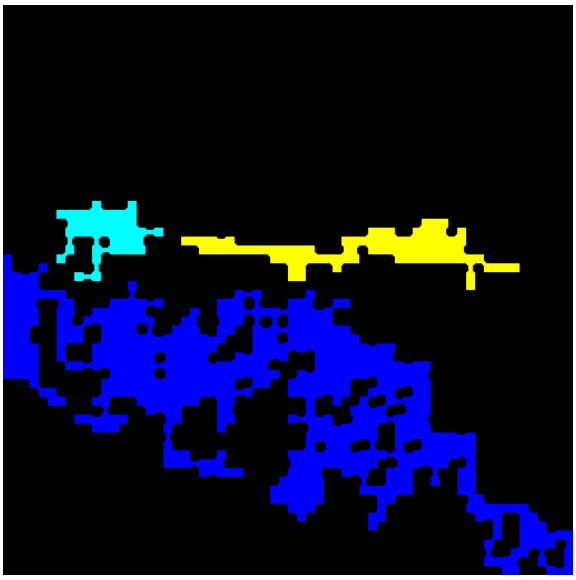

Figura 6-14: Exemplo do método Open Close

Outro método similar a este é proposto na seção seguinte.

PUC-Rio - Certificação Digital Nº 0410268/CA

6.3.7. Método Close Open

O método Close Open primeiro aplica a operação morfológica de fechamento e depois a de abertura. Como nos outros métodos, isso é feito com um disco de raio *r* dado. O diagrama de fluxo do método Open Close é apresentado na Figura 6-15. Como resultado são obtidas regiões que se assemelham a focos centrados em áreas de interesse como pode ser percebido pela Figura 6-16.

Este é o último método proposto para a segmentação usando *quadtree*. Após a escolha das regiões da panorâmica para as quais o robô deverá prosseguir, será feita a navegação para estes nós.

A comparação entre os três métodos propostos será apresentada com experimentos na seção 7.6.

A próxima seção descreve um algoritmo de *Visual Tracking* proposto para auxiliar as etapas de navegação para nós não explorados ainda e também para os nós já conhecidos.

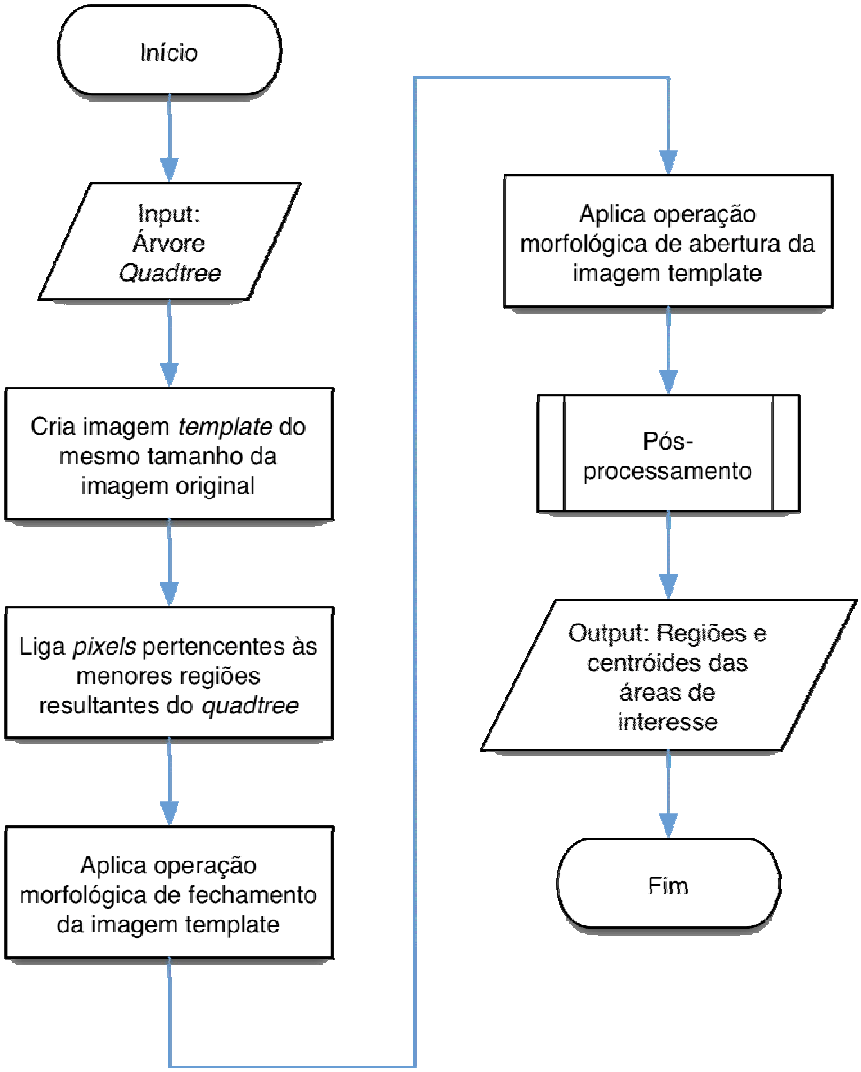

Figura 6-15: Método Close Open

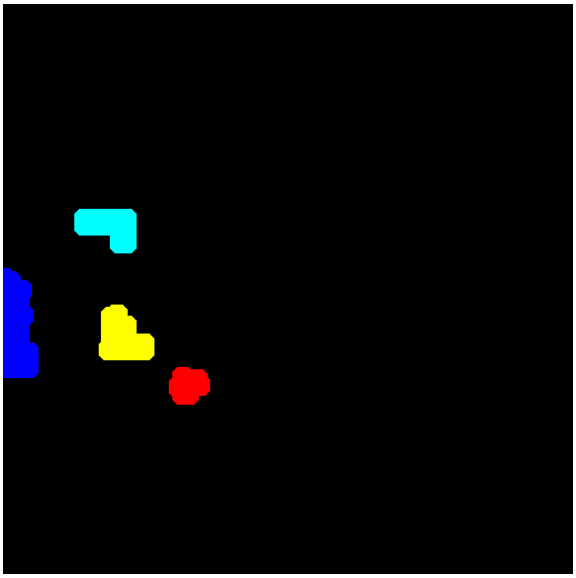

Figura 6-16: Exemplo do método close open

PUC-Rio - Certificação Digital Nº 0410268/CA

6.4. *Visual Tracking*: Acompanhamento de *Features* e Navegação

6.4.1. Introdução

Em diversos momentos, é interessante se fazer o acompanhamento de pontos na tela enquanto o robô está em movimento. A principal função deste acompanhamento está em permitir que o robô possa ajustar sua trajetória a partir das imagens observadas em tempo real (5 a 10 Hz nos experimentos realizados).

O objetivo é se acompanhar pontos que não se modifiquem muito ao longo da trajetória, possivelmente relativos ao fundo do ambiente observado. Perceba que pontos ao fundo são menos variáveis ao longo do tempo permitindo um melhor acompanhamento e a correção da direção do robô.

Algumas dificuldades em se fazer o acompanhamento de pontos na tela são:

- Podem ocorrer oclusões;
- A aparência de objetos e da cena sofre mudanças ao longo do tempo devido à iluminação ou mesmo mudanças geométricas relativas à câmera;
- Mudanças temporárias ou permanentes relativas a variações em 3D dos objetos ou cenas observados;

De modo a se lidar com mudanças relativas aos pontos observados, o procedimento deve se adaptar a pequenas mudanças. Um problema é que esta adaptação pode induzir a erros.

O procedimento também deve ser robusto a ponto de saber descartar pontos que não estão sendo acompanhados corretamente, ou que sofreram oclusão durante muito tempo, ou que não são mais encontrados com a certeza desejada.

Caso o número de pontos acompanhados caia muito, alguma estratégia mais robusta, como utilizando a técnica SIFT (descrita mais adiante), pode ser utilizada para se reencontrar as regiões que estão sendo acompanhadas. Isto implicaria em o robô parar de se movimentar para se reencontrar e depois prosseguir seu movimento.

De modo a se atender às necessidades apresentadas, optou-se por trabalhar com correlações de pequenas regiões na imagem para imagens subseqüentes, em

PUC-Rio - Certificação Digital Nº 0410268/CA

conjunto com algumas técnicas que serão apresentadas, de modo a se fazer o acompanhamento visual de pontos na tela quando necessário. Esta estratégia é similar à utilizada para a correspondência de pontos entre as imagens de uma panorâmica.

Como será visto, o algoritmo SIFT foi escolhido para se reencontrar as regiões procuradas caso fosse preciso.

O problema a ser tratado será definido da seguinte maneira:

- Deseja-se fazer o acompanhamento de coordenadas na tela chamadas de pontos chave. Estas coordenadas são acompanhadas indiretamente a partir de pontos de referência;
- Pontos de referência são pontos com características de interesse (particulares a cada aplicação) que serão procurados em *frames* subseqüentes e podem ser descartados ao longo do tempo. Da mesma forma, novos pontos de referência podem ser escolhidos;

A escolha dos pontos chave e dos pontos de referência é particular a cada problema tratado. A estratégia descrita a seguir é geral para pontos chave e pontos de referência dados.

Os tópicos teóricos utilizados no método a ser descrito foram apresentados nos capítulos 3 e 4.

A seção seguinte descreve como foi feita a correspondência dos pontos a cada momento.

PUC-Rio - Certificação Digital Nº 0410268/CA

6.4.2. Correspondência de Múltiplos Pontos para Imagens em Movimento

Definem-se os pontos chave por PC_i, onde i é o índice de cada um destes pontos. Cada ponto chave guarda as seguintes informações:

- Coordenadas *(x,y)* no momento inicial do *tracking*;
- Coordenadas *(x,y)* no momento atual;

Definem-se os pontos de referência por PR_i, onde i é o índice de cada um destes pontos. Cada ponto de referência guarda as seguintes informações:

- Coordenadas *(x,y)* no momento inicial do tracking;
- Coordenadas *(x,y)* no momento atual;

- Modelo, ou *template*, que descreve cada ponto e será usado para fazer sua correspondência em cada imagem. Este modelo será adaptado ao longo do tempo. O modelo de cada ponto é dado por w_i;
- Modelo inicial;

Como previamente citado, o acompanhamento dos pontos de referência será feito através de correlação. Portanto, o modelo inicial de cada ponto é dado por uma janela de tamanho *(Sx1, Sy1)* ao redor de cada ponto obtida no primeiro momento do *Visual Tracking*.

O diagrama de fluxo do processo é apresentado na Figura 6-17.

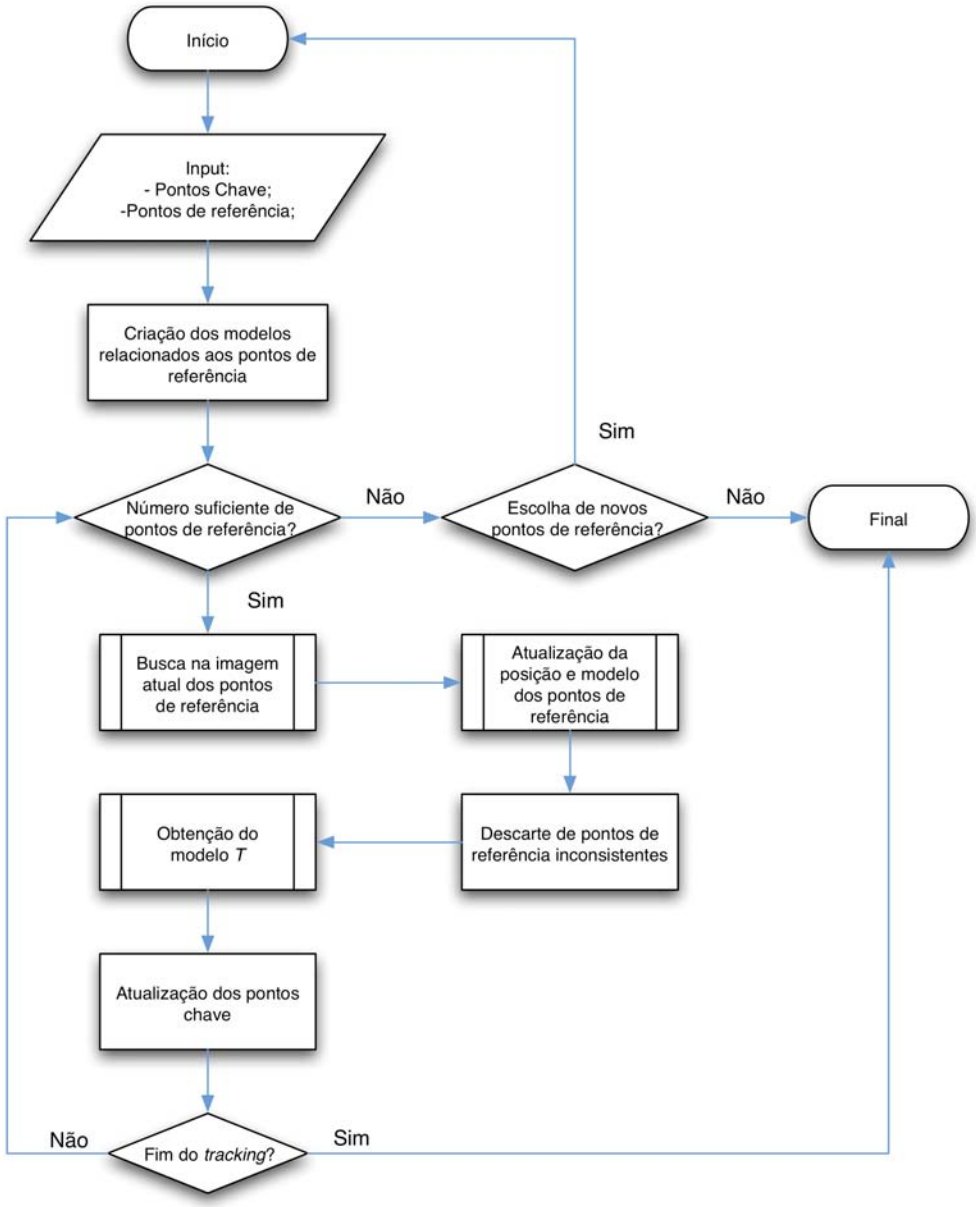

Figura 6-17: Diagrama do processo de *Visual Tracking*

PUC-Rio - Certificação Digital Nº 0410268/CA

A cada momento, as seguintes etapas são realizadas:

1. Busca dos pontos de referência: Procura dos pontos de referência na imagem atual através de correlação. Será descrito na seção 6.4.3;
2. Atualização dos pontos de referência: A posição e o modelo dos pontos pode ou não ser atualizada. Pontos ruins também são descartados. Esta etapa também será descrita na seção 6.4.3;
3. Obtenção do modelo: Escolha dos pontos que serão utilizados para se computar os pontos chave e cálculo da transformada afim que mapeia as posições iniciais dos pontos chaves para as posições atuais. Este passo está descrito na seção 6.4.4;
4. Caso não existam pontos de referência suficientes, o *tracking* é interrompido e novos pontos de referência devem ser escolhidos;

A seção seguinte descreve como é feita a busca dos pontos de referência a cada momento e como são atualizados estes pontos.

PUC-Rio - Certificação Digital Nº 0410268/CA

6.4.3. Busca e Atualização dos Pontos de Referência

A busca dos pontos de referência é feita para cada ponto buscando a posição de correlação máxima dentro de uma vizinhança de tamanho *(Sx2, Sy2)* ao redor da última posição de cada ponto. Veja a Figura 6-18. Caso a janela de busca extrapole o tamanho da imagem, o ponto de referência é descartado.

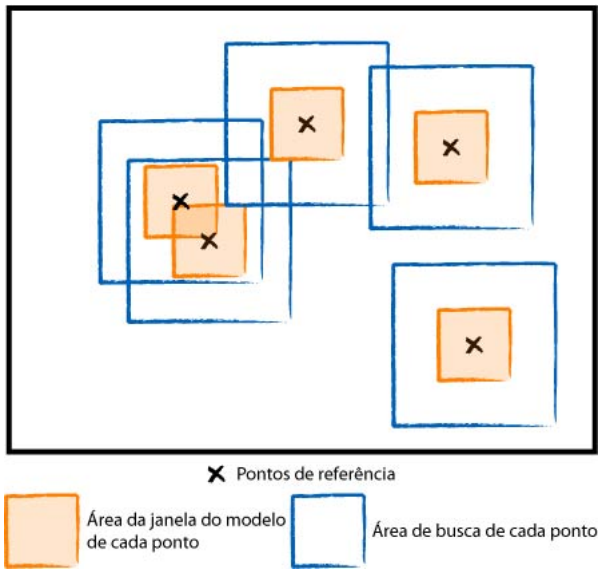

Figura 6-18: Busca dos pontos de referência

O valor de correlação máxima de um ponto PR_i será definido como $Cmax_i$.

A atualização de cada ponto deve seguir as seguintes regras:

- Pontos com $Cmax_i$ inferior a um limiar α escolhido não são atualizados;
- Define-se $Caux_i = \rho Cmax_i$ para $0 < \rho < 1$. Computa-se o percentual de pontos dentro da janela de busca que tiveram correlação superior a $Caux_i$. Caso este percentual seja superior a β dado, o ponto não é atualizado. Ou seja, $Caux_i$ é um limiar, de preferência próximo ao valor de correlação máxima, utilizado para se eliminar pontos que obtiveram alta correlação para uma alta percentagem de pontos dentro da janela de busca. Caso isto tenha acontecido, pode-se imaginar que este ponto não foi encontrado de modo bem definido e portanto é melhor não atualizá-lo;
- Caso a posição encontrada esteja fora da imagem, o ponto não é atualizado;

Pontos não atualizados não são imediatamente descartados, porém não são utilizados para se computar o modelo de transformação entre o momento inicial e o modelo atual.

Pontos que não forem atualizados um número k seguido de vezes são descartados da busca. Perceba que isto permite que oclusões temporárias não afetem o procedimento, ao mesmo tempo em que se consegue descartar pontos que não estão bem definidos ou que não são encontrados durante um tempo significativo.

Os pontos que são atualizados têm sua posição (x,y) recalculada a partir da posição de correlação máxima. Estes pontos também têm seu modelo atualizado da seguinte maneira:

- Utiliza-se uma janela $\hat{w}_i$ ao redor da localização atual do ponto de tamanho $(Sx1, Sy1)$;
- Utiliza-se esta janela para atualizar o modelo fazendo $w_i' = \mu w_i + (1-\mu)\hat{w}_i$, onde μ deve obedecer $0 < \mu < 1$, w' é o modelo atualizado e w é o modelo anterior. O valor de μ deve ser próximo de 1 para que o modelo não seja deteriorado rapidamente levando a falsas

PUC-Rio - Certificação Digital Nº 0410268/CA

correspondências do ponto chave. O objetivo de se atualizar o modelo é conseguir se adaptar a pequenas variações de luminosidade e pequenas variações de perspectiva;

Com os pontos de referência calculados, deve-se então obter o modelo de transformação *T* que leva estes pontos da imagem obtida em um momento anterior para um momento seguinte. Isto é visto na seção seguinte.

6.4.4. Obtenção do modelo *T*

Deseja-se encontrar o modelo *T* que mapeie as coordenadas dos pontos de referência de suas posições iniciais para as posições atuais. A matriz *T* e as técnicas utilizadas para tal foram apresentadas no capítulo 4.

Apesar de objetos no ambiente sofrerem variações geométricas independentes do fundo, considera-se que as variações geométricas do fundo do ambiente observado possam ser aproximadas por uma transformação afim, que leva em consideração mudanças de perspectiva. Desta maneira, é esperado que o método proposto consiga eliminar pontos de referência que não estejam associados ao fundo e que possam estar levando à construção de um modelo errôneo de *T*.

Propõe-se que o modelo de transformação *T* seja inicialmente construído a partir da minimização dos erros médios quadráticos de modo a se mapear as posições inicias dos pontos de referência para as posições atuais. A partir de então *T* pode ser refinado com a combinação de uma ou mais das seguintes técnicas já apresentadas no capítulo 4:

- Transformada de Hough;
- RANSAC;
- Reajuste da matriz de pesos W;

A escolha de quais técnicas são usadas deve ser feita de modo a contrabalançar a robustez do modelo encontrado com a eficiência de processamento. Não se pode esquecer que o procedimento está sendo realizado em tempo real (5 a 10 Hz nos experimentos realizados) e que o sistema experimental usado já implica em baixa eficiência computacional.

PUC-Rio - Certificação Digital Nº 0410268/CA

Os pontos de referência que forem considerados inconsistentes um número *k* seguido de vezes devem ser descartados da busca, conforme já mencionado. Estes pontos ou estão errados ou não condizem com objetos de fundo da cena, mas sim com algum objeto que estiver tendo sua posição relativa ao robô alterada de modo diferente dos objetos do fundo da cena.

O modelo *T* encontrado é utilizado para computar as coordenadas atuais dos pontos chaves.

O *Visual Tracking* é assim feito de modo a auxiliar a navegação. A navegação utilizando *Visual Tracking* será detalhada na seção a seguir.

6.4.5. Navegação usando *Visual Tracking*

O principal uso do acompanhamento de pontos chave e pontos de referência nesta dissertação é o de fazer a navegação do robô auxiliada por visão.

A seguir é apresentado como utilizar o *Visual Tracking* de modo a auxiliar a navegação do robô. Serão apresentados dois tópicos semelhantes: como se fazer a correção da direção do robô auxiliada por visão e como navegar para um ponto chave dado auxiliado por visão. Apesar de serem tarefas parecidas, é interessante fazer a explicação das mesmas separadamente.

6.4.5.1. Correção de Direção do Robô

O problema é definido da seguinte maneira:

- Deseja-se fazer com que o robô corrija sua direção auxiliado por pontos de referência e ponto chave dados;
- O ponto chave indica a direção que se deseja obter;
- Os pontos de referência são utilizados para se calcular a localização do ponto chave durante a correção de direção;

A tarefa consiste em se fazer o acompanhamento de pontos de referência na tela de modo a se determinar a cada momento a posição do ponto chave e conseqüente direção desejada do robô. Caso a direção desejada não corresponda à direção real do robô, então o robô deve se movimentar de modo a corrigir sua orientação.

PUC-Rio - Certificação Digital Nº 0410268/CA

Perceba que quanto maior o número de pontos de referência, maior a redundância associada à tarefa e, conseqüentemente, maior a robustez do método. Ao mesmo tempo, quanto maior o número de pontos tratados, maior o custo computacional.

Em linhas gerais, o procedimento é apresentado no diagrama de fluxo da Figura 6-19.

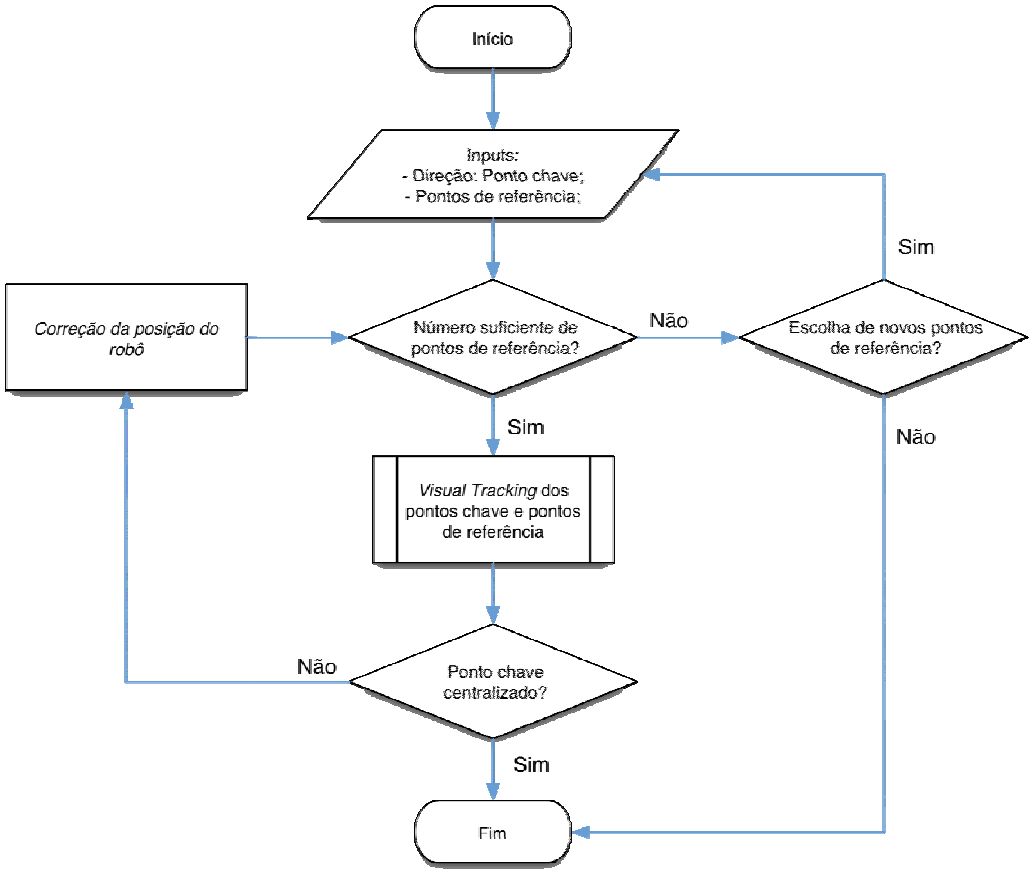

Figura 6-19: Correção da direção do robô

O procedimento de correção é realizado seguindo-se as seguintes etapas:

1. É realizado o *Visual Tracking* dos pontos de referência dados de modo a: encontrá-los na imagem atual; fazer a atualização dos mesmos; e computar a posição do ponto chave que indica a direção da imagem;
2. Caso a direção do robô já esteja alinhada com a direção desejada, o procedimento termina. Caso a direção do robô não corresponda à direção desejada, o robô deve se movimentar para esquerda ou

PUC-Rio - Certificação Digital Nº 0410268/CA

direita de modo a buscar centralizar o ponto chave. A posição do ponto de referência indica a direção desejada para a qual o robô deveria estar apontando. A correção de direção é feita quando o ponto chave se distancia um número η de *pixels* do centro da imagem (sugerido o valor de 5% da resolução horizontal para o sistema experimental utilizado). Esta distância representa o erro da direção do robô em relação à imagem observada;

3. Caso o número de pontos de referência caia abaixo de *m* dado, novos pontos de referência devem ser obtidos;
4. O procedimento recomeça a partir de 1;

Para o caso do sistema robótico utilizado, a correção da direção do robô é feita através de pequenas rotações (aproximadamente 0.5^o). Caso uma nova correção seja computada antes de que um movimento instruído ao robô tenha sido terminado, este movimento é interrompido e nova instrução é dada.

A rotação de aproximadamente 0.5^o é a menor possível para o robô. Esta rotação pode ser grande o suficiente para se ter alguma instabilidade na correção da direção caso o limiar η de erro estipulado seja muito pequeno.

Pontos de referência devem ser constantemente selecionados durante a tarefa. Caso o número de pontos de referência caia abaixo de um número *m* dado, novos pontos de referência podem ser escolhidos de acordo com a necessidade da aplicação.

A seção seguinte descreve o uso de *Visual Tracking* para a tarefa de navegar seguindo uma direção e não somente corrigir a direção.

6.4.5.2. Navegação para Ponto Chave Auxiliado por Visão

Outra tarefa comum de navegação pode ser definida da seguinte maneira:

- Deseja-se fazer com que o robô navegue para um lugar desejado auxiliado por pontos de referência e ponto chave dados;
- A direção ao longo da trajetória pode ser corrigida usando o ponto chave;
- Para auxiliar a tarefa de se obter a localização do ponto chave a cada momento, pontos de referência são utilizados;

PUC-Rio - Certificação Digital Nº 0410268/CA

- São dadas condições para a parada do robô;

O diagrama de fluxo da tarefa pode ser visto na Figura 6-20.

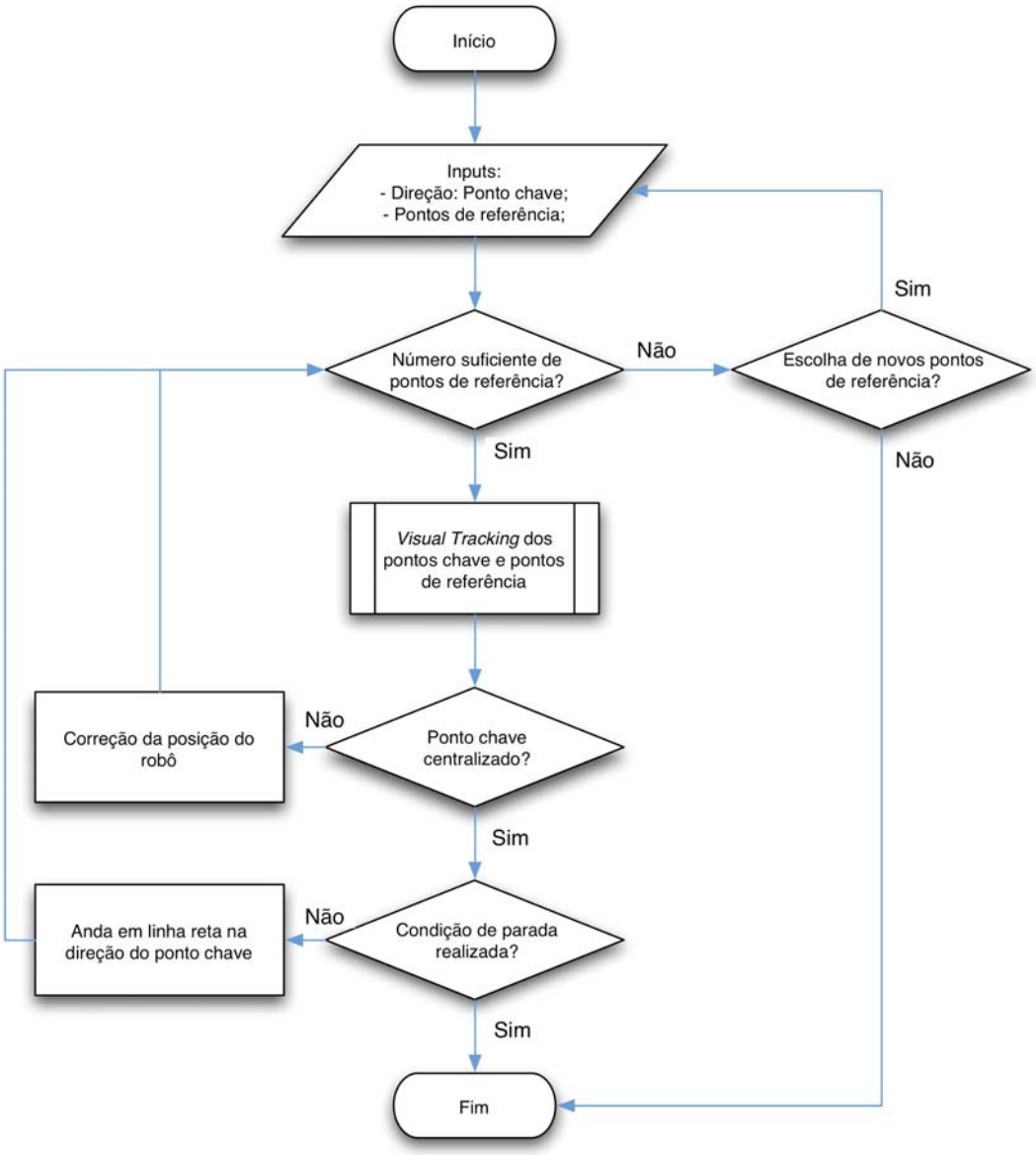

Figura 6-20: Navegação para ponto chave auxiliado por visão

Repare que a tarefa é bem semelhante à correção de direção apresentada, com a diferença de que neste caso, além do robô corrigir sua direção quando incorreta, ele também deve se movimentar de modo a se aproximar do ponto chave.

A cada momento se verifica se a direção está correta e caso não esteja é feita correção. Caso a direção esteja correta, o robô se move para frente até que uma ou várias condições de parada sejam encontradas.

PUC-Rio - Certificação Digital Nº 0410268/CA

Como na tarefa de correção de direção, caso o número de pontos de referência caia abaixo de um número *m* dado, novos pontos de referência podem ser escolhidos para que o procedimento continue.

A seção seguinte apresenta a componente 3 do algoritmo de exploração. Esta componente utiliza a navegação auxiliada por *Visual Tracking* aqui apresentada para fazer com que o robô seja guiado até um nó novo a ser explorado.

6.5. Componente 3 - Navegação para Nó Desconhecido

A navegação para um nó desconhecido consiste em, após ter sido feita a escolha das direções para o qual o robô irá navegar a partir de um nó atual, fazer com que ele siga seu rumo sem perder a direção escolhida e saber o momento de parar.

Os nós desconhecidos são encontrados a partir da segmentação (apresentada na seção 6.3) da imagem panorâmica em um nó conhecido. Esta segmentação gera regiões que foram denominadas de regiões de interesse. O centróide de cada região de interesse define a direção dos nós a serem explorados, denominados de nós desconhecidos.

A direção que o robô deve manter é dada portanto pelo centróide da região de interesse relativa ao nó em questão.

A navegação para um novo nó definido pelo centróide é feita através das etapas apresentadas na Figura 6-21.

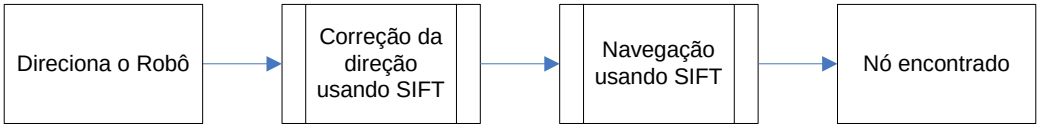

Figura 6-21: Navegação para nó desconhecido

Primeiramente deve-se direcionar o robô para o nó a ser explorado. O robô é, portanto, girado levando-se em conta os parâmetros dados pelos *encoders*. Como a resposta do sistema experimental adotado para os comandos dados possui baixa precisão, é interessante se refinar essa direção a partir das imagens obtidas pela câmera.

PUC-Rio - Certificação Digital Nº 0410268/CA

A correção da direção do robô é feita através da comparação dos descritores SIFT obtidos para a imagem observada e para imagem extraída da panorâmica referente à direção desejada. Este procedimento será descrito com maiores detalhes a seguir.

Com a direção do robô corrigida, deseja-se fazer com que ele navegue até o nó a ser explorado. Ao longo da sua trajetória, o robô deve acompanhar o centróide da região de interesse e fazer um controle visual de modo a corrigir possíveis desvios de direção. Este procedimento também será descrito com maiores detalhes a seguir.

Fica estabelecido que o robô chegou ao nó desconhecido caso o centróide suma da imagem, por cima ou por baixo, ou caso algum obstáculo seja detectado pelos sensores infra-vermelhos (*infra-red*, IR).

A próxima seção descreverá em maiores detalhes como é feita a correção de direção usando de descritores SIFT para se apontar o robô para o local a ser explorado.

PUC-Rio - Certificação Digital Nº 0410268/CA

6.5.1. Correção de Direção Usando a Transformação SIFT

Como a resposta do robô ER1 possui baixa precisão para comandos de movimento, é altamente recomendado que se corrija a direção do ER1 em relação à direção desejada antes de se navegar para um nó desconhecido.

Esta correção será feita com auxílio de *Visual Tracking* como apresentado na seção 0. Para tal, é preciso se estabelecer o ponto chave e os pontos de referência e optou-se por utilizar a técnica SIFT.

O diagrama de fluxo do processo está na Figura 6-22.

Como se deseja que o robô se dirija para uma região escolhida na imagem panorâmica do nó em que ele está, é extraída a imagem da panorâmica relativa ao ângulo θ em que está centralizado o centróide da região. Então, aplica-se a transformação SIFT na imagem extraída e na imagem atual da câmera do robô.

A partir dos descritores SIFT encontrados, é possível calcular o modelo T (verificar capítulos 3 e 4) que mapeia as coordenadas da imagem extraída da panorâmica para a imagem observada. Com T computado, encontra-se a posição

do centróide da região na imagem atual da câmera mapeando-se sua posição na imagem extraída para a imagem atual usando *T*.

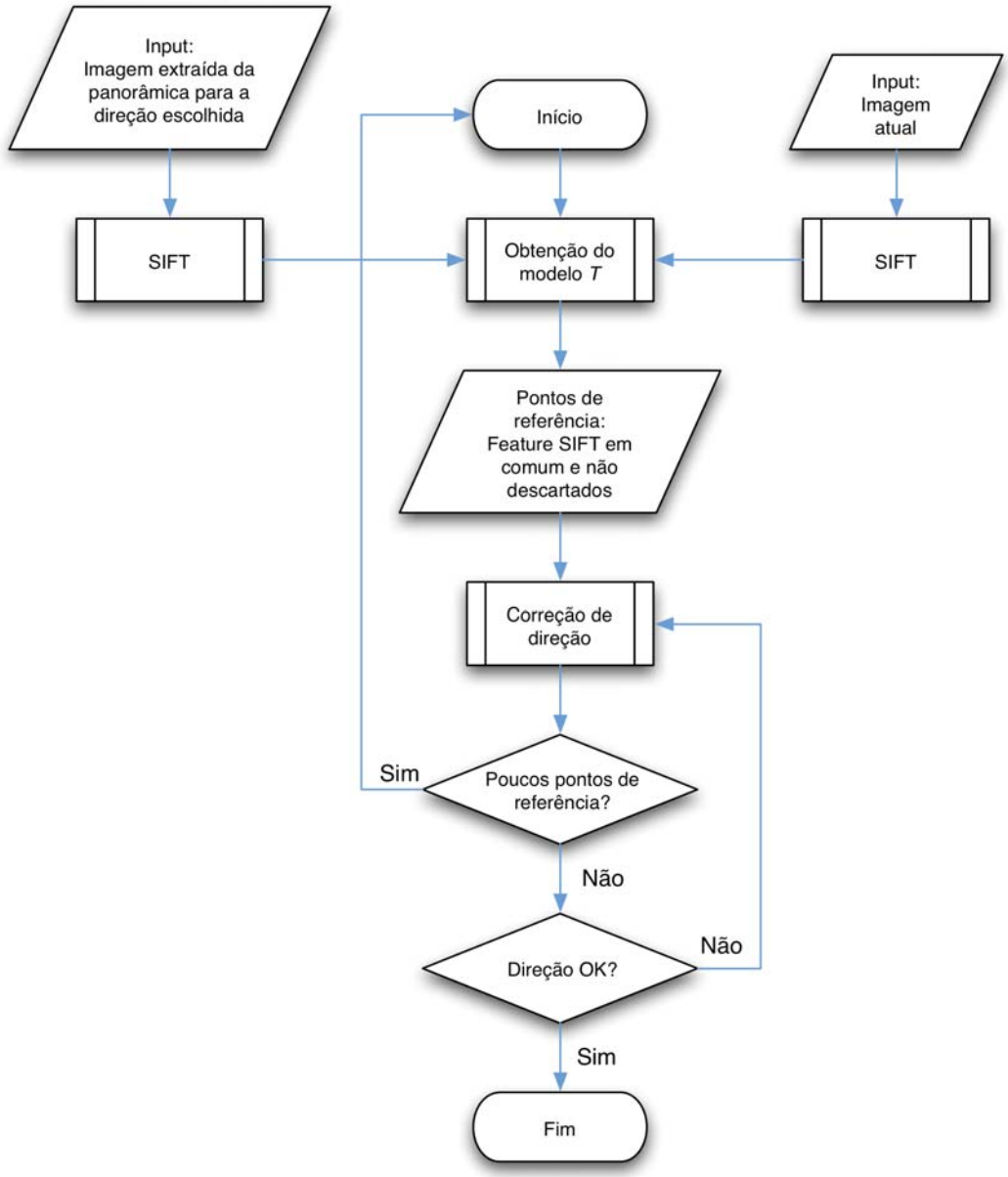

Figura 6-22: Correção da direção usando a transformação SIFT

O ponto da imagem extraída da panorâmica referente à posição do centróide será usado como ponto chave.

Aqueles pontos da imagem atual que foram correlacionados através do método SIFT na imagem extraída da panorâmica serão usados como pontos de referência.

PUC-Rio - Certificação Digital Nº 0410268/CA

A partir de então pode-se fazer a correção da direção com o procedimento apresentado na seção 0.

Caso ao longo da correção o número de pontos de referência caia abaixo de um número *m* dado, o algoritmo é recomeçado, encontrando-se novamente ponto chave e pontos de referência usando a transformação SIFT.

Após a correção de direção do robô, este deve seguir para o nó como visto na seção a seguir.

6.5.2. Seguindo para o Nó Desconhecido

A Figura 6-23 apresenta o fluxograma do procedimento feito para se navegar para um nó desconhecido.

Após o robô já ter sido direcionado corretamente, deseja-se seguir a trajetória até o nó desconhecido. Isto será feito com procedimento semelhante ao aplicado na correção da direção:

- Aplica-se a transformação SIFT na imagem extraída da panorâmica para o ângulo θ dado pela direção escolhida. A imagem é extraída da panorâmica como proposto na seção 6.2.5;
- Aplica-se a transformação SIFT na imagem observada;
- Encontram-se os pontos correlacionados e computa-se o modelo que leva da imagem panorâmica para a imagem observada;
- Para o centróide da região escolhida na imagem panorâmica, calcula-se sua posição na imagem observada usando *T*. Este será o ponto chave;
- Os descritores SIFT obtidos na imagem observada encontrados em comum com a imagem extraída da panorâmica são usados como pontos de referência para se fazer o *Visual Tracking* junto à navegação;

A partir de então o robô procede navegando para o ponto chave com o procedimento apresentado na seção 0. Enquanto não é obedecida nenhuma condição de parada, o robô prossegue. As condições de parada do robô são:

- É encontrado algum obstáculo, ou seja, os sensores IR detectaram algum obstáculo;

PUC-Rio - Certificação Digital Nº 0410268/CA

- O ponto chave está acima de um limiar superior ou abaixo de um limiar inferior da imagem;
- O número de pontos de referência está abaixo de *m* dado;

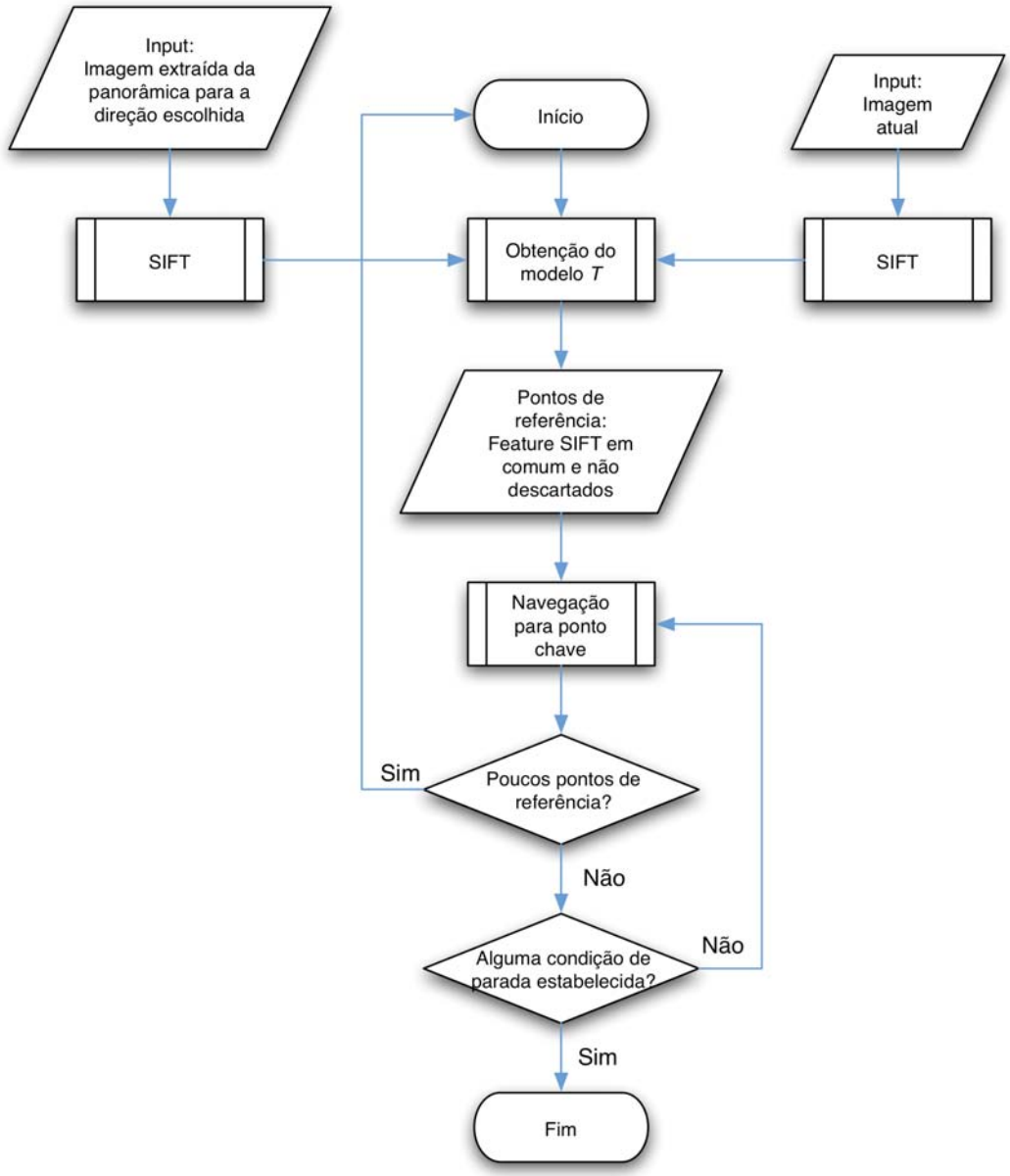

Figura 6-23: Navegando para nó desconhecido usando a transformação SIFT

Para o caso de a condição de parada ser relativa ao baixo número de pontos de referência, pode-se aplicar a transformação SIFT na imagem atual e estabelecer novos pontos de referência comparando-se estes pontos com a transformação SIFT da imagem inicial. O ponto chave pode ser recalculado a partir destes pontos

PUC-Rio - Certificação Digital Nº 0410268/CA

de referência e do modelo *T* calculado. Então, o robô pode continuar seguindo até chegar ao nó desconhecido.

De modo a retornar para os nós já conhecidos, os procedimentos propostos são um tanto diferentes dos apresentados para se navegar para locais desconhecidos. A próxima seção descreve as duas alternativas propostas e estudadas.

6.6. Componente 4 - Navegação para Nó Conhecido

Para se navegar para um nó conhecido, pode-se utilizar a imagem panorâmica do nó conhecido como referência de navegação. A partir da imagem panorâmica, consegue-se extrair uma imagem que representa a visão do robô quando este chegar ao nó conhecido a partir de um nó atual.

A partir de então, duas abordagens diferentes foram dadas ao problema.

A primeira abordagem implementada lidava com a comparação da visão do robô com a imagem de referência utilizando-se transformações invariáveis. Estas comparações eram feitas constantemente e utilizadas para se corrigir a trajetória do robô. A seção 6.6.3 descreve esse procedimento.

A segunda abordagem tomada é parecida com a navegação para um nó não conhecido. A imagem de referência é procurada na visão do robô utilizando-se do método *SIFT* e a partir de então faz-se o acompanhamento dos pontos gerados pela comparação *SIFT* através de procedimento de *Visual Tracking*. A seção 6.6.2 descreve esse procedimento.

A seção seguinte descreve como é obtida a imagem de referência que é utilizada nas duas abordagens implementadas do problema.

6.6.1. Obtendo a Imagem de Referência

É possível se obter da panorâmica do nó pai uma imagem que corresponda à visão que o robô terá ao completar sua navegação para o nó pai. Veja a Figura 6-24. Esta figura apresenta a posição do nó filho, a posição do nó pai, o caminho que deve ser feito de um ao outro e uma representação da imagem panorâmica feita no nó pai. Repare que sabendo-se a direção do nó filho para o nó pai, pode-se

PUC-Rio - Certificação Digital Nº 0410268/CA

extrair uma janela da panorâmica relativa à visão do nó pai na direção dada. A extração desta imagem pode ser feita como descrito na seção 6.2.5.

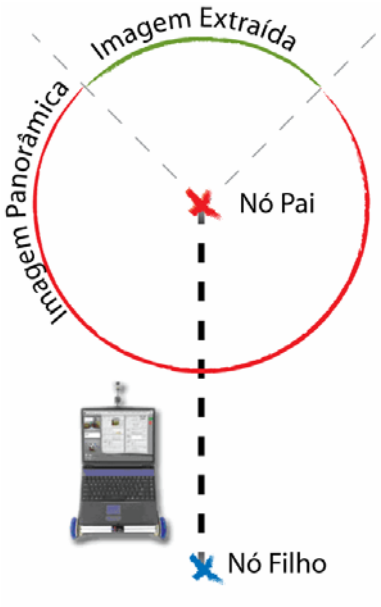

Figura 6-24: Obtendo a imagem de referência

Com auxílio da imagem de referência são propostas duas alternativas de navegação. A primeira estratégia é apresentada na seção seguinte e utiliza *Visual Tracking.*

6.6.2. Navegação Utilizando-se de a técnica SIFT e *Visual Tracking*

A navegação para um nó conhecido utilizando-se o método SIFT é semelhante àquela realizada para nós desconhecidos. A principal diferença está na imagem utilizada para se fazer a busca com a técnica SIFT. Outra diferença está em não ser necessário fazer uma correção de ângulo inicial, pois esta será feita automaticamente ao longo do processo. Além disso, as condições de parada não são as mesmas.

Para se fazer a navegação para um nó desconhecido, uma imagem referente à área de interesse da exploração era extraída da panorâmica do robô no nó atual. Por outro lado, aqui se usa a imagem de referência obtida no nó pai pelo processo descrito na seção anterior.

PUC-Rio - Certificação Digital Nº 0410268/CA

A partir de então o processo segue de modo parecido. O procedimento como um todo pode ser entendido através das seguinte etapas:

- Na imagem extraída da panorâmica é aplicada a transformação SIFT;
- Na imagem observada é aplicada a transformação SIFT;
- Os pontos em comum entre as duas imagens são encontrados e, então, é calculado o modelo que leva da imagem de referência para a imagem observada;
- Usando *T*, é calculada a posição na imagem observada do centro da imagem de referência. O ponto da imagem com esta posição será o ponto chave;
- Os pontos de referência do *Visual Tracking* são dados pelos descritores SIFT obtidos na imagem observada encontrados em comum com os descritores da imagem de referência;

Caso o número de pontos de referência caia abaixo de *m* dado, novos pontos de referência e ponto chave devem ser recalculados e o procedimento de *Visual Tracking* deve ser retomado.

Outra grande diferença está nas condições de parada do robô que não são as mesmas da navegação para um nó desconhecido. No presente caso, espera-se parar o robô no momento em que a imagem observada corresponder à imagem vista. Algumas estratégias foram propostas:

- Determinar a condição de parada a partir do acompanhamento do valor de correlação da imagem vista com a imagem de referência;
- Determinar a condição de parada a partir do acompanhamento do valor de comparação de alguma transformação invariável da imagem vista com a imagem de referência;
- Fazer o acompanhamento do enquadramento da imagem de referência na imagem vista. Quando se encontram pontos em comum entre as imagens, passa a ser possível descobrir qual o enquadramento da imagem de referência na visão do robô. Pode-se então acompanhar este enquadramento ao longo da navegação;

Para as opções em que se faz a comparação da imagem vista com a de referência, se espera que os valores de comparação cresçam até o momento em que o robô alcança na posição desejada, a partir de então estes valores

PUC-Rio - Certificação Digital Nº 0410268/CA

começariam a cair. Neste momento o robô pararia e corrigiria sua posição para que volte a de maior valor de comparação.

Na verdade existem algumas possibilidades para se parar o robô utilizando comparações. Uma delas é que quando os valores de comparação chegarem a um limiar dado, o robô pode parar o movimento. Outra delas é observar o momento em que o valor de comparação começa a subir e proceder como já dito.

Para o caso de se acompanhar o enquadramento, isto é feito da seguinte maneira:

- A posição dos pontos relativos às extremidades da imagem de referência pode ser calculada na visão do robô através do modelo *T* que é computado a cada momento;
- Ao longo da navegação, estes pontos tendem a se aproximar das extremidades da visão observada pelo robô.
- Após o robô passar da posição desejada, estes pontos começam a se afastar das extremidades.

A cada momento se calcula a distância euclidiana média entre estes pontos acompanhados e as extremidades da tela. O valor computado é um valor de erro que deve ser minimizado. Como no caso das comparações diretas entre imagens, pode-se ou estabelecer um limiar para este erro, ou acompanhar este erro e verificar o momento em que este começa a crescer novamente.

Outra possibilidade de navegação experimentada é apresentada na seção seguinte.

PUC-Rio - Certificação Digital Nº 0410268/CA

6.6.3. Navegação Utilizando-se de Comparação de Transformações invariáveis

A navegação para um nó conhecido usando transformações invariáveis tem como base a comparação da imagem de referência com a imagem vista pelo robô utilizando-se da técnica de *Window Growing* apresentada no capítulo 2.

A idéia aqui apresentada admite que a imagem de referência pode ser encontrada como uma sub-janela da imagem atual, ou seja, que a imagem de referência seria uma versão em escala de uma janela centralizada na imagem atual. Portanto, ao se aplicar *Window Growing* pode-se encontrar a sub-janela da

imagem atual correspondente à imagem de referência. Acompanhando o valor de comparação desta sub-janela com a imagem de referência, passa a ser possível determinar se é necessária uma correção na direção do robô e encontrar a direção correta.

A Figura 6-25 apresenta o robô em uma posição inicial e o mesmo em uma posição desejada. Esta figura busca exemplificar a idéia de que a visão do robô em um nó pai pode ser vista em um nó filho como uma sub-imagem do robô no mesmo.

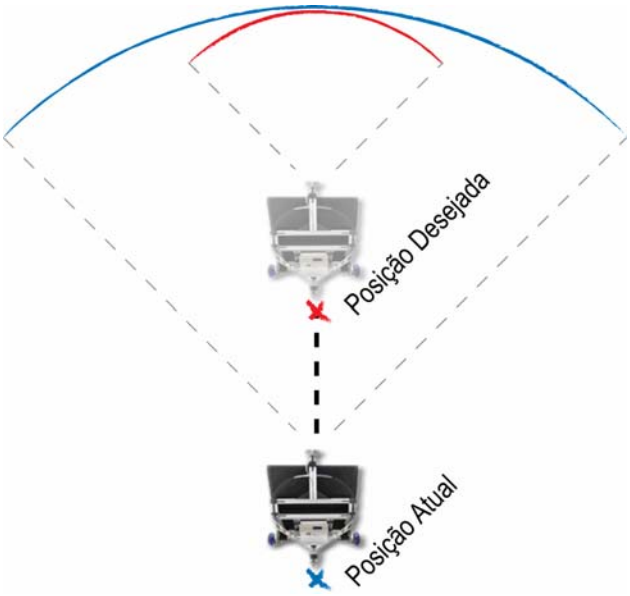

Figura 6-25: Navegação para um nó conhecido

A Figura 6-26 traz outro exemplo em que visões do robô em diferentes distâncias podem ser entendidas de modo bruto como versões em escala uma da outra. No próprio exemplo apresentado, pode-se perceber que esta concepção não é robusta mas sim uma aproximação da realidade.

A partir da idéia apresentada a navegação pode ser compreendida como apresentado no diagrama de fluxo da Figura 6-27.

O procedimento é feito através das seguintes etapas:

- Faz-se um procedimento de correção de ângulos que será apresentado adiante;
- A cada momento se compara a imagem atual com a imagem de referência usando *Window Growing*;

PUC-Rio - Certificação Digital Nº 0410268/CA

- Através da análise do valor das comparações ao longo do tempo, decide-se a necessidade de fazer nova correção de ângulo;
- Quando determinadas condições de paradas forem estabelecidas, a navegação é dada por terminada;
- Ao terminar a navegação, o robô corrige novamente o ângulo e anda para trás até a posição de melhor valor de comparação;

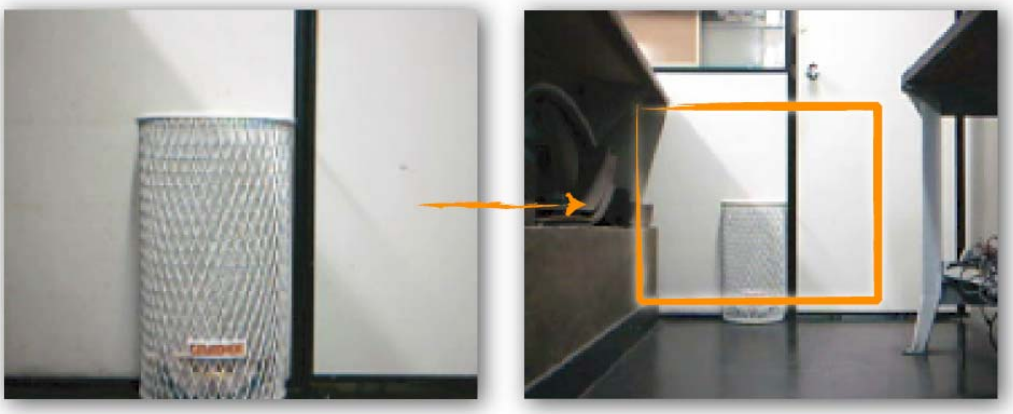

Figura 6-26: Encontrando imagens para diferentes distâncias

A decisão de se fazer uma correção de ângulo é feita verificando-se a cada momento se houve uma queda no valor de comparação. Espera-se que ao longo do tempo, o valor de comparação cresça. Porém, se este começar a decrescer, pode-se admitir que o robô esteja perdendo a direção. Sempre que o valor de comparação for inferior ao valor de um momento anterior dado um percentual escolhido, pode-se fazer uma correção de ângulo.

As condições de parada são as seguintes:

- A janela de comparação encontrada no *Window Growing* deve ser do mesmo tamanho que a imagem inteira;
- O valor de comparação está abaixo do melhor valor de comparação encontrado com relação a um percentual escolhido;

O valor de comparação pode representar o erro de comparação. Neste caso deseja-se encontrar o menor valor e portanto monitora-se a subida do erro;

A correção de direção é feita da seguinte maneira:

- Faz-se com que o robô se movimente um número n de θ graus em uma direção;

PUC-Rio - Certificação Digital Nº 0410268/CA

- Faz-se com que o robô se movimente um número $2n$ de θ graus na direção contrária. Para cada passo dado, uma imagem é obtida;
- O conjunto de $2n$ imagens obtidas é comparado com a imagem de referência utilizando-se *Window Growing*. Aquela imagem que possuir melhor valor de comparação deve representar a melhor direção;
- Por fim a direção é corrigida voltando à posição do robô para aquela onde se obteve a imagem de melhor comparação;

A Figura 6-28 ilustra como se fazer a correção de ângulo utilizando-se comparações de transformações invariáveis.

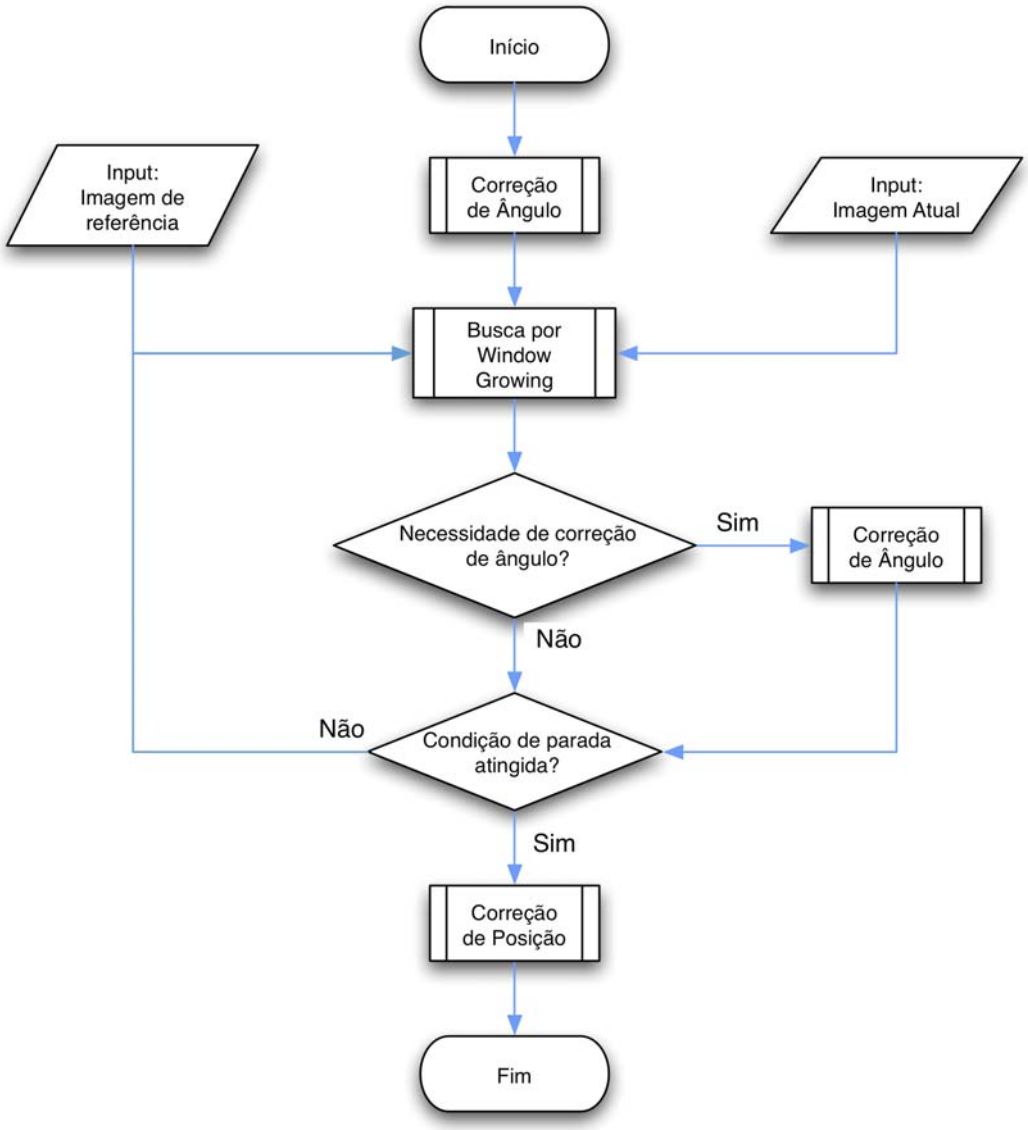

Figura 6-27: Navegação através de transformadas invariáveis

PUC-Rio - Certificação Digital Nº 0410268/CA

Figura 6-28: Correção de ângulo

Esta seção termina de apresentar as quatro principais componentes do algoritmo de exploração proposto. A próxima seção mostra uma quinta componente que pode ser utilizada para refinar o grafo que o algoritmo de exploração gera em uma segunda etapa de exploração. Esta componente só foi testada em simulações.

6.7. Componente 5 - Refinando o modelo criado através de algoritmos genéticos

O modelo criado pelo robô é uma árvore de nós. Cada adjacência carrega a informação de distância entre dois nós e o ângulo entre eles. Cada nó guarda a imagem panorâmica criada naquele ponto do espaço. Esta modelagem possibilitará uma futura navegação baseada em busca de imagens, seja de objetos, lugares, ou outros.

Em uma segunda etapa da modelagem, podemos melhorar o modelo criado. Para tal, buscam-se novas adjacências entre os nós encontrados. Desta maneira, quando houver navegação, o robô não necessitará percorrer muitos caminhos desnecessários para chegar à algum lugar escolhido.

PUC-Rio - Certificação Digital Nº 0410268/CA

Para criar novas adjacências, o robô precisará tentar percorrer o caminho entre os dois nós que a compõem. Em alguns casos, este caminho pode conter obstáculos. Como esta tarefa requer tempo, seria impensável tentar criar todas as adjacências possíveis, para todos os nós. Portanto, seria interessante escolher as adjacências a serem percorridas. Esta escolha seria feita de maneira a otimizar a futura navegação.

Este problema é abordado utilizando o método de Algoritmos Genéticos que não será detalhado aqui, podendo ser verificado em [48-50].

Aqui entra o problema de adicionar novas adjacências em um grafo que será descrito na próxima seção.

6.7.1. Descrição do problema de adicionar novas adjacências em um grafo

Dado o grafo $G = (X, A)$, composto do conjunto de nós X e o conjunto de adjacências A, deseja-se definir um conjunto A' de n' novas adjacências para acrescentá-lo a G, criando-se um novo grafo $G'(X,A+A')$. A escolha do conjunto A' deve ser feita de maneira a minimizar a soma das distâncias dos menores caminhos entre todos os nós:

$$F(x_1,x_2,...,x_n)=\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}g(x_i,x_j)}{2} \tag{174}$$

Onde x_i representa o nó i de X, e $g(x_i,x_j)$ é uma função que retorna a distância do menor caminho entre os nós x_i e x_j.

A função $g(x,y)$ é calculada utilizando o algoritmo A^* [4, 51] que encontra o menor caminho entre dois nós de um grafo.

O conjunto de nós X foi definido neste projeto como pontos em um plano cartesiano com coordenadas (x,y), pois os grafos trabalhados representam um modelo de um ambiente em 2-D. A distância associada a uma adjacência é dada pela distância entre os dois nós que esta adjacência liga:

$$d(i,j)=\sqrt{(x_i-x_j)^2+(y_i-y_j)^2} \tag{175}$$

PUC-Rio - Certificação Digital Nº 0410268/CA

6.7.2. O Algoritmo A*

Para um grafo *G* qualquer, quando as adjacências em *G* possuem um custo e o custo de um caminho é definido como a soma dos custos relativos às adjacências que compõem o caminho, pode ser interessante definir o caminho de menor custo entre um nó inicial N_{init} e um nó objetivo N_{goal}. Um algoritmo clássico para gerar este caminho é o *A** [4, 51].

*O A** explora *G* iterativamente, percorrendo caminhos originados em N_{init}. No início de cada iteração, há nós que o algoritmo já visitou e outros ainda não visitados. Para cada nó *N* visitado, as iterações anteriores produziram um ou mais caminhos conectando N_{init} a *N*, porém o algoritmo só guarda a representação do caminho de menor custo. A qualquer momento, o conjunto de tais caminhos forma uma árvore *T* com o grupo de nós explorados. T é representada associando a cada nó visitado *N* (exceto N_{init}) um ponteiro ao nó parente.

*O A** associa uma função custo *f(N)* para cada nó *N* em *T* atual. Esta função é uma estimativa de custo do custo mínimo para o caminho entre N_{init} e N_{goal} através de *N*:

$$f(N) = g(N) + h(N) \tag{176}$$

Onde:

- *g(N)* é o custo do caminho entre N_{init} em *N* em *T* atual;

- *h(N)* é uma estimativa heurística do custo *h*(N)* do custo do menor caminho entre *N* e N_{goal};

A função *h* é dita admissível se e somente se satisfizer:

$$\forall N \in G : 0 \le h(N) \le h^*(N) \tag{177}$$

Para o problema em questão poderia se fazer *h(N)* igual à distância cartesiana entre *N* e N_{goal} , o que provavelmente faria o algoritmo rodar mais rápido, porém foi adotado *h(N) = 0*. Fica proposto o uso de *h(N)* igual à distância cartesiana.

O algoritmo *A** é feito como a seguir.

As entradas consistem em *G*, N_{init} , N_{goal} , *k*, e *h*, onde $k : X \times X \to R^+$ é a função que especifica a distância de cada adjacência em *G*. Todos os nós são inicialmente marcados como não visitados. O algoritmo faz uso de uma lista denominada *OPEN* que suporta as seguintes operações:

PUC-Rio - Certificação Digital Nº 0410268/CA

- *FIRST(OPEN)*: Remove o nó de *OPEN* com o menor valor de *f* e retorna o mesmo;

- *INSERT(N,OPEN)*: Insere o nó *N* em *OPEN*;

- *DELETE(N,OPEN)*: Remove o nó *N* de *OPEN*;

- *MEMBER(N,OPEN)*: Retorna *verdadeiro* se *N* estiver em *OPEN* e *falso* caso contrário;

-*EMPTY(OPEN)*: Retorna *verdadeiro* se *OPEN* estiver vazio e *falso* caso contrário;

6.7.2.1. Procedimento A*(G, Ninit , Ngoal , k, h)

inicio
    insira *Ninit* em *T*;
    INSERT(N_{init} , OPEN);
    marcar N_{init} como visitado;
    while ! EMPTY(OPEN) do
        N = FIRST(OPEN)
        if N = N_{goal} then sair do *loop while*;
        for todo nó *N'* adjacente a *N* em *G do*
            if N' não visitado *then*
                adicionar *N'* em *T* com ponteiro para *N*;
                f(N) = f(N) + k(N,N');
                INSERT(N',OPEN);
                marcar *N'* como visitado;
                fim;
            else if f(N') > g(N) + k(N,N') then
                f(N) = f(N) + k(N,N');
                modificar *T* redirecionando o ponteiro de *N'* para *N*
                fim;
            fim
        fim
    if N = N_{goal} then
        retorna o caminho construido traçando os ponteiros em *T* de N_{goal} a N_{inti};
    else retorna *falha*;
fim

6.7.3. Aplicando o Algoritmo Genético

Para encontrar *n* novas adjacências a serem adicionadas em um grafo, será empregada um Algoritmo Genético [48-50] com o objetivo de minimizar a soma das distâncias mínimas entre todos os nós após a adição das novas adjacências.

PUC-Rio - Certificação Digital Nº 0410268/CA

6.7.4. Representação do cromossomo

O cromossomo utilizado deve representar as novas adjacências a serem criadas. Algumas possíveis representações são:

- Máscara de bits: Cada gene do cromossomo está associado a uma adjacência do grafo e representa sua adição ao grafo. Se o bit estiver ligado, aquela adjacência é adicionada, se desligado, não. O tamanho do cromossomo deve ser de $k^2/2$, onde k é o número de nós do grafo. O espaço de busca é dado por $2^{\frac{k^2}{2}}$. Tenhamos como exemplo um cromossomo definido como (1-0-0-1). Neste exemplo há 4 possíveis adjacências, sendo que a primeira e a última serão adicionadas ao grafo;
- Representação por inteiros: Os genes são números inteiros representando os nós do grafo. Cada adjacência a ser adicionada ao grafo é representada por um par de genes. O tamanho do cromossomo deve ser de 2n, onde n é o número de nós a serem adicionados ao cromossomo. O espaço de busca é dado por $k^{2\pi}$. Tenhamos o cromossomo definido por ((1 - 2) , (5 - 8)). Neste exemplo, n = 2 e as adjacências entre os nós 1 e 2 e entre os nós 5 e 8 são adicionadas.

A segunda representação foi a escolhida para este projeto. Perceba que diferentes cromossomos podem representar a mesma solução, atrapalhando o desempenho do algoritmo.

6.7.5. Operadores utilizados

Somente dois operadores foram utilizados, *crossover* e mutação.

O operador de crossover utilizado foi o *crossover* de um ponto. Dado um ponto aleatório entre dois genes, os cromossomos pais são divididos e recombinados para gerar os cromossomos filhos. Outros operadores de *crossover* possíveis são o *crossover* de dois pontos e o *crossover* uniforme.

PUC-Rio - Certificação Digital Nº 0410268/CA

O operador de mutação utilizado, sorteava a troca do valor de um gene por um valor aleatório. É proposto, apesar de não utilizado, fazer uma mutação por adjacência. Esta mutação se daria da seguinte maneira. Sorteado um gene para ocorrer mutação, seu novo valor seria um nó sorteado dentre os nós vizinhos ao nó representado por tal gene.

6.7.6. Método de seleção

O método de seleção utilizado foi o método de seleção por roleta, com uso de normalização e *Steady-State* com duplicados.

6.7.7. Avaliação do Algoritmo Genético

Para avaliar os cromossomos gerados, as adjacências são adicionadas ao grafo *G* trabalhado, e então, este grafo é avaliado segundo a seguinte métrica denominada *Eficiência do Grafo*:

$$f(G) = \frac{D(x_1, x_2, ..., x_n)}{F(x_1, x_2, ..., x_n)} \tag{178}$$

Sendo a função *D*, somatório das distâncias cartesianas entre todos os nós. dada por:

$$D(x_1, x_2, ..., x_n) = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} d(x_i, x_j)}{2} \tag{179}$$

A função *f* será máxima e igual a 1 quando ou todos os nós estiverem ligados, ou o caminho mínimo para todos os nós seja uma reta.

Caso algum dos nós não tenha adjacência com nenhum dos outros nós, a função de avaliação retorna 0. Isto permite que o Algoritmo Genético seja utilizado para gerar adjacências em grafos sem adjacências. Desta maneira o A.G. pode solucionar problemas como o de encontrar o melhor conjunto de rotas para um conjunto de localizações. Porém, apesar de possibilitar tal uso, esta avaliação não é a melhor para este tipo de problema, pois o Algoritmo Genético pode demorar muito tempo para encontrar soluções validas (que não retornem $f = 0$) sem conseguir encontrar bons schematas, dado que todos os cromossomos que representam soluções inválidas retornam o mesmo valor, sem diferenciação.

PUC-Rio - Certificação Digital Nº 0410268/CA

6.7.8.
O Programa desenvolvido

Para testar a solução proposta foi desenvolvido um programa em C++ utilizando o programa *Microsoft Visual C++ 6.0*. Este programa permite gerar os grafos testados e visualizá-los, verificar as métricas utilizadas, rodar o algoritmo *A** e rodar o Algoritmo Genético. Além disto, o programa gera curvas de desempenho do Algoritmo Genético e possibilita salvar os resultados para serem visualizados em outros programas, como o *Microsoft Excel* por exemplo. Imagens do programa podem ser vistas em Figura 6-29 e Figura 6-30.

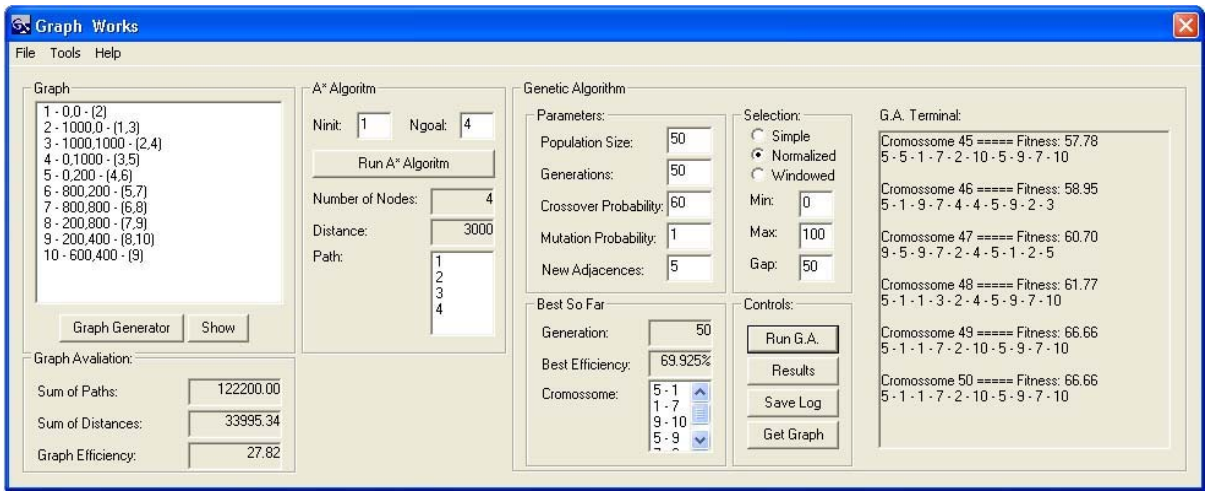

Figura 6-29: Interface do programa desenvolvido

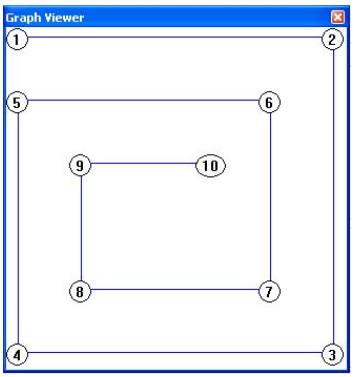

Figura 6-30: Interface de uma das ferramentas do programa desenvolvido

Este programa foi feito com as seguintes limitações:

- Grafos com o máximo de 100 nós;
- Espaço cartesiano limitado em $0 \le x \le 1000, 0 \le y \le 1000$;

PUC-Rio - Certificação Digital Nº 0410268/CA